ISTITUTO ITALIANO
DI TECNOLOGIA

WHOLE-BODY TORQUE CONTROL OF UNDERACTUATED FREE-FLOATING MECHANICAL SYSTEMS

Francesco Romano

Thesis submitted in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Istituto Italiano di Tecnologia
Università degli studi di Genova

April 2016

Francesco Romano: *Whole-body Torque Control of Underactuated Free-Floating Mechanical Systems*.

DOCTORAL SCHOOL:
Life and Humanoid Technologies (XXVIII cycle)

DOCTORAL COURSE:
Robotics, Cognition and Interaction Technologies

SUPERVISORS:
Francesco Nori
Daniele Pucci

LOCATION:
Genova

TIME FRAME:
April 2016

DECLARATION

This work has been carried out by Francesco Romano during his Ph.D. course in *Robotics, Cognition And Interaction Technologies*, from January 2013 to February 2016 at the Italian Institute of Technology, Genova, Italy. He was under the joint supervision of Prof. Francesco Nori, and the additional supervision of Dr. Daniele Pucci. The work has been fulfilled initially at the *Robotics, Brain and Cognitive Sciences Department*, directed by Prof. Giulio Sandini, and subsequently at the *iCub Facility*, directed by Prof. Giorgio Metta. The Ph.D. program has been financially supported by the Italian Ministry of Education, University and Research (MIUR) and the Fondazione Istituto Italiano di Tecnologia (IIT).

Genova, April 2016

Francesco Romano

ABSTRACT

Despite tremendous efforts, the field of robotics is nowadays still far from what has been imagined some decades ago. The fact that the factory environment is highly structured and that industrial manipulators are usually fully actuated, leads to highly automated assembly lines. On the contrary, robots have great difficulties in forcing themselves outside the industrial world. Envisioned applications require robots to be capable of acting autonomously while sharing the same environment of humans. The intrinsically unpredictability of this environment poses great challenges to robotics research, from vision, to cognition, from manipulation to motion control. Humanoid robots are the ideal candidates thanks to their similarity with humans. From the control point of view, difficulties arise in the fact that humanoid robots are intrinsically underactuated and free-floating. It then becomes essential to properly exploit contacts and interaction forces.

This thesis deals with the control of underactuated free-floating mechanical systems, a category comprising, but not limited to, humanoid robots. Contribution has been divided in two main parts. In the first part improvements to classical fixed-base adaptive control and optimal control have been proposed. The second part, instead, focuses on the whole-body control of free-floating mechanical systems. The control framework proposed is responsible of controlling the balancing capabilities of the robot while simultaneously performing other tasks. This control framework has also been implemented, and is currently enabling the iCub humanoid robot to stand and interact safely with humans and the environment.

ACKNOWLEDGEMENTS

In theory, there is no difference between theory and practice.
But, in practice, there is.
— Jan L. A. van de Snepscheut.

Est modus in rebus.
— Horatius, *Sermones*.

I will start this acknowledgement section with the classic disclaimer, that is begging pardon for all the people I will forget to thank and that they should be listed in the following instead. It is not because I want to do it purposely (well, maybe), simply I am becoming older, and my memory is becoming progressively less efficient! Sorry for this.

First of all, I want to thank my supervisor, Francesco Nori. In these three years he has been more than a mentor at work. I have to thank him for the responsibilities and trust he have given to me in this period. While sometimes I have seen the last minute tasks as a "burden", they have given me instead the opportunity to professionally grow. I would not be here without the support and supervision of two other guys: Daniele Pucci and Andrea Del Prete. Daniele taught me a lot about control theory and how to meticulously and rigorously analyse every problem I had to face. He has been a friend before being a colleague and co-supervisor. Most of my work in these three years are due to his support, supervision and encouragement. About Andrea, I have to thank you for all the things you taught me about robotics, for the time spent together and for the opportunity you gave me to be hosted in Toulouse (and I take this chance to thank Roberta too for her support and friendship).

I want to thank all the friends I have in IIT and who have contributed to make this place a wonderful place to work. Thanks to Claudia. I don't know how I managed to become your "bodyguard #1", but I am honoured to have this role even if I think it is a bit undeserved. Thank Cive for all the support you gave me and you are still giving to me, even if from too many kilometres of distance. I have already disappointed you too many times, and I am afraid I will do it a lot more in the future. Please, don't take it personally. Thanks Silvio and Cammo for all the discussions during our daily

trip to work, commenting the best radio out there: Radio 105. Thanks to Fabio, my "new deskmate" and to Luca, the previous one. A big kiss to Casta for all your "craziness". Thanks to Sara, Jorh, Fra (Rea.. too many Francesco in the lab), Chiara, Matte, Ilaria, Ugo, Vadim, Randaz, Sean, Ilaria (the Roman one) and Giulia.

Even if we meet less often than I desire, I want to thank all my friends outside IIT. Thanks to Diego, Fra & Monica, Giuse, Sara, Mirco, Albe, Andrea, Tommy, Benve, Aigor, Matte and all the others guys from the university. Thanks to all the basketball team, but especially to Robbi and Milos for the Friday nights we spent together. Lastly I want to thank Pines, Isi and all the (almost infinite) friends I have made since I was young. I would probably need ten pages if I had to name them all. This is why I will simply thank *the group of the Equipe*.

These last lines are all Cive's fault. He said I must add some kind of easter egg in this thesis if I don't want to be unlucky for all the rest of my life. Being very superstitious I have to compel to this order. This is why I added something very easy to find, but probably difficult to understand. If someone finds it he deserves one beer, or equivalent. If someone also manages to understand it, two beers will be the prize. Anyway, even if you don't find it we can have a drink together anytime.

CONTENTS

Part I

INTRODUCTION

1 MOTIVATIONS AND CONTRIBUTIONS

1.1 MOTIVATION

Nowadays, robotics is successfully employed in the industrial world, and it constitutes the key ingredient for mass production in modern societies. Since industrial robots operate in structured, known and ad-hoc environments, they are highly performant in their predefined and limited tasks. Lately, the need of conceiving systems capable of cooperating with humans has demanded research to focus on safety and compliance, thus endowing industrial manipulators with the ability to work together with humans in the same space.

When the robot has to perform outside factories, different issues come into play. The environment is no longer structured or known a-priori. Terrain can be uneven, and with the presence of obstacles. Light is not controlled, thus penalizing the performances of both teleoperation and artificial vision. Human-Robot Interaction further complexifies the scenario as human behaviour is unpredictable, thus endangering both the robot and the human itself.

Depending on the envisaged application for the robot, some of these issues are more pressing than others. For example, one important use for modern robotics is as military or disaster-response robots. This kind of application requires robots to be robust versus the environment, to reliably traverse an obstructed or harsh area, to be versatile in the number of tasks they can accomplish. Even if teleoperation is still the main way to remotely control the robot, a certain level of autonomy is required as the robot can enter areas where telecommunications are degraded. As a matter of fact, robots are also used to clear minefields [85], or to support soldiers in the field. After the Fukushima nuclear disaster in 2011, a big push has been in the use of robots during nuclear fallouts or earthquakes or in general to help or substitute humans during operations in hostile environments.

A completely different application views the robots side by side with humans executing tasks useful to our daily life. This is usually called *service robotics*. Robotic major-domos will per-

Figure 1.1: Industrial robots in an assembly line. It is dangerous for humans to work side-by-side with the robots.

form house-holding tasks, such as cleaning, thus assisting and saving time to humans and improving their quality of life. Furthermore, with the progressively ageing of the society, more people will be requested to provide assistance to elderly, or as healthcare assistants. Robots could fill this gap. For this kind of applications, robots have to operate autonomously or semi-autonomously. Safe interaction with humans is also of crucial importance.

Rehabilitation robotics is also profoundly linked, in requirements, to service robotics. Indeed, exoskeletons and prostheses take to the extreme the concept of interaction with humans.

Humanoid robots are the ideal choice as service robots. From a psychological point of view, their look facilitates the interaction with humans [96]. But from an engineering standpoint, their increased mobility constitutes an additional advantage. Differently from wheeled robots, biped robots are potentially able to directly share the human environment, e.g. climb stairs or ladders or move in narrow passages. This advantage has been the main motivation behind the attention they have received from the robotics community. Conversely, legged locomotion arises interesting issues, which must be solved before humanoids can be successfully deployed in the human environment. Indeed, the recent DARPA Robotics Challenge, held last June, has clearly highlighted these difficulties. Most of the failures during the trials were due to a not perfect balancing or walking.

All the applications listed above share some common elements. Because of the robot mobility, they are classified as free-

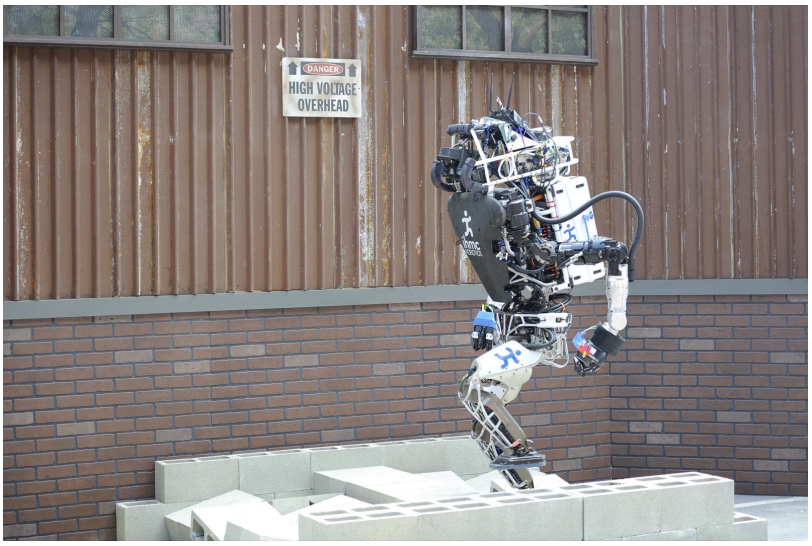


Figure 1.2: IHMC Team during the Rubble trial at the DARPA Robotics Challenge Finals.

floating systems, i.e. their configuration cannot be fully determined by only their internal variables. To explain better the problem free-floating systems pose, let's consider the first cardinal equation of classic mechanics: *the acceleration of the center of mass can be changed only by external forces*. Internal forces, thus, do no have any influence on the "global" motion of the system. Differently from space robots, humanoid robots can exploit the contacts with the environment to produce external forces and thus generate motions. This is, in fact, what humans do.

The pose of the base is, in general, not directly actuated, rendering the system an *underactuated system*. Underactuated mechanical systems raise specific issues when attempting the control of the entire state space.

Finally, state-of-the-art humanoid robots are highly redundant, that is, in general, they possess more degrees of freedom than it is required to accomplish a specified task. On one hand, this feature opens the possibility to fully exploit humanoids to achieve multiple tasks at the same time, as we humans do. On the other hand multiple tasks can conflict with each other. For example a humanoid robot has to keep balance, while performing manipulation tasks. Tasks coordination and conflict resolution becomes thus a critical issue in robotics. Whole-body control is a control formulation in which the whole robot is taken into account during the execution of a single task, with all the possible constraints and by considering all the tasks the robot should execute.

Citing the definition from the RAS Technical Committee [36]
[...] robots have become increasingly proficient in performing many

different, non-trivial tasks [...] In most cases, however, each of these tasks is addressed individually [...] Whole-Body Control (WBC) has been proposed as a promising research direction. WBC aims to i) define a small set of simple, low-dimensional rules (e.g., equilibrium, self collision avoidance, etc.) ii) that are sufficient to guarantee the correct execution of any single task, whenever feasible [...], and of simultaneous multiple tasks [...], iii) exploiting the full capabilities of the entire body of redundant, floating-based robots in compliant multi-contact interaction with the environment. [...]

1.2 STATE OF THE ART

We divide the description of the state of the art in two main sections thus following the same structure of the contributions of this thesis.

1.2.1 NONLINEAR CONTROL STRATEGIES

Control of free floating, highly-redundant, underactuated mechanical systems poses various concerns to the control community. For instance, assuming that the underactuated system desired configuration is feasible, the nature of a stabilizing controller for this configuration is intimately related to the nature of the system itself. In particular, mechanical systems without potential terms in general forbid the existence of time-invariant feedback continuous stabilizers [86]. This claim, which follows from an application of Brockett's Theorem [7], has motivated the development of discontinuous and/or time-varying feedback stabilizers for specific classes of underactuated systems (see, e.g., [14, 26, 28, 78]).

The complexity of the control problem associated with underactuated mechanical systems reduces when attempting to stabilize only a subset of the system degrees of freedom. In the specialized literature, several methods have been proposed to achieve this objective. Inverse dynamics [15, 101], sliding mode [92], and energy based techniques [100] are among the main tools exploited by these works.

Model-based control techniques rely, as the name suggests, of a representation of the dynamical systems to be controlled. The level of details required from the model, and the underlining hypotheses, greatly vary in the different strategies. Usually the more information is used, the more the control system is tuned for a specific application, and as a consequence performances are usually better. Unfortunately the model will never be per-

fect and a mismatch between the real plant and the assumed one has to be expected.

To cope with model uncertainties, different techniques exist nowadays. Robust control [1], model-reference adaptive control (MRAC) [54], and adaptive control are some of them. The adaptive control of generic dynamical systems, has received much attention from the control community. Most works in the specialized literature make specific assumptions on the relationship between the system dynamics and the set of parameters that characterize it. More precisely, adaptive control of feedback linearizable systems is feasible [93]. This work, however, assumes that the dynamics can be expressed linearly with respect to the system parameters, and this is not the case for the collocated dynamics of an underactuated mechanical system. An attempt to the adaptive control of nonlinearly parametrized systems can be found in [84]. But the assumption that there exists a parameter independent input ensuring global stability irrespective of the parameters complexifies the application of this theory to our case.

When considering the specific class of mechanical systems, adaptive stabilizations of the collocated and noncollocated dynamics can be achieved [29]. The main drawback of the approach is that the measurement of the system acceleration is required by the feedback action. Leaving aside causality issues, this measurement may not be always available.

In the case of fully actuated mechanical systems, adaptive stabilization of time varying reference trajectories can be achieved [99, 102]. The key assumption is that the system inverse dynamics can be expressed linearly with respect to a set of constant *base parameters*. The extension of these works to the underactuated case is not straightforward.

Optimal control theory is another technique that in the last decades has received major interests. In the case of linear plants, or when linearization is a viable solution, and no constraints are considered, the Linear Quadratic (Gaussian) Regulator has become the de-facto control method to stabilize MIMO (Multiple Inputs Multiple Outputs) systems. When nonlinear systems are considered instead, the lack of analytical solutions to the optimal control problems greatly complexifies its application. Indeed, Dynamic Programming becomes computational intractable with moderately high state and control space dimensions. Some approaches [39, 107] apply the dynamic programming principles to local approximations of the nonlinear system along the state trajectory evolution. A recent application

of [107] for the whole-body control of a humanoid robot can be found in [50].

Methods based on the Maximum Principle, or on the discretization of the continuous problem into a NonLinear Program (NLP), instead yield open loop control laws and feedback must be closed by other means, such as by continuously reiterating the optimization procedure. In [30], optimal control has been used to generate online walking trajectories by controlling the linearized dynamics of the center of mass of the humanoid robot. Because constraints are added to the formulation, LQR is not directly applicable and the problem is discretized into a QP.

1.2.2 WHOLE-BODY CONTROL AND ROBOT BALANCING

Given the complex nature of highly redundant mechanical systems, such as humanoid robots, researchers have progressively shifted from classic nonlinear control techniques to optimization-based solutions. These optimization-based control techniques allow to specify, at a high level, the tasks to be accomplished by the robot and to transfer the responsibility of finding the solution to the underlining solver. One disadvantage of this process is that properties of the solution can be more difficult to find. Indeed, if some properties have to be enforced, they can be specified by means of constraints or optimization criteria. But how this high-level criteria influence the solution is not trivial and depends on the application.

Legged locomotion, and especially biped locomotion, does pose interesting issues. The necessity to balance, meaning as avoiding to fall to the ground, becomes of primary importance for a biped. Indeed, it can be compared to an inverted pendulum at its unstable equilibrium point.

Most of nowadays approaches make use of a simplified model of some sort to describe the balancing task. Some of them approximate the humanoid robot with simpler models, such as the linear inverted pendulum [41] and its derivations. Posture stability criteria such as the Zero Moment Point [110] or the Capture Point [81] are then used. Other approaches instead do not approximate the full robot dynamics but they try instead to control a different representation, such as the robot momenta [42].

One of the first works in humanoid balancing is [35]. In this work the authors control the center of mass position through the use of the contact forces. They explicitly avoid the use of inverse dynamics techniques and the commanded torques are

generated such that in the static configuration a gravity compensation term plus a dissipative term is applied. In this paper it is highlighted the problem of redundancy occurring when the contact forces are greater than one. Their solution, as it is for most of the works, is to select the minimum L_2 norm solution, i.e. it can be obtained with the pseudoinverse.

A similar approach is the one used by [76]. As in [35] they do not use any inverse dynamics algorithms. The forces are obtained with a gravity compensation term plus a term to control the linear momentum. They do not control the angular momentum, but instead they try to control the orientation of the hip link.

More recent approaches are all based on the solution of a quadratic problem (QP). A quadratic problem is an optimization problem in which the cost functional is a quadratic form of the optimization variables. Constraints, both equality and inequality, can be specified in a linear form. Analytical solutions exists if no inequalities are specified. Otherwise iterative algorithms must be used.

In the balancing problem formulation most of the contact stability constraints, such as the center of pressure, yield a linear inequality constraint. On the contrary, friction cone constraints give rise to conic inequality constraints. To treat this kind of constraints a conic solver should be used, as in [114]. More often, instead, they are approximated so to fit in the linear case.

Numerous works use the full QP formulation. In [52] the authors optimizes the momenta error, i.e. the error between the robot momenta and the desired one, together with a regularization term on the contact forces and joint accelerations. Hard constraints consist in the momenta equation and in the desired tasks to be accomplished specified as desired joint accelerations. Once a solution, in term of joint accelerations, is found, it is transformed into joint torques through inverse dynamics.

Similar approaches are the one of [21, 33], but they integrate the solution once, to obtain joint velocity references, or twice, to obtain joint positions references, to be commanded to the robot.

In [103] the authors make a division between a single contact or multiple contacts. Because in their formulation the desired momenta is imposed as a hard constraint, in case of single foot support, the resulting forces can be unfeasible with respect to (w.r.t.) the friction cone and center of pressure constraints. In this case the desired momenta constraint is relaxed and the problem is solved in the least squares sense.

On the contrary, [55, 113] modify the optimization problem when more than one contact is present. In particular they

choose the redundancy of the forces as to minimize the ankle joint torques.

In [53] a preliminary step is taken before solving the QP. A time-varying LQR problem is formulated as a tracking problem for the (linear) ZMP dynamics. The explicit expression for the optimal value function is then used as a cost function in the instantaneous quadratic problem.

A different approach when multiple tasks are present is to organize them in an strict hierarchy. Each level of the hierarchy defines the priority of the task. Multiple tasks within the same level of priority can be specified together in a single cost function. The peculiar feature of this approach is that higher priority tasks are put as hard constraints during resolution of lower priority tasks. This implies that a task is feasible only if it does not change the optimality of all higher priority tasks. Indeed, in case of conflict between two or more tasks, we may desire that the most important task is achieved, at the expenses of the others. This control approach — known as prioritized or hierarchical control — has been used in robotics and computer animation since the 80's [70]. Researchers have applied prioritized control in different forms. Nakamura et al. [70] and Siciliano et al. [98] used it to control the cartesian velocity of multiple points of a robotic structure. Sentis [95] was the first one to apply this technique on the robot dynamics, extending the Operational Space formulation [47]. This allowed the control of contact forces, besides cartesian and joint-space motion. In recent years, new formulations [87, 67] have contributed to improving the computational efficiency of this approach. Mansard, Escande and Saab[90, 19] have studied efficient ways to include inequalities in the problem formulation, which can be used to model joint limits and motor torque bounds. Research in computer animation [12, 13] has followed a similar path, trying to generate artificial motion by solving one or more Quadratic Programming (QP) problems. Stability of the joint space after the convergence of the task space error dynamics, i.e. the so called zero dynamics, is an interesting issue which is still under investigation [80, 77].

The application of the task hierarchy to the balancing case has been used in [32, 97, 114, 87]. The common denominator between these works is the fact that the stack of tasks is composed by i) physical consistency, i.e. the momenta equations, together with center of pressures and friction cone constraints; ii) the balance and/or motion tasks; iii) the so called *postural task*, i.e. a task which removes any redundancy left with the aim to stabilize the zero dynamics.

1.3 CONTRIBUTIONS AND THESIS OUTLINE

The present thesis tries to cover the gaps resulting from the analysis of the state of the art in the previous section. In particular the following shortfalls can be highlighted.

- The application of the adaptive control to an underactuated mechanical system still presents some pitfalls, e.g. it requires measurement of the acceleration.
- While instantaneous prioritized control is widely applied, strict tasks hierarchy in the optimal control setting has never been investigated.
- Whole-body control has become the major trend for the control of humanoid robots. Nevertheless all the implemented approaches in literature present some variations, and no analysis on the properties of the obtained control solutions has been performed yet.

1.3.1 THEORETICAL CONTRIBUTIONS

Model-based control algorithms should be robust to model uncertainties. In the case the system parameters are unknown, or not known with enough precision, an estimation procedure can be put to work. One class of this algorithms goes under the name of *Adaptive Control*. Application of adaptive control to underactuated mechanical system is still an open problem. In Chapter 4 we present an approach to control a subset of an underactuated system degrees of freedom.

Optimal control has lately received increasing attention in the robotics community. In the optimal control formulation tasks are usually specified in the cost function as a weighted sum. To solve potential conflicts weights must be chosen carefully depending on the application. This can become a tedious procedure as soon as the number of tasks increases. As in the instantaneous control setting *prioritized control* provides an alternative to task weighting, we extends optimal control to include strict task prioritization. The algorithm, called *Prioritized Optimal Control*, together with results in simulation, is presented in Chapter 5.

State-of-the-art balancing controllers for humanoid robots are optimization based. A quadratic program (QP) is composed by specifying a quadratic cost, usually composed of the tasks to be accomplished, and linear equalities and inequalities constraints such as dynamic feasibility and contacts stability. Solutions of

this problem are then obtained at high frequency rate, i.e. usually at least 100 Hz and the resulting torques, or positions, are commanded to the robot. Part III is devoted to the description of the momentum-based whole-body torque control and its implementation in the iCub humanoid robot. In particular, Chapter 6 describes the algorithm and the control objectives. While often neglected in literature, the choice of the contact forces is of crucial importance. In Chapter 7 we analyze two of the infinite different choices available and their implication on the performances on the robot. Lastly Chapter 8 presents a stability analysis of the zero dynamics in momentum-based stack of tasks control algorithms.

1.3.2 PUBLICATIONS

This work has been carried out during my Ph.D. course in *Robotics, Cognition and Interaction Technologies*, from January 2013 to February 2016. The three-year project resulted in the following publications (at the time of writing):

- F. Romano, A. Del Prete, N. Mansard, and F. Nori. Prioritized optimal control: A hierarchical differential dynamic programming approach. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 3590–3595, May 2015. doi: 10.1109/ICRA.2015.7139697
- A. Del Prete, F. Romano, L. Natale, G. Metta, G. Sandini, and F. Nori. Prioritized optimal control. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2540–2545, May 2014. doi: 10.1109/ICRA.2014.6907214
- D. Pucci, F. Romano, and F. Nori. Collocated adaptive control of underactuated mechanical systems. *Robotics, IEEE Transactions on*, 31(6):1527–1536, Dec 2015. ISSN 1552-3098. doi: 10.1109/TRO.2015.2481282

This thesis provides a structured discussion on top of these aforementioned papers, by providing a better understanding of the overall contribution of this Ph.D. project. As such, some ideas and figures have already appeared in those publications.

During the three-year project I had the opportunity to work on topics different from the whole-body control. These works have led to the following publications which have not been included in the thesis.

- F. Romano, L. Fiorio, G. Sandini, and F. Nori. Control of a two-DoF manipulator equipped with a pnr-variable stiffness actuator. In *Intelligent Control (ISIC), 2014 IEEE International Symposium on*, pages 1354–1359, Oct 2014. doi: 10.1109/ISIC.2014.6967620

- L. Fiorio, F. Romano, A. Parmiggiani, G. Sandini, and F. Nori. Stiction compensation in agonist-antagonist variable stiffness actuators. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014

- L. Fiorio, F. Romano, A. Parmiggiani, G. Sandini, and F. Nori. On the effects of internal stiction in pnrVIA actuators. In *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, pages 362–367, Oct 2013. doi: 10.1109/HUMANOIDS.2013.7030000

- E. Mingo, S. Traversaro, A. Rocchi, M. Ferrati, A. Settimi, F. Romano, L. Natale, A. Bicchi, F. Nori, and N.G. Tsagarakis. Yarp based plugins for gazebo simulator. In *2014 Modelling and Simulation for Autonomous Systems Workshop (MESAS)*, Roma, Italy, 5 -6 May 2014, 2014

- F. Nori, S. Traversaro, J. Eljaik, F. Romano, A. Del Prete, and D. Pucci. iCub Whole-body Control through Force Regulation on Rigid Noncoplanar Contacts. *Frontiers in Robotics and AI*, 2(6), 2015. ISSN 2296-9144. doi: 10.3389/frobt.2015.00006

1.3.3 SOFTWARE

Development of software libraries and applications is a key element of the work done during the Ph.D. course. Almost all the software developed has been released as open source software and hosted on GitHub in the *Robotology* organization. The software developed consists in, but is not limited to,

- contribution to the development of the YARP middleware
- development of control- and dynamics- related libraries
- development of the whole-body torque control balancing application for the iCub humanoid robot.

2 BACKGROUND

In this chapter, we introduce the mathematical tools necessary for the modelling, analysis and control of mechanical systems.

We first introduce the general notation used throughout the thesis. Notation specific to particular topics will be introduced when needed. We then proceed to describe the dynamical model of free-floating mechanical systems, some of its useful properties, and a description of rigid contacts. Lastly we introduce the optimal control theory and we briefly describe the state of the art of the most diffuse methods in the optimal control formulation.

2.1 NOTATION

The following notation will be used throughout the whole thesis.

- We denote with $\mathbb{R}$ the set of real numbers
- Given a vector $u \in \mathbb{R}^m$, $\|u\|$ is its euclidean or 2-norm
- Given two orientation frames A and B, and vectors of coordinates expressed in these orientation frames, i.e. ${}^A p$ and ${}^B p$, respectively, the rotation matrix ${}^A R_B$ is such that ${}^A p = {}^A R_B \, {}^B p$
- We denote with $1_n \in \mathbb{R}^{n \times n}$ the identity matrix of dimension n
- Given two vectors $x, y \in \mathbb{R}^n$ we denote with $x^\top y \in \mathbb{R}$ their inner product.
- We denote with $S(x) \in \mathbb{R}^{3\times 3}$ the skew-symmetric matrix such that $S(x)y = x \times y$, where $\times$ denotes the cross product operator in $\mathbb{R}^3$. If $x = (\bar{x}^\top, 0)^\top$ and $y = (\bar{y}^\top, 0)^\top$, with $\bar{x}, \bar{y} \in \mathbb{R}^2$ one has $S(x)y = -\bar{x}^\top S \bar{y} e_3$, with $S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$.

THE SPECIAL ORTHOGONAL GROUP, SO(N) The special orthogonal group is defined as

$$SO(n) = \{R \in \mathbb{R}^{n\times n} : RR^\top = 1_n, \ \det(R) = 1\}$$

If $n = 3$ the group is also called *rotation group* in $\mathbb{R}^3$.

Given an element $R \in SO(3)$,

$$\exists\, \iota \in \mathbb{R}^3 : R = \exp(S(\iota))$$

with $S(\iota) \in \mathbb{R}^{3\times 3}$ the skew symmetric matrix defined as

$$S(\iota) = \begin{bmatrix} 0 & -\iota_3 & \iota_2 \\ \iota_3 & 0 & -\iota_1 \\ -\iota_2 & \iota_1 & 0 \end{bmatrix}.$$

$S(\iota)$ is the Lie algebra of $SO(3)$ and it is also denoted as $\mathfrak{so}(3)$.

The interested reader can refer to more detailed texts about Lie groups and differential manifolds [69, 94, 112, 108].

2.2 NEWTON-EULER EQUATIONS AND THE ROBOT MOMENTA

The equations of motion of an articulated body are often presented by using the Lagrangian formalism. Conversely, the Newton-Euler equations are more typically used when a rigid system is considered.

In literature it is quite common to find the control of the robot momenta as a high-level control task. One interesting question arising at this point is the following.

How are the momenta of an articulated body defined?

To properly introduce the concept of momenta of an articulated system, we start by describing first the dynamic equations of a system of particles and how they can be used to derive the momenta of an articulated rigid body.

2.2.1 DYNAMICS OF A SYSTEM OF PARTICLES

Consider an inertial frame $\mathcal{I}$ with origin O, and N particles P_i each of constant mass $\hat{m}_i$. Each particle is identified by its position vector $x_i \in \mathbb{R}^3$ w.r.t. $\mathcal{I}$, i.e. $P_i - O$, and is subject to external forces, with resultant indicated with F_i, and internal forces f_{ij}, that is the force exerted by P_j on P_i. The resultant of the internal forces on P_i is thus $\sum_{j=1}^{N} f_{ij}$.

Assume a newtonian system, i.e. the third axiom of the classical mechanics is satisfied:

$$\begin{aligned} f_{ij} &= -f_{ji}, \\ r_{ij} \times f_{ij} &= 0, \quad \forall\, i, j = 1, \cdots, N \end{aligned} \tag{2.1}$$

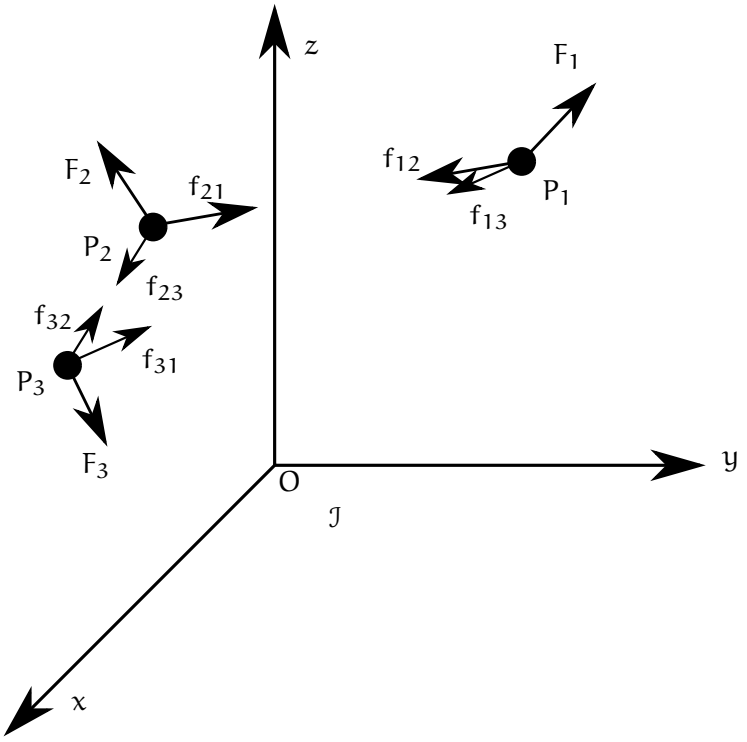


Figure 2.1: System of three particles. External and internal forces are shown

where $r_{ij} \in \mathbb{R}^3$ is the vector going from P_j to P_i, i.e. $P_i - P_j$. Let $Q \in \mathbb{R}^3$ be any point in $\mathbb{R}^3$, and denote with $r_i^Q \in \mathbb{R}^3$ the vector going from Q to P_i, i.e. $P_i - Q$. Equation (2.1) in turn implies the following properties:

$$R^{(int)} := \sum_{i=1}^{N} \sum_{j=1}^{N} f_{ij} = 0,$$

$$M_Q^{(int)} := \sum_{i=1}^{N} \sum_{j=1}^{N} r_i^Q \times f_{ij} = 0, \quad \forall \text{ point } Q.$$

In other words, the resultant of the internal forces and moments are equal to zero.

Define now the center of mass of the system of particles as

$$x_G := \frac{\sum_{i=1}^{N} \hat{m}_i x_i}{\sum_{i=1}^{N} \hat{m}_i}.$$

By differentiating the definition of the center of mass and by defining with $m = \sum_{i=1}^{N} \hat{m}_i$ the total mass of the system we obtain the definition of the linear momentum

$$\frac{d}{dt}(m x_G) = \frac{d}{dt}\left(\sum_{i=1}^{N} \hat{m}_i x_i\right),$$

$$m v_G = \sum_{i=1}^{N} \hat{m}_i v_i =: P \quad \text{linear momentum.} \tag{2.2}$$

With respect to an inertial frame $\mathcal{I}$, each point satisfies the second Newton law, i.e.

$$F_i + \sum_{j=1}^{N} f_{ij} = \hat{m}_i a_i, \quad \forall\, i = 1, \cdots, N. \tag{2.3}$$

If we sum them over the index i we obtain

$$\sum_{i=1}^{N} F_i + \sum_{i=1}^{N}\sum_{j=1}^{N} f_{ij} = \sum_{i=1}^{N} \hat{m}_i a_i,$$

$$R^{(ext)} + R^{(int)} = \sum_{i=1}^{N} \hat{m}_i a_i,$$

$$R^{(ext)} = \frac{dP}{dt} \tag{2.4}$$

that is the resultant of the external forces is equal to the derivative of the linear momentum.

We define now the angular momentum with respect to an arbitrary point Q. The angular momentum is thus defined as

$$L_Q = \sum_{i=1}^{N} r_i^Q \times \hat{m}_i v_i \tag{2.5}$$

which can be seen as the resultant of the moment of the linear momentum. We can also differentiate the angular momentum w.r.t. time obtaining the following expression:

$$\begin{aligned}\frac{dL_Q}{dt} &= \frac{d}{dt}\left(\sum_{i=1}^{N} r_i^Q \times \hat{m}_i v_i\right) \\ &= \sum_{i=1}^{N} (v_i - v_Q) \times \hat{m}_i v_i + \sum_{i=1}^{N} r_i^Q \times \hat{m}_i a_i \\ &= \sum_{i=1}^{N} -v_Q \times \hat{m}_i v_i + \sum_{i=1}^{N} r_i^Q \times \hat{m}_i a_i. \end{aligned} \tag{2.6}$$

Consider again (2.3) and before summing over i let's multiply the equations on the left by $r_i^Q \times$.

$$\sum_{i=1}^{N} r_i^Q \times F_i + \sum_{i=1}^{N} r_i^Q \times \sum_{j=1}^{N} f_{ij} = \sum_{i=1}^{N} r_i^Q \times \hat{m}_i a_i,$$

$$M_Q^{(ext)} + M_Q^{(int)} = \sum_{i=1}^{N} r_i^Q \times \hat{m}_i a_i. \tag{2.7}$$

Observe that the right hand side is related to the derivative of the angular momentum in Eq. (2.6). We can thus rewrite the previous equation as:

$$M_Q^{(ext)} = \frac{dL_Q}{dt} + v_Q \times m_G v_G. \tag{2.8}$$

If we choose $Q \equiv G$ then (2.8) simplifies into

$$M_G^{(ext)} = \frac{dL_G}{dt}.$$

2.2.2 DYNAMICS OF A RIGID BODY

So far we considered the dynamics of a system of particles. In order to model the dynamics of an articulated rigid body we consider now rigid systems, i.e. for every possible couple of particles forming the system their distance remains constant. Alternatively, the definition of rigid constraints can be also done by considering the velocity of the particles in the system. Consider the inertial frame $\mathcal{I}$ and a frame $\mathcal{B}$ with origin $O_{\mathcal{B}}$ rigidly connected to the system $\mathcal{S}$. Consider a generic particle P composing the rigid system. Its velocity w.r.t the inertial frame can be written as the following:

$$v^{\mathcal{I}} = v_{O_{\mathcal{B}}} + \omega \times (P - O_{\mathcal{B}}), \quad \forall\, P, O_{\mathcal{B}} \in \mathcal{S}$$

where $v_{O_{\mathcal{B}}}$ is the velocity of the origin $O_{\mathcal{B}}$ w.r.t. the inertial frame and $\omega \in \mathbb{R}^3$ is the angular velocity of the frame $\mathcal{B}$ w.r.t. the frame $\mathcal{I}$ and is defined such that

$$\dot{R}R^\top = S(\omega).$$

A rigid body system can be seen as composed of a continuum of particles satisfying the rigid constraints. The body has a volume V and has a mass density μ, such that:

$$m = \int_V \mu \, dV$$

where dV is the elementary volume element.

The center of mass can be defined as:

$$x_G = \frac{\int_V \mu x \, dV}{m}$$

where x is the coordinate vector of the point P associated with the infinitesimal volume element dV in the body. We can now introduce the inertial operator $I_Q : \mathbb{R}^3 \to \mathbb{R}^3$ as

$$I_Q(u) := \int_V \mu r_x^Q \times [u \times r_x^Q] \, dV, \quad \forall \, u \in \mathbb{R}^3$$

where $r_x^Q = (P - Q) \in \mathbb{R}^3$ is the distance between the point P associated with the infinitesimal volume element dV to the arbitrary point Q. By using the inertial operator we can define the angular momentum of a rigid body as:

$$L_G := m(x_G - Q) \times v_Q + I_Q(\omega)$$

which differentiated yields the following expression:

$$\frac{dL_Q}{dt} = m v_G \times v_Q + m(x_G - Q) \times a_Q + I_Q(\dot{\omega}) + \omega \times I_Q(\omega).$$

Finally, if we choose the point Q to coincide with the center of mass G we can rewrite (2.4) and (2.8) for a rigid body:

$$\begin{aligned} R^{(ext)} &= \frac{dP}{dt} = m a_G \\ M_G^{(ext)} &= \frac{dL_G}{dt} = I_G(\dot{\omega}) + \omega \times I_G(\omega). \end{aligned} \tag{2.9}$$

2.2.3 MOMENTA OF AN ARTICULATED RIGID BODY

Let's consider again a system of N particles grouped in n separate bodies of n_k points each, i.e. $\sum_{k=1}^{n} n_k = N$. As we did before let's consider (2.3) and let's sum over all the particles in a body for each body, i.e.

$$\begin{aligned} \sum_{k=1}^{n} \sum_{i=1}^{n_k} \left(F_i + \sum_{j=1}^{N} f_{ij} \right) &= \sum_{k=1}^{n} \sum_{i=1}^{n_k} (\hat{m}_i a_i), \\ \sum_{k=1}^{n} R^{(ext),k} &= \sum_{k=1}^{n} m_k a_{G_k} = \sum_{k=1}^{n} \frac{dL^k}{dt}, \\ \sum_{k=1}^{n} R^{(ext),k} &= \frac{d}{dt} \sum_{k=1}^{n} P^k \end{aligned} \tag{2.10}$$

where m_k is the mass of the body k, i.e. $\sum_{i=1}^{n_k} \hat{m}_i$.

Before performing the double summation we can multiply (2.3) on the left by $r_i^Q \times$ where Q is an arbitrary point but fixed for all bodies:

$$\sum_{k=1}^{n}\sum_{i=1}^{n_k} r_i^Q \times \left(F_i + \sum_{j=1}^{N} f_{ij}\right) = \sum_{k=1}^{n}\sum_{i=1}^{n_k} r_i^Q \times (\hat{m}_i a_i),$$
$$\sum_{k=1}^{n} M_Q^{(ext),k} = \sum_{k=1}^{n}\left(\frac{dL_Q^k}{dt} + v_Q \times m_k v_{G_k}\right). \tag{2.11}$$

If we choose $Q = G$, i.e. coinciding with the center of mass of the articulated rigid body, defined as:

$$x_G = \frac{\sum_{k=1}^{n_k} m_k x_{G_k}}{\sum_{k=1}^{n_k} m_k}$$

where x_{G_k} is the center of mass of the k-th body, we can simplify (2.11) to

$$\sum_{k=1}^{n} M_G^{(ext),k} = \sum_{k=1}^{n} \frac{dL_G^k}{dt} = \frac{d}{dt}\sum_{k=1}^{n} L_G^k. \tag{2.12}$$

$P := \sum_{k=1}^{n} P^k$ is the linear momentum and $L_G := \sum_{k=1}^{n} L_G^k$ is the angular momentum w.r.t. the center of mass of the articulated system.

As we did in Section 2.2.2 it is possible to perform the same analogy and consider a continuum of particles for each body.

2.3 DYNAMICAL MODEL OF FREE-FLOATING MECHANICAL SYSTEMS

This section introduces the dynamical model of mechanical systems with a particular focus on the free-floating ones.

We assume that the robot is composed of $n + 1$ rigid bodies – called links – connected by n joints with one degree of freedom each.

Definition 2.1. *Free-floating mechanical system. A mechanical system is called* free-floating *if none of its links has a constant pose, i.e. position and orientation, with respect to the inertial frame* $\mathcal{I}$.

On the contrary, if one of the links has a constant and known a-priori pose, the system is called *fixed base*. Note that the fact

that the link pose remains constant implies that there exist reaction forces that can counterbalance any external forces. Free-floating mechanical systems comprise humanoid robots. Space and submarine robots are free-floating mechanical systems too.

The configuration space of the multi-body system can then be characterized by the *position* and the *orientation* of a frame rigidly connected to the robot – called *base frame* $\mathcal{B}$ – and the joint configurations. More precisely, the robot configuration space is defined by

$$\mathbb{Q} = \mathbb{R}^3 \times SO(3) \times \mathbb{R}^n.$$

An element of the set $\mathbb{Q}$ is then a triplet $q = ({}^{\mathcal{I}}p_{\mathcal{B}}, {}^{\mathcal{I}}R_{\mathcal{B}}, q_j)$, where $({}^{\mathcal{I}}p_{\mathcal{B}}, {}^{\mathcal{I}}R_{\mathcal{B}})$ denotes the origin and orientation of the *base frame* expressed in the inertial frame, and q_j denotes the *joint angles*.

It is possible to define an operation associated with the set $\mathbb{Q}$ such that this set is a group. More precisely, given two elements γ and ρ of the configuration space, the set $\mathbb{Q}$ is a group under the following operation:

$$\gamma \cdot \rho = (p_\gamma + p_\rho, R_\gamma R_\rho, q_j + \rho_j). \tag{2.13}$$

Being the direct product of Lie groups, the set $\mathbb{Q}$ is itself a Lie group. The *velocity* of the multi-body system can then be characterized by the *algebra* $\mathbb{V}$ of $\mathbb{Q}$ defined by:

$$\mathbb{V} = \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^n.$$

An element of $\mathbb{V}$ is then a triplet $\nu = ({}^{\mathcal{I}}\dot{p}_{\mathcal{B}}, {}^{\mathcal{I}}\omega_{\mathcal{B}}, \dot{q}_j)$, where ${}^{\mathcal{I}}\omega_{\mathcal{B}}$ is the angular velocity of the base frame expressed w.r.t. the inertial frame, i.e. ${}^{\mathcal{I}}\dot{R}_{\mathcal{B}} = S({}^{\mathcal{I}}\omega_{\mathcal{B}})\,{}^{\mathcal{I}}R_{\mathcal{B}}$.

The choice of the group operation in (2.13) implies that an element $\nu \in \mathbb{V}$ is composed of $\dot{p}$, i.e. the time derivative of the origin of the floating base frame. Other choices for the group operation would imply a different algebra and, consequently, a different representation of the system *velocity*.

It is quite common in the robotics literature to parametrize the rotation group $SO(3)$ with a minimal set of coordinates in $\mathbb{R}^3$, e.g. Euler angles. Because every minimal parameterization of $SO(3)$ is singular at some configuration, thus holding only locally, when not stated differently we will always consider the group [104].

The Euler-Lagrange formulation assumes Euclidean space and thus cannot be applied directly to retrieve the dynamical model of a free-floating mechanical system. A more general extension, applied to elements belongings to arbitrary Lie groups are the Euler-Poincaré equations [59, Ch. 13.5].

Assuming that the robot is interacting with the environment through n_c distinct contacts, the application of the Euler-Poincaré formalism to the multi-body system yields the following equations of motion:

$$M(q)\dot{\nu} + C(q,\nu)\nu + g(q) = B\tau + \sum_{k=1}^{n_c} J_{\mathcal{C}_k}^{\top} f_k. \tag{2.14}$$

$M \in \mathbb{R}^{n+6\times n+6}$ is the mass matrix and it can be further decomposed as

$$M(q) = \begin{bmatrix} M_b(q) & M_{bj}(q) \\ M_{bj}^{\top}(q) & M_j(q) \end{bmatrix}$$

where $M_b(q) \in \mathbb{R}^{6\times 6}$ is the term relative to the base frame, $M_j(q) \in \mathbb{R}^{n\times n}$ is the term relative to the joints and $M_{bj} \in \mathbb{R}^{6\times n}$ is the coupling term. $C \in \mathbb{R}^{n+6\times n+6}$ is the Coriolis matrix describing the Coriolis and centrifugal forces in the equation of motion, $g \in \mathbb{R}^{n+6}$ is the gravity term, $B = (0_{n\times 6}, 1_n)^{\top}$ is a selector matrix and $\tau \in \mathbb{R}^n$ are the internal actuation torques. When the decomposition between Coriolis and gravitational effects are not needed, it is common to denote the whole vector of bias forces as

$$h(q,\nu) = C(q,\nu)\nu + g(q).$$

The mass matrix term relative to the base, M_b, has the following structure

$$M_b(q) = \begin{bmatrix} m1_3 & -mS({}^{\mathcal{B}}p_G - {}^{\mathcal{B}}p_{\mathcal{B}}) \\ mS({}^{\mathcal{B}}p_G - {}^{\mathcal{B}}p_{\mathcal{B}}) & I(q) \end{bmatrix}$$

where ${}^{\mathcal{B}}p_G$ is the position of the center of mass and ${}^{\mathcal{B}}p_{\mathcal{B}}$ is the position of the base frame, both expressed in the frame $\mathcal{B}$.

The number of actuation variables τ are, in general, at most equal to to number of joints the system possesses which are less than the total degrees of freedom. This is because (2.14) describes an underactuated mechanical system. In literature there exists multiple definitions of underactuation.

Definition 2.2. *Underactuated dynamical system. The following dynamical system*

$$\dot{\nu} = f_1(q,\nu,t) + f_2(q,\nu,t)u$$

is said to be underactuated *at time* t *in state* (q,ν) *if*

$$\operatorname{rank}(f_2(q,\nu,t)) < \dim(\nu).$$

Even if the definition is for a particular state and time, we will assume it to be a global property of the system. Note that in the literature underactuation is also defined w.r.t. the degrees of freedom necessary to accomplish a specified task, and not as a property of the system.

When we are in presence of an underactuated mechanical system it is often useful to divide the degrees of freedom in two groups. The *collocated* variables are the *actuated configuration variables*. The remaining variables are called *noncollocated*[101] [1].

The last term to be defined in (2.14) is the external total wrench applied by the environment on the link of the k-th contact denoted with $f_k \in \mathbb{R}^6$. We assume that the application point of the external wrench is associated with a frame $\mathcal{C}_k$, which is attached to the robot link where the wrench acts on, and has its z axis pointing as the normal of the contact plane. Then, the external wrench f_k is expressed in a frame whose orientation coincides with that of the inertial frame $\mathcal{I}$, but whose origin is the origin of $\mathcal{C}_k$, i.e. the application point of the external wrench f_k. The Jacobian $J_{\mathcal{C}_k} = J_{\mathcal{C}_k}(q)$ is the map between the robot's velocity ν and the linear and angular velocity ${}^{\mathcal{I}}\nu_{\mathcal{C}_k} := ({}^{\mathcal{I}}\dot{p}_{\mathcal{C}_k}, {}^{\mathcal{I}}\omega_{\mathcal{C}_k})$ of the frame $\mathcal{C}_k$, i.e.

$$ {}^{\mathcal{I}}\nu_{\mathcal{C}_k} = J_{\mathcal{C}_k}(q)\nu. \tag{2.15} $$

The Jacobian has the following structure:

$$ J_{\mathcal{C}_k}(q) = \begin{bmatrix} J^b_{\mathcal{C}_k}(q) & J^j_{\mathcal{C}_k}(q) \end{bmatrix} \in \mathbb{R}^{6\times n+6}, \tag{2.15a} $$

$$ J^b_{\mathcal{C}_k}(q) = \begin{bmatrix} 1_3 & -S({}^{\mathcal{I}}p_{\mathcal{C}_k} - {}^{\mathcal{I}}p_{\mathcal{B}}) \\ 0_{3\times 3} & 1_3 \end{bmatrix} \in \mathbb{R}^{6\times 6}. \tag{2.15b} $$

2.3.1 STATE SPACE FORM OF FREE-FLOATING MECHANICAL SYSTEMS

A free-floating mechanical system as described in Equation (2.14) can always be written state space form, i.e. in the form

$$ \dot{x} = f(x, u). $$

1 To the authors' knowledge a rigorous definition of collocated and noncollocated variables does not exist in literature. The collocated variables can be seen as the set of variables whose dynamics can be imposed at will when all the other degrees of freedom are kept fixed.

If we neglect the external forces for simplicity and we use a quaternion parametrization of $SO(3)$ we can define the following variables:

$$x := \begin{bmatrix} q \\ \nu \end{bmatrix}$$

where

$$q = \begin{bmatrix} {}^{\mathcal{I}}p_{\mathcal{B}} \\ \mathcal{Q} \\ q_j \end{bmatrix} =: \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \nu = \begin{bmatrix} {}^{\mathcal{I}}\dot{p}_{\mathcal{B}} \\ {}^{\mathcal{I}}\omega_{\mathcal{B}} \\ \dot{q}_j \end{bmatrix} =: \begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix}.$$

The unit quaternion $\mathcal{Q}$ is defined as

$$\mathcal{Q} = \begin{bmatrix} s \\ r \end{bmatrix}, \quad \|\mathcal{Q}\| = 1$$

where $s \in \mathbb{R}$ is the real part and $r \in \mathbb{R}^3$ is the imaginary part of the quaternion.

We can now compute the dynamics of x, obtaining

$$\dot{x} = \begin{bmatrix} {}^{\mathcal{I}}\dot{p}_{\mathcal{B}} \\ \frac{1}{2}\bar{S}({}^{\mathcal{B}}R_{\mathcal{I}}(q)\,{}^{\mathcal{I}}\omega_{\mathcal{B}})\mathcal{Q} \\ \dot{q}_j \\ M(q)^{-1}\left(\begin{bmatrix} 0 \\ \tau \end{bmatrix} - C(q,\nu)\nu - g(q)\right) \end{bmatrix} = f(x,u) \tag{2.16}$$

where we defined $\bar{S}(\omega)$ as the skew-symmetric matrix

$$\bar{S}(\omega) := \begin{bmatrix} 0 & -\omega^\top \\ \omega & -S(\omega) \end{bmatrix}.$$

The derivative of the quaternion [34, Sec. 3.2.2] in (2.16) should be corrected when the system is integrated numerically. Indeed it is possible that, when the integration step is not small enough, the quaternion loses the unitary property thus not representing a rotation anymore. By introducing a positive definite gain matrix $K_{\mathcal{Q}} \in \mathbb{R}^{4\times4}$, the derivative of the quaternion can be modified into

$$\frac{1}{2}\bar{S}({}^{\mathcal{B}}R_{\mathcal{I}}(q)\,{}^{\mathcal{I}}\omega_{\mathcal{B}})\mathcal{Q} + K_{\mathcal{Q}}(1 - \|\mathcal{Q}\|)\mathcal{Q}\,.$$

2.3.2 MODEL ASSUMPTIONS

The dynamical model in (2.14) possesses various properties which are usually exploited in the analysis or in design of controllers. We will now list the most used one.

Assumption 2.1. $M(q)$ *is symmetric and positive definite. This is equivalent to*

$$\alpha_1 1_{n+6} \leqslant M(q) \leqslant \alpha_2 1_{n+6} \quad \forall\, q$$

where $\alpha_1 > 0$ *and* $\alpha_2 > 0$ *are positive constants .*

Assumption 2.2. *The matrix* $\dot{M} - 2C$ *is skew symmetric, i.e.*

$$(\dot{M} - 2C) = -(\dot{M} - 2C)^\top.$$

Note that the choice of the matrix C *is not unique. Nevertheless it is always possible to choose it such that the property is satisfied. This property is also called* passivity *property of the mechanical system.*

Assumption 2.3. *Boundedness of the nonlinear terms.*

$$\begin{aligned} \left\| C(q,\nu) \right\| &\leqslant c_0 \|\nu\| \quad \forall\, q \\ \left\| g(q) \right\| &\leqslant g_0 \quad \forall\, q \end{aligned}$$

for some bounded constants $c_0 \in \mathbb{R}$ *and* $g_0 \in \mathbb{R}$.

2.4 RIGID CONTACTS

In general, a contact can be seen as a continuum of infinitesimal forces acting on the surface of a rigid body. We can summarize the effects of this continuum of forces by a single wrench f_k computed as the integral over the surface of the infinitesimal forces and moments.

Often we associate to a contact also a unilateral kinematic constraint. For example, a biped walking has a varying set of at least two unilateral constraints, i.e. the two feet touching and lifting from the ground.

This kind of constraints are usually unilateral, i.e. they can be modelled by inequality constraints. In the walking example, the force at the ground can only push, and not pull, the robot away from the ground. The constraints associated to rigid contacts possess an interesting property, that is the allowed motion directions are orthogonal to the resulting constraint forces.

Assuming the contact does not break, the constraint is usually modelled as a holonomic bilateral constraint that forbids any motion of the frame $\mathcal{C}_k$, i.e.

$$c_{\mathcal{C}_k}(q) = \text{constant} \tag{2.17}$$

where $c_{\mathcal{C}_k}(q) : \mathcal{Q} \to \mathbb{R}^3 \times SO(3)$ is the map between the configuration q and the pose (position and orientation) of the frame

$\mathcal{C}_k$. It is also possible to express the same constraint by using the velocity ν, i.e

$$J_{\mathcal{C}_k}(q)\nu = 0. \tag{2.18}$$

It is often useful to differentiate (2.18) in order to obtain the same order of the ordinary differential equation which models the system dynamics, i.e.

$$J_{\mathcal{C}_k}(q)\dot{\nu} + \dot{J}_{\mathcal{C}_k}(q)\nu = 0. \tag{2.19}$$

From a control and analysis standpoint bilateral constraints are more easily tractable than unilateral constraints. Particular care must be taken in ensuring that the constraints remain valid, i.e. the contacts do not break. This avoid to introduce in the control problem the complexity of modelling the system as an hybrid or switching system.

We thus define the stability of a contact as in the following:

Definition 2.3. *Contact stability. A contact is called* stable *if the associated motion constraint* (2.17) *remains valid over time.*

If the contact is kept "stable", we can safely substitute the unilateral constraints with the bilateral constraints previously introduced.

When multiple contacts are present, global stability criteria exist in literature. Nevertheless, in the present thesis we adopt criteria to ensure only local contact *stability*. Note that in Part III we consider only rigid contacts exerted on planar surfaces.

Given the definition of contact stability we provided, we must ensure that the resulting contact wrenches remains admissible, i.e. they do not lead to a breakage of the contact. Center of pressure and friction cone constraints are among the various criteria to be satisfy by the contact wrenches in order to ensure contact stability.

2.4.1 CENTER OF PRESSURE

Given a rigid body subject to contact forces we associate a field of pressure to the contact itself. Pressure is thus defined as the amount of normal force acting on a unit area element. The center of pressure (CoP) is defined as *the point on the contact surface where the resultant of the reaction forces acts*. Note that for the unilaterally of the constraint the normal forces are always positive. Thus the center of pressure can lie only inside the convex hull, i.e. the smallest convex set enclosing the contact surface.

Considering a rigid planar contact with the associated frame $\mathcal{C}_k$ defined with the z axis pointing as the normal of the contact

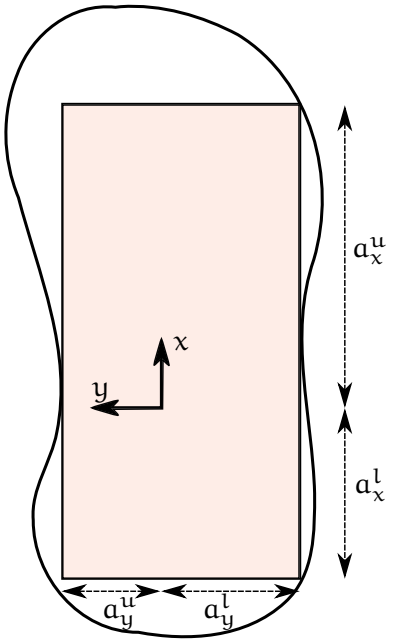


Figure 2.2: Example of a contact surface and the largest rectangle fitting inside the surface used as approximation.

surface, the contact wrench f_k is composed by the contact force F_k and moment μ_k, i.e.

$$f_k = \begin{bmatrix} F_k \\ \mu_k \end{bmatrix}$$

where the force and moment are defined by their components

$$F_k := \begin{bmatrix} F_{k_x} \\ F_{k_y} \\ F_{k_z} \end{bmatrix}, \quad \mu_k := \begin{bmatrix} \mu_{k_x} \\ \mu_{k_y} \\ \mu_{k_z} \end{bmatrix}.$$

We can then compute the center of pressure $\sigma_k \in \mathbb{R}^2$ associated at the k-th contact as

$$\begin{aligned} \sigma_{k_x} &= -\frac{\mu_{k_y}}{F_{k_z}} \\ \sigma_{k_y} &= \frac{\mu_{k_x}}{F_{k_z}}. \end{aligned}$$

2.4.1.1 *Inequality formulation*

We assume that the contact surface of the k-th contact is of rectangular shape, or we consider the largest rectangle fitting inside the contact surface. With respect the origin of the frame on the surface, we define $a_{k_x}^l$, $a_{k_x}^u$, $a_{k_y}^l$ and $a_{k_y}^u$ as positive constants

representing the dimension of the rectangle with respect to the origin of the k-th contact frame. In particular $a^l_{k_x}$ and $a^u_{k_x}$ are the distance of the boundary on the x axis and $a^l_{k_y}$ and $a^u_{k_y}$ are on the y axis. The superscript l or u denotes the lower bound and the upper bound respectively.

For each contact we decompose the constraints on the x and y axis. On the x axis the constraints to be imposed due to the k-th contact are

$$-a^l_{k_x} < -\frac{e_5^\top f_k}{e_3^\top f_k} < a^u_{k_x}$$

which become

$$\begin{cases} -(e_5 + a^u_{k_x} e_3)^\top f_k < 0 \\ (e_5 - a^l_{k_x} e_3)^\top f_k < 0. \end{cases}$$

On the y axis the constraints to be enforced are the following:

$$-a^l_{k_y} < \frac{e_4^\top f_k}{e_3^\top f_k} < a^u_{k_y}$$

which can be transformed into

$$\begin{cases} -(e_4 + a^l_{k_y} e_3)^\top f_k < 0 \\ (e_4 - a^u_{k_y} e_3)^\top f_k < 0. \end{cases}$$

To summarize the k-th contact gives rise to the following inequality constraint

$$\begin{bmatrix} 0 & 0 & -a^u_{k_x} & 0 & -1 & 0 \\ 0 & 0 & -a^l_{k_x} & 0 & 1 & 0 \\ 0 & 0 & -a^l_{k_y} & -1 & 0 & 0 \\ 0 & 0 & -a^u_{k_y} & 1 & 0 & 0 \end{bmatrix} f_k < \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Writing the previous equation as $A^{cop}_k f_k < 0_4$ we can write the inequality constraint due to the center of pressures of the n_c contacts as

$$\begin{bmatrix} A^{cop}_1 & 0_{4\times 6} & \cdots & 0_{4\times 6} \\ 0_{4\times 6} & A^{cop}_2 & \cdots & 0_{4\times 6} \\ \vdots & & & \\ 0_{4\times 6} & 0_{4\times 6} & \cdots & A^{cop}_{n_c} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n_c} \end{bmatrix} < 0_{4n_c}. \tag{2.20}$$

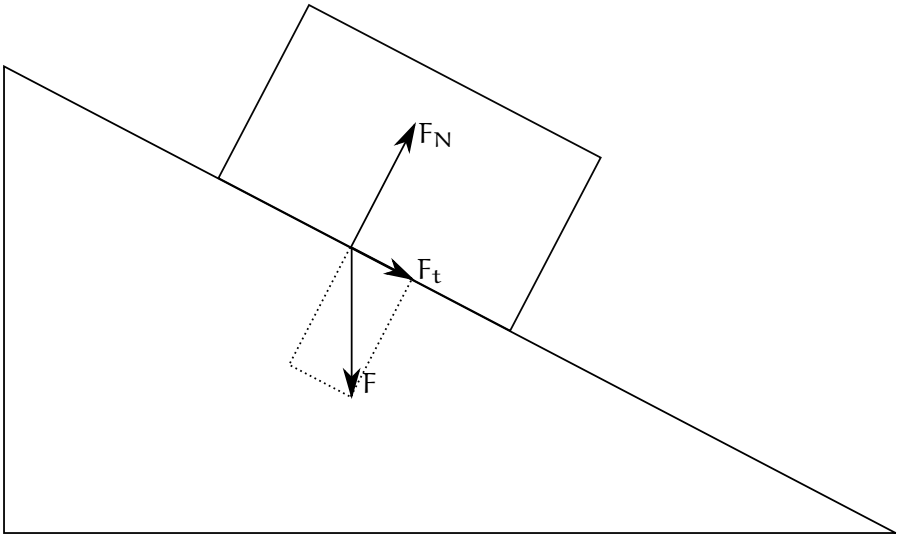


Figure 2.3: Normal and tangential forces acting on a block on an inclined plane. If the magnitude of the tangential force $\|F_t\|$ is less than the force due to static friction ($\mu_s\|F_N\|$) the block does not move.

2.4.2 FRICTION CONE

Friction is a complex and difficult phenomenon to model, yet it is essential in our daily life. Indeed we walk and we can grab objects because friction exists.

Different models of friction, depending on the level of details we want to achieve and of the type of materials considered, exist [73]. The Coulomb model of dry friction yields a simple, yet quite effective, model to represent the effect of friction between solid surfaces.

When two bodies are in contact and at rest, i.e. the relative velocity and acceleration is zero, the Coulomb model of static friction can be summarized with the following empirical law:

$$\|F_f\| \leqslant \mu_s\|F_N\| . \tag{2.21}$$

F_f is the force due to friction, is tangential to the surface and it acts against the resulting force, i.e. in the direction of the potential motion. μ_s is the static friction coefficient which depends on the materials of the two surfaces and F_N is the normal component of the contact force.

The Coulomb model in (2.21) thus states that in order to make a body accelerate w.r.t. the other, we have to produce a tangential force proportional to the normal force at the contact. Forces lower than this threshold are simply compensated for by the static friction.

Note that as soon as the inequality is no longer satisfied, i.e. the tangential force acting on the body is greater than the static

friction force, a different friction regime is established and (2.21) is no longer valid until the two bodies are at rest again. Instead the following law is used to model the friction effect:

$$\|F_f\| = \mu_d \|F_N\|$$

where μ_d is the coefficient of the kinetic dry friction.

When we control the interaction forces between the robot and a surface we are often interested in avoiding relative motion between the two bodies. We thus want to apply forces that satisfy (2.21).

If we consider the z axis coinciding with the normal direction, and the $x-y$ plane on the surface, equation (2.21) can be written as

$$\sqrt{F_x^2 + F_y^2} \leqslant \mu_s F_z,$$

which clearly defines a cone where the apex angle is function of the friction coefficient. We define the angle of friction θ as the maximum angle at which the two surfaces start sliding, i.e. when (2.21) holds with the equal sign. The friction angle is related to the static friction coefficient by the following relation:

$$\tan\theta = \mu_s.$$

The apex angle is twice the angle of friction.

The friction cone is often approximated with a pyramidal shape, which can be represented as linear constraints and thus can be used as constraints in a quadratic program. Figure 2.4 shows the friction cone and an approximation with a pyramid with square base.

2.4.2.1 *Inequality Formulation*

Instead of solving the associated Second-Order Cone Program (SOCP), usually the constraints are approximated into linear constraints.

We now show the approximation of the friction cone with a pyramid with square base for the sake of simplicity. It is possible however to choose a different, with more sides, base. We now describes the constraints resulting from the four-side-base approximation. The more points are used in the approximation the more precise it will be, but more constraints have to be added to the QP, possibly slowing down the solve procedure.

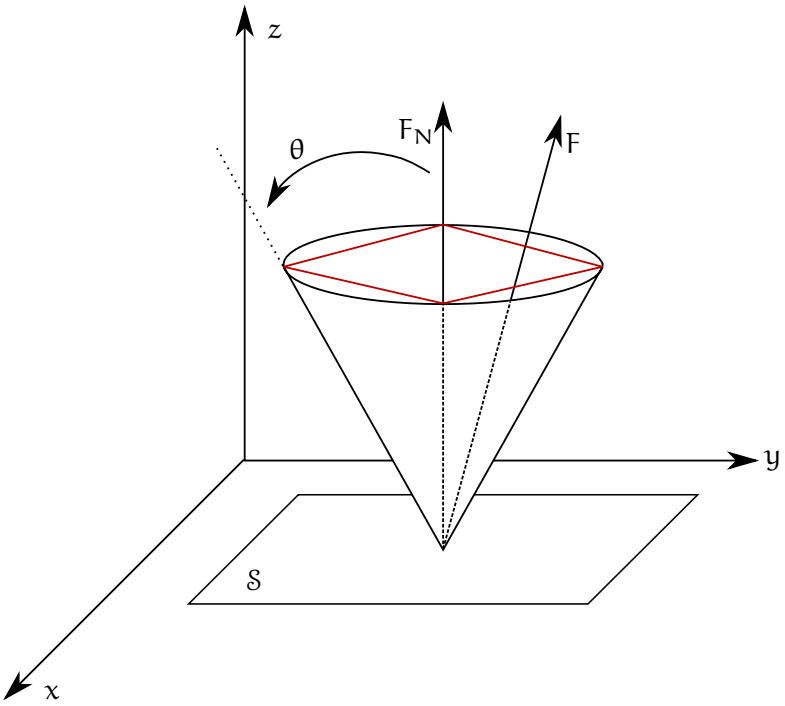


Figure 2.4: Friction cone constraints and in red an approximation using a pyramid with square base.

We approximate the friction cone with the chords passing between four equidistant points on the circle:

$$\begin{cases} F_{k_x} - F_{k_y} - \mu_s F_{k_z} \leqslant 0 \\ F_{k_x} + F_{k_y} - \mu_s F_{k_z} \leqslant 0 \\ -F_{k_x} + F_{k_y} - \mu_s F_{k_z} \leqslant 0 \\ -F_{k_x} - F_{k_y} - \mu_s F_{k_z} \leqslant 0 \end{cases}$$

which can be rewritten in matrix form as:

$$\begin{bmatrix} 1 & -1 & \mu_s & 0 & 0 & 0 \\ 1 & 1 & \mu_s & 0 & 0 & 0 \\ -1 & 1 & \mu_s & 0 & 0 & 0 \\ -1 & -1 & \mu_s & 0 & 0 & 0 \end{bmatrix} f_k \leqslant 0_4.$$

Denoting the previous inequality as $A_k^{f_c} f_k \leqslant 0_4$ we can write the friction cone constraints as

$$\begin{bmatrix} A_1^{f_c} & 0_{4\times 6} & \cdots & 0_{4\times 6} \\ 0_{4\times 6} & A_2^{f_c} & \cdots & 0_{4\times 6} \\ \vdots & & & \\ 0_{4\times 6} & 0_{4\times 6} & \cdots & A_{n_c}^{f_c} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n_c} \end{bmatrix} < 0_{4n_c}. \tag{2.22}$$

The last friction constraint to be added to the inequalities, models the friction along the z axis, i.e. along the normal direction. The constraint for the k-th contact has the simple form

$$\left\|\mu_{k_z}\right\| \leqslant \mu_s F_{k_z}.$$

The resulting matrix is

$$\begin{bmatrix} 0 & 0 & -\mu_s & 0 & 0 & 1 \\ 0 & 0 & -\mu_s & 0 & 0 & -1 \end{bmatrix} f_k \leqslant 0_2.$$

Denoting the previous inequality as $A_k^\tau f_k \leqslant 0_2$ we can rewrite the torsional friction inequality constraints as

$$\begin{bmatrix} A_1^\tau & 0_{2\times 6} & \cdots & 0_{2\times 6} \\ 0_{2\times 6} & A_2^\tau & \cdots & 0_{2\times 6} \\ \vdots & & & \\ 0_{2\times 6} & 0_{2\times 6} & \cdots & A_{n_c}^\tau \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n_c} \end{bmatrix} < 0_{2n_c}. \tag{2.23}$$

2.4.3 POSITIVITY OF THE NORMAL FORCES

The last constraint that is necessary to ensure the stability of the contact, is to ensure that the resulting contact force normal points outwards the contact surface, i.e. considering the definition of the frame $\mathcal{C}_k$, it must have a positive z component.

2.4.3.1 *Inequality formulation*

For the k-th contact force the constraint can be easily described by

$$e_3^\top f_k > 0 \Rightarrow -e_3^\top f_k < 0.$$

The constraints for all the forces can be formulated as

$$\begin{bmatrix} -e_3^\top & 0 & & \cdots & 0 \\ 0 & -e_3^\top & 0 & \cdots & 0 \\ \vdots & & & & \\ 0 & 0 & & \cdots & -e_3^\top \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n_c} \end{bmatrix} < 0_{n_c} \tag{2.24}$$

where the constraint matrix is $\in \mathbb{R}^{n_c \times 6n_c}$.

2.5 OPTIMAL CONTROL

Optimal control is a framework in which the control law is obtained so as to minimize an optimality criterion. A distinguishing feature of optimal control is the fact that the optimization

is performed by taking into account the time evolution of the dynamical system.

Optimal control has its roots in the calculus of variations. Its formalization is mainly due to the work of Lev Pontryagin and Richard Bellmann [49]. Optimal control has been used mainly in aerospace and chemical process industries. For example the famous Goddard's missile problem [27] can be formulated as an optimal control problem. Recently it has also been applied to robotics [50] and computer animation [105, 13, 68].

This section introduces the definition of an optimal control problem and the conditions of optimality of the solution. Depending on the kind and size of application a particular method can be better suited than others. For this reason this section briefly summarizes the main methodologies used to solve an optimal control problem, even if only a small subset has been used in the algorithm presented in Chapter 5.

2.5.1 WHAT AN OPTIMAL CONTROL PROBLEM IS

We first define a nonlinear dynamical system as

$$\dot{x} = f(x, u, t) \tag{2.25}$$

where $t \in [0, T]$ is the time, $x \in \mathbb{X} \subseteq \mathbb{R}^n$ is the state variable, $u \in \mathbb{U} \subseteq \mathbb{R}^m$ is the control variable and $f : \mathbb{X} \times \mathbb{U} \times \mathbb{R} \to \mathbb{X}$ is the dynamic function. To perform the analysis of the dynamical system, we do not lose generality if we do not explicitly consider the variable u, thus rewriting (2.25) into

$$\dot{x} = \tilde{f}(x, t).$$

If the dynamical system does not depend on time, i.e. $\dot{x} = \tilde{f}(x)$ then the system is said to be *autonomous*.

An optimal control problem can be defined as following:

$$\begin{aligned} \underset{x\in\mathbb{X},u\in\mathbb{U}}{\text{minimize}} \int_0^{t_f} \phi(x(\tau), u(\tau))\, d\tau + \psi(x(t_f)) \\ \text{subject to } 0 = \dot{x}(t) - f(x(t), u(t)), \quad t \in [0, t_f] \\ 0 = x(0) - x_0 \\ 0 \geqslant h(x(t), u(t)), \quad t \in [0, t_f]\ . \end{aligned} \tag{2.26}$$

The term inside the integral, ϕ, is also called *Lagrange* term while the final cost term, ψ is called *Mayer* term. The functional comprising both terms is in *Bolza* form. An Optimal Control problem can be reformulated in any of these forms.

Note that (2.26) is quite generic. Explicit dependence on the time or unknown parameters can be dealt with by state augmentation. The same approach can be used in case the final time t_f has to be optimized as well, e.g. minimum time problem.

The problem (2.26) describes an optimisation problem on a continuous time system. A part from specific cases where the solution can be obtained also in continuous time, the problem needs to be brought to a discrete formulation in order to be solved numerically on a computer.

2.5.2 DYNAMIC PROGRAMMING

Dynamic Programming is an approach to solve optimal control problems formulated by Richard Bellman in the 50s. It is based on the Principle of Optimality[2], which can be roughly defined as:

Definition 2.4. *Principle of Optimality. An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.*

In other words, the principle of optimality states that, if you want to be optimal, you can forget how you get to the current state and focus only on the remaining path to the goal.

Dynamic Programming is a very powerful tool. It is a convenient alternative to brute force algorithms and it is used in different fields. A comprehensive book regarding Dynamic Programming is [4]. We start by describing the Dynamic Programming method by first introducing its discrete formulation.

2.5.2.1 *Discrete time Dynamic Programming*

We consider the discrete time version of (2.25), i.e.

$$x_{i+1} = f(x_i, u_i), \quad i = 0, \dots N-1$$

where as before $x_i \in \mathbb{X}$ and, $u_i \in \mathbb{U}$. Note that for a practical implementation $\mathbb{X}$ and $\mathbb{U}$ must be finite.

2 also known as *Gianfrancioschio and the short fable*, which states the following. Gianfrancioschio was a person so unimportant that his story, even before beginning,... the end.

Problem (2.26), without considering additional constraints, can be written in discrete time as

$$\begin{aligned} \underset{x_0,u_0,x_1\ldots u_{N-1},x_N}{\text{minimize}} \quad & \sum_{i=0}^{N-1} \phi(x_i,u_i)+\psi(x_N) \\ \text{subject to } & 0 = x_{i+1} - f(x_i,u_i), \quad i=0,\ldots,N-1 \\ & 0 = x_0 - \bar{x}_0 \ . \end{aligned}$$

We then define the *value function* or optimal *cost-to-go* as

$$\begin{aligned} J_k(\bar{x}_k) = \min_{x_k,u_k,x_{k+1},\ldots,u_{N-1},x_N} & \sum_{i=k}^{N-1} \phi(x_i,u_i)+\psi(x_N) \\ \text{subject to } 0 = x_{i+1} - f(x_i,u_i), & \quad i=k,\ldots,N-1 \\ 0 = x_k - \bar{x}_k & \end{aligned}$$

with $k \geqslant 0$. The optimal total cost corresponds to the value function at time zero, i.e. $J_0(\bar{x}_0)$.

The power of the *principle of optimality* comes now into play. In fact, applying it to the definition of the cost-to-go we obtain that

$$J_k(\bar{x}_k) = \min_{u_k} \phi(\bar{x}_k,u_k) + J_{k+1}(f(\bar{x}_k,u_k)), \tag{2.27}$$

that is that at a given step k we can focus only on the current control variable because the paths with $j > k$ are already optimal by the application of the principle, and the iterations with $j < k$ does not influence the decision to be taken at step k.

Equation (2.27) is the *dynamic programming recurrent equation*. By setting

$$J_N(x_N) = \psi(x_N),$$

equation (2.27) is solved backward in time starting from $k = N-1$ to $k = 0$.

The optimal feedback control is thus the control which minimizes (2.27), i.e.

$$u_k^* = \underset{u_k}{\operatorname{argmin}}\, J_k(\bar{x}_k) = \underset{u_k}{\operatorname{argmin}}\, \phi(\bar{x}_k,u_k) + J_{k+1}(f(\bar{x}_k,u_k)).$$

2.5.2.2 *Infinite-space variables: the Curse of Dimensionality*

A critical point in the application of Dynamic Programming is the minimization of (2.27). In the discrete case, e.g. graph navigation, we can enumerate all possible values for the control variable. If, instead, the state and control variables are continuous variables, it is necessary to enact a discretization procedure, e.g. grid sampling. The number of points in the grid grows exponentially with the dimension of the continuous space. This is

what Bellmann called *Curse of Dimensionality*. Even with moderately low dimensional space the problem becomes easily computationally intractable.

2.5.2.3 *Special Case: Linear Quadratic Problem*

An exception to the curse of dimensionality exists, and, with some more hypothesis, leads to a very powerful and diffuse control tool: the Linear Quadratic Regulator. If the dynamical system is linear and the cost to be optimize is in quadratic form the dynamic programming recurrent equation can be analytically solved offline.

Let's consider the discrete linear system

$$x_{i+1} = A_i x_i + B_i u_i$$

where $A_i \in \mathbb{R}^{n\times n}$ and $B_i = \mathbb{R}^{n\times m}$. The cost assumes a quadratic form, i.e.

$$\phi(x_i, u_i) = \begin{bmatrix} x_i^\top & u_i^\top \end{bmatrix} \begin{bmatrix} Q_i & S_i^\top \\ S_i & R_i \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix}$$
$$\psi(x_N) = x_N^\top Q_N x_N.$$

Equation (2.27) can now be solved in this simplified case. Interestingly $J_k(x_k)$ is a quadratic form of x_k, i.e. $J_k(x_k) = x_k^\top P_k x_k$. We can now show the *Difference Riccati Equation*:

$$\begin{aligned} P_k = {} & Q_k + A_k^\top P_{k+1} A_k \\ & - (S_k^\top + A_k^\top P_{k+1} B_k)(R_k + B_k^\top P_{k+1} B_k)^{-1}(S_k + B_k^\top P_{k+1} A_k), \end{aligned} \tag{2.28}$$

requiring only $(R_k + B_k^\top P_{k+1} B_k)$ being positive definite. By initializing the recursion at $J_N(x_N) = \psi(x_N) = x_N^\top Q_N x_N$ (2.28) can be solved backward in time from $k = N-1$ to $k = 0$.

A very important point is that all the matrices P_k can be computed offline. The optimal control is defined by the following time-varying state-feedback action

$$u_k^*(x_k) = -(R_k + B_k^\top P_{k+1} B_k)^{-1}(S_k + B_k^\top P_{k+1} A_k) x_k$$

and can be applied online.

LINEAR QUADRATIC REGULATOR If the system is time invariant, the matrices Q_k, R_k, S_k are constant, and the problem to be solved is an infinite-horizon optimal control, i.e. $\psi = 0$ and $N \to \infty$, then the Difference Riccati Equation degenerates into the *Riccati Algebraic discrete time equation*:

$$P = Q + A^\top P A - (S^\top + A^\top P B)(R + B^\top P B)^{-1}(S + B^\top P A) \tag{2.29}$$

which must be solved for the symmetric positive definite matrix P.

The optimal control becomes then

$$u^*(x_k) = -(R + B^\top PB)^{-1}(S + B^\top PA)x_k = -Kx_k$$

which is a static state-feedback control.

2.5.2.4 *Continuous time Dynamic Programming: Hamilton-Jacobi-Bellman Equation*

Let's consider now how we can apply the dynamic programming principle to a continuous time problem. The performance measure we want to optimize is the following:

$$J(x(t), u(s), t) = \int_t^{t_f} \phi(x(\tau), u(\tau), \tau)\, d\tau + \psi(x(t_f), t_f) \tag{2.30}$$

for $t \leqslant s \leqslant t_f$.

Let's now divide the time interval to obtain:

$$\begin{aligned} J^*(x(t), t) &= \min_{\substack{u(s) \\ t \leqslant s \leqslant t_f}} \int_t^{t+\Delta t} \phi(\cdot)\, d\tau + \int_{t+\Delta t}^{t_f} \phi(\cdot)\, d\tau + \psi(x(t_f), t_f) \\ &= \min_{\substack{u(s) \\ t \leqslant s \leqslant t+\Delta t}} \int_t^{t+\Delta t} \phi(\cdot)\, d\tau + J^*(x(t+\Delta t), t+\Delta t) \end{aligned}$$

where we applied the *principle of optimality*.

If we assume differentiability of the optimal cost-to-go, we can expand it into its Taylor's series in $(x(t), t)$:

$$\begin{aligned} J^*(x(t), t) = \min_{\substack{u(s) \\ t \leqslant s \leqslant t+\Delta t}} & \int_t^{t+\Delta t} \phi(\cdot)\, d\tau + J^*(x(t), t) \\ & + \frac{\partial J^*}{\partial x}(x(t), t)[x(t+\Delta t) - x(t)] + \frac{\partial J^*}{\partial t}\Delta t \\ & + o(x(t+\Delta t) - x(t)) + o(\Delta t). \end{aligned}$$

Now, for small Δt,

$$\begin{aligned} J^*(x(t), t) = \min_{u(t)} & \phi(x(t), u(t), t)\Delta t + J^*(x(t), t) \\ & + \frac{\partial J^*}{\partial x}(x(t), t) f(x(t), u(t), t)\Delta t + \frac{\partial J^*}{\partial t}\Delta t + o(\Delta t). \end{aligned}$$

We can now reorder the terms depending on the minimization variable $u(t)$ and by performing $\lim_{\Delta t \to 0} \frac{J^*(x(t),t)}{\Delta t}$ we obtain:

$$-\frac{\partial J^*}{\partial t} = \min_{u(t)} \phi(x(t), u(t), t) + \frac{\partial J^*}{\partial x}(x(t), t) f(x(t), u(t), t) \tag{2.31}$$

which is a partial differential equation to be solved backward in time and initialized at $t = t_f$ with

$$J^*(x(t_f), t_f) = \psi(x(t_f), t_f).$$

If we define the *Hamiltonian* as

$$\mathcal{H}(x(t), u(t), \lambda, t) := \phi(x(t), u(t), t) + \lambda^\top f(x(t), u(t), t) \quad (2.32)$$

and

$$\mathcal{H}^*(x(t), \lambda, t) = \mathcal{H}(x(t), u^*(t), \lambda, t) = \min_{u(t)} \mathcal{H}(x(t), u(t), \lambda, t)$$

we can rewrite Equation (2.31) in term of its hamiltonian, to obtain the Hamilton-Jacobi-Bellman equation:

$$-\frac{\partial J^*}{\partial t} = \mathcal{H}^*(x(t), u^*(t), \frac{\partial J^{*\top}}{\partial x}, t). \quad (2.33)$$

LINEAR QUADRATIC PROBLEM We can apply the Hamilton-Jacobi-Bellman (HJB) equation to a linear time varying systems as we applied the Dynamic Programming equation to the discrete time system.

The problem we are trying to solve is the following optimal control problem:

$$\begin{aligned} \underset{x(t),u(t)}{\text{minimize}} & \int_{t=0}^{t_f} \begin{bmatrix} x(\tau)^\top & u(\tau)^\top \end{bmatrix} \begin{bmatrix} Q(\tau) & S(\tau)^\top \\ S(\tau) & R(\tau) \end{bmatrix} \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} d\tau \\ & + x(T)^\top Q_N x(T) \\ \text{subject to } & 0 = \dot{x} - A(t)x(t) - B(t)u(t), \quad t \in [0, t_f] \\ & 0 = x(0) - x_0. \end{aligned}$$

By applying the HJB equation we finally obtain to the *Differential Riccati Equation*:

$$-\dot{P} = Q + PA + A^\top P - (S^\top + PB)R^{-1}(S + B^\top P) \quad (2.34)$$

initialized with $P(N) = Q_N$ and which must be integrated backwards in time. The optimal control is now

$$u(x(t), t) = -R(t)^{-1}(S(t) + B^\top(t)P(t))x(t).$$

CONTINUOUS TIME LINEAR QUADRATIC REGULATOR As we did for the discrete time system, let's consider the infinite horizon optimal control problem, with linear time-invariant dynamical system and constant cost matrices:

$$\begin{aligned}\underset{x(t),u(t)}{\text{minimize}} & \int_{t=0}^{\infty} \begin{bmatrix} x(\tau)^\top & u(\tau)^\top \end{bmatrix} \begin{bmatrix} Q & S^\top \\ S & R \end{bmatrix} \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} d\tau \\ \text{subject to } & 0 = \dot{x} - Ax(t) - Bu(t), \quad t \in [0, \infty] \\ & 0 = x(0) - x_0.\end{aligned}$$

By imposing the stationary of the solution of (2.34), i.e. $\dot{P} = 0$ we obtain the *Algebraic Riccati equation in continuous time*:

$$Q + PA + A^\top P - (S^\top + PB)R^{-1}(S + B^\top P) = 0 \tag{2.35}$$

and the static state-feedback control

$$u(x(t)) = -R^{-1}(S + B^\top P)x(t) = -Kx(t).$$

The importance of the LQ Regulator for its practical use lies in the following theorem:

Theorem 2.1. *Assume the following facts:*

- *The pair* (A, B) *is stabilizable*
- *The pair* (A, D)*, with* $Q = D^\top D$ *is detectable*
- $R = R^\top$ *is positive definite*
- $Q = Q^\top$ *is positive semi-definite*

then the following results hold true:

i) *The total cost* $J^*(x_0, 0)$ *is minimized*

ii) $\bar{P} = \bar{P}^\top$ *is the unique positive-definite solution to* (2.35)

iii) *The equilibrium point* $x = 0$ *of the closed loop dynamics* $(A - BK)$ *is asymptotically stable, i.e. all the eigenvalues have strictly negative real part.*

The proof can be found in [3, Ch. 3.2].

2.5.3 PONTRYAGIN MAXIMUM PRINCIPLE AND INDIRECT METHODS

The *Pontryagin Maximum Principle*, or *minimum principle*, are necessary optimality conditions for the optimal control in continuos time. By using these conditions, it is possible to eliminate the control variable from the original problem and then solve the resulting Boundary Value Problem (BVP). This is the indirect method to solve an optimal control problem.

The minimum principle is based on the *calculus of variations*, and adds the possibility to include path dependent inequality constraints.

Necessary conditions for the trajectory $x^*(t)$ and the control $u^*(t)$ to be optimal is that:

$$\begin{aligned}
\dot{x}^*(t) &= \nabla_\lambda \mathcal{H}(x^*(t), \lambda^*(t), u^*(t)) \\
\dot{\lambda}^*(t) &= -\nabla_x \mathcal{H}(x^*(t), \lambda^*(t), u^*(t)) \\
u^*(t) &= \underset{u}{\operatorname{argmin}}\, \mathcal{H}(x^*(t), \lambda^*(t), u(t)) \\
x^*(0) &= x_0 \\
\lambda^*(t_f) &= \nabla_x \psi(x^*(t_f)).
\end{aligned} \tag{2.36}$$

The above equations are a boundary value problem in the state (x) and costate (λ) variables. Initial condition is provided for the state, i.e. $x(0) = x_0$ while terminal condition is provided for the costate, i.e. $\lambda(t_f) = \phi(t_f)$.

In order to solve the BVP in the variables $(x(t), \lambda(t))$, the optimal control $u^*(x(t), \lambda(t))$ must be derived and eliminated from the problem. We thus impose the first order optimality conditions, and we solve

$$\frac{\partial \mathcal{H}}{\partial u}(x(t), \lambda(t), u^*(t)) = 0 \tag{2.37}$$

in the variable $u^*(t)$. If the relation (2.37) is not invertible w.r.t. $u(t)$ we have a *singular arc*. A solution is to totally differentiate (2.37) with respect to time, until the control variable can be explicitly obtained, i.e.

$$\left(\frac{d}{dt}\right)^{(n)} \frac{\partial \mathcal{H}}{\partial u}(x(t), \lambda(t), u^*(t)) = 0.$$

Path constraints in the form of $h(x(t), u(t)) \leqslant 0$ can be incorporated as well in the formulation with different complexity depending if they are simple control constraints, state constraints or mixed constraints.

The minimum principle presents some disadvantages which limit its practical use. In particular, it is not always straightforward, or sometimes even impossible, to eliminate the control variables from the problem. Furthermore, the differential equation in the resulting BVP can become very nonlinear and unstable, thus rendering very complex its numerical resolution.

2.5.4 DIRECT METHODS

The direct approach consists today in the state-of-the-art method to solve an optimal control problem, a part from the LQR solution when applicable.

The basic principle behind the direct approach is that the optimal control problem (2.26) is discretized and transformed into a large, finite dimensional nonlinear program (NLP). At this point an NLP solver can be used to efficiently solve this problem [71, 6].

The three different direct methods differ in how the continuous time problem is "transcribed" into an NLP.

We refer the interested reader to [5] for a more detailed description of the methods. The following works also treats about solving optimal control problems through direct methods [109, 48, 56].

2.5.4.1 *Single shooting*

Single shooting method is a *sequential* approach, i.e. the system is first simulated and then optimized.

The first step for a single shooting method is to finitely parametrize the control variable $u(t)$. A common approach is by piecewise constant function, i.e.

$$u(t) = \bar{u}_i, \quad t \in [t_i, t_{i+1}]$$

with $0 = t_0 < t_1 < \cdots < t_N = T$ and $\bar{u} \in \mathbb{R}^{Nm}$, but more rich parametrizations are possible.

At this point the state trajectory $x(t)$ is obtained by integration, with an ODE integrator, starting from the initial condition $x(0) = x_0$ and by the application of the discretized control.

The obtained NLP in the only variable $\bar{u} \in \mathbb{R}^{Nm}$ can be solved by a dense solver. The original sparsity of the optimal control is lost during the transcription.

2.5.4.2 *Multiple shooting*

In the multiple shooting approach the control is discretized on a grid, like we did in the single shooting approach. Differently than before the state trajectory is not computed on the whole timeframe, but only in the single time interval starting with artificial initial conditions. Also the integral term in the objective function is evaluated between t_i and t_{i+1}. (2.26) thus become

$$\begin{aligned}
\underset{\bar{x},\bar{u}}{\text{minimize}} \quad & \sum_{i=0}^{N-1} \bar{\phi}(\bar{x}_i, \bar{u}_i) + \psi(\bar{x}_N) \\
\text{subject to} \quad & \bar{x}_{i+1} - f_i(\bar{x}_i, \bar{u}_i) = 0, \quad i = 0, \dots, N-1 \\
& 0 \geqslant h_i(x(t), u(t)), \quad i = 0, \dots, N \\
& 0 = \bar{x}_0 - x_0
\end{aligned}$$

where $\bar{\phi}(\bar{x}_i, \bar{u}_i) = \int_{t_i}^{t_{i+1}} \psi(\bar{x}_i, \bar{u}_i)\, d\tau$ and $\bar{x}_i$ corresponds to the artificial initial conditions.

This obtained system should be solved by a sparse NLP solver. Indeed, in the single shooting method the number of variables was Nm. Now the problem is larger, having $N(n+m)$ variables, but the structure is preserved and should be exploiting for an efficient solution.

2.5.4.3 *Direct collocation*

A third class of direct methods commonly used is *direct collocation*.

We start by parameterizing both control and states on a fine grid, $t_i, i = 0, \ldots, N$, corresponding to an integrator step. As before we denote $\bar{x}_i$ and $\bar{u}_i$ the state and the control value on the grid.

In each interval, l collocation points are chosen, $t_i^{(1)}, \ldots t_i^{(l)}$ and the trajectory is approximated by a l-order polynomial $p_i(t, w_i)$ with coefficient vector w_i, i.e.

$$p(t, w) := \sum_{j=0}^{l} w_j t^j.$$

We now impose as equalities constraints that the collocation conditions are met at the collocation points, i.e. the derivative of the polynomial

$$p'(t, w) := \sum_{j=0}^{l} j w_j t^{j-1}$$

must match the value of the dynamic function at the corresponding points. These conditions are thus:

$$\begin{aligned} \bar{x}_i &= p_i(t_i, w_i) \\ f(p_i(t_i^{(1)}, w_i), \bar{u}_i(t_i^{(1)})) &= p_i'(t_i^{(1)}, w_i) \\ &\vdots \\ f(p_i(t_i^{(l)}, w_i), \bar{u}_i(t_i^{(l)})) &= p_i'(t_i^{(l)}, w_i). \end{aligned}$$

Additionally, continuity must be preserve across interval boundaries, i.e. $p_i(t_{i+1}, w_i) = \bar{x}_{i+1}$ and the integral must be approximated by a quadrature formula on all the collocation points. Path constraints can be enforced on the grid.

The resulting NLP is very large, but at the same time sparse. It should be solved by a sparse NLP solver.

2.5.5 CLOSING THE LOOP IN THE OPTIMAL CONTROL FRAMEWORK

As we showed in the previous sections, the only method that leads to feedback control laws is the one based on the Hamilton-Jacobi-Bellman equation or on the Dynamic Programming principle in the discrete case. Unfortunately all the other methods yield open-loop sequences of control actions to be applied to the dynamical system. Because a control system must be robust to disturbances and perturbations, it must close the loop through a feedback action.

Interestingly, all the numerical methods described compute the solution starting from an initial state x_0. This fact is used in the Receding Horizon Control, or Model Predictive Control (MPC) approach [62]. In this method, the control action is obtained by solving online, at each time step, a finite-horizon optimal control problem by setting the initial state x_0 as the current measured plant state. Once a solution is obtained, only the first control of the sequence is applied to the system. At the next time instant a new solution is computed again, and the process is reiterated.

Algorithms for Model Predictive Control usually focus on obtaining an (also approximate) solution in the least time possible. Some algorithms also try to use the information coming from the initial condition x_0 as late as possible [18].

Stability of a feasible equilibrium point through the use of Model Predictive Control has been proved [63, 44, 9, 16]. Stability-ensuring constraints or criteria have to be added to the optimization problem. For example forcing the system to be at the final time in the equilibrium point or in a set comprising the equilibrium point are commonly used constraints.

3 EXPERIMENTAL PLATFORM: THE ICUB HUMANOID ROBOT

The test platform used to validate the control algorithms presented in this thesis is the state-of-the-art humanoid robot iCub. iCub is an open source robotic platform for research in cognitive robotics and it is extensively described in [65]. This chapter describes the relevant component divided into hardware and software.

3.1 HARDWARE

The iCub humanoid robot is 104 cm tall. The weight varies from version to version. The iCub version used in this thesis weights 33 kg.

iCub possesses in total 53 degrees of freedom. Most of these, are located in the hands and in the eyes. The degrees of freedom considered in our application are 23 and they are distributed in the following way:

i) 6 DoFs for each legs

ii) 3 DoFs for the torso

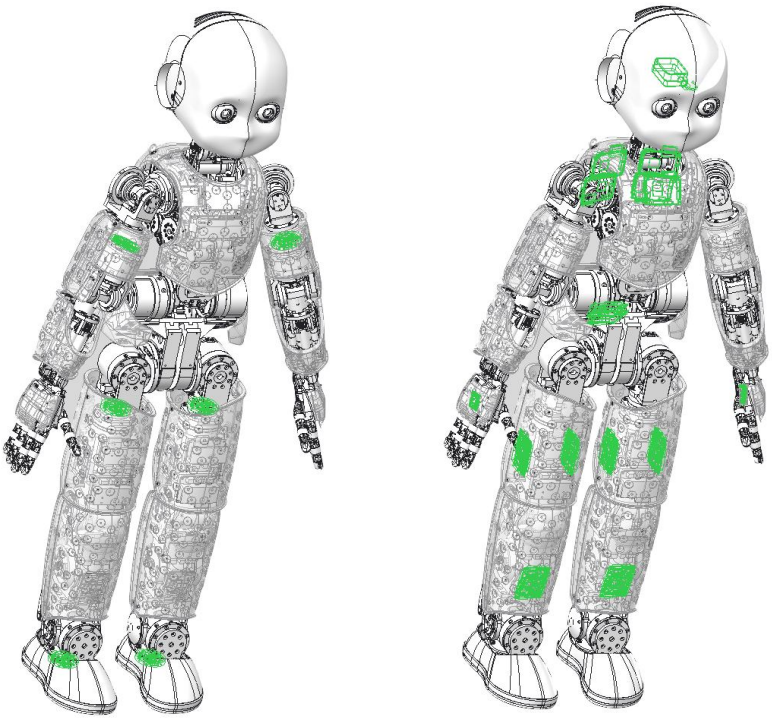
iii) 4 DoFs for the upper arm, 3 of which in the shoulder and one in the elbow.

Notably the torso and the shoulder joints are mechanically coupled and are driven by tendon mechanisms. All the 23 joints are controlled by brushless electric motors coupled with harmonic drive gears.

3.1.1 SENSORS

A peculiar feature of iCub is the vast array of sensors available, included force/torques sensors, accelerometers, gyroscopes, a distributed tactile skin, two cameras and microphones.

More in detail, iCub possesses 6 six-axes force/torque sensors [24]. Two of them are mounted at the shoulder, the other four respectively at the hip and at the ankle. Figure 3.1a shows the location of the sensors in the mechanical structure. iCub



(a) Locations of the 6 six-axis force/torque sensors

(b) Locations of the motor micro-controllers and the commercial inertial sensor

Figure 3.1: Locations of the sensors and controllers in the mechanical structure of iCub

also possesses distributed tactile sensors, the *artificial skin* [8, 58], which provides information about both the location and the intensity of the contact forces. Notably, the skin is distributed on the torso, arms, the palm and fingertips of the hands and legs. Because iCub is not endowed with joint torque sensors, the F/T sensors and the skin are used to provide an estimation of both the internal torques and external forces [24].

Finally, an inertial sensor providing acceleration and angular velocity information is mounted on the head, as Figure 3.1b depicts.

3.2 SOFTWARE ARCHITECTURE

3.2.1 YARP

Yet Another Robot Platform (YARP) [64] is a software middleware which main purpose is to allow seamless communication between different "applications", which can reside on different computers. Furthermore, it provides interfaces to interact with

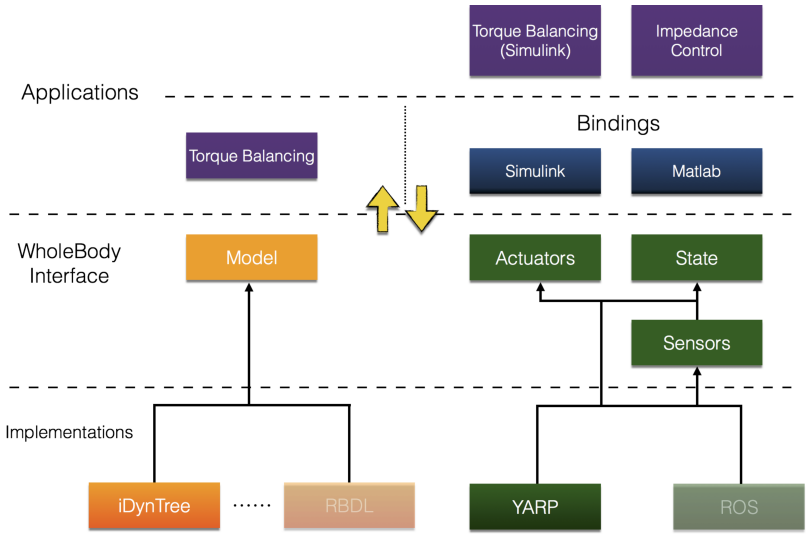


Figure 3.2: Architecture of the Whole-Body Interface, its implementations and some of the applications currently exploiting the middleware.

physical devices independently of the actual implementation, thus facilitating code reuse and modularity.

iCub functionalities are based on YARP, and the different sensors and motor controller are exposed through the YARP interfaces.

3.2.2 TORQUE CONTROL

The 23 degrees of freedom used for the whole-body control are supplied of a low-level joint torque control implemented in firmware and running at 1 kHz.

The torque control is in charge of stabilizing any desired torque reference. Viscous and coulomb friction is also compensated by the control loop. Motor parameters, namely the two friction coefficients and the motor torque constant, are identified beforehand for each joint.

The control law implemented in the firmware to provide torque control to the robot is

$$V = PI(\tau - \tau_d) + \frac{k_c}{k_\tau} sign(\dot{\vartheta}) + \frac{k_v}{k_\tau}(\dot{\vartheta})$$

where PI is a proportional and integral action applied on the torque error, ϑ is the motor variable and k_c, k_v and k_τ are the Coulomb and viscous friction coefficients and the motor torque constant identified beforehand.

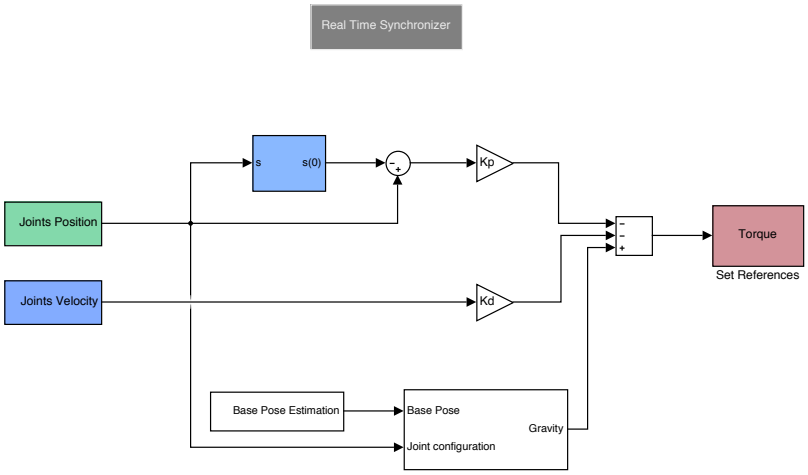


Figure 3.3: Example of impedance control exploiting the Simulink bindings of the Whole-Body Interface

3.2.3 WHOLE-BODY INTERFACE

On top of the YARP middleware, we added an additional software layer, called Whole-Body Interface.

The objective of this set of protocols is to simplify the writing of Whole-body controllers. It wraps dynamic computations such as the computation of the mass matrix or inverse dynamics, together with the possibility to interface with the robot, acquiring the state and commanding the actuators. Figure 3.2 shows the architecture of the interface with the functionalities grouped in four big areas, i.e. model, actuators, state and sensors. The applications are independent on the particular implementation, e.g. the one exploiting YARP for the communication and *iDynTree* for the kinematic and dynamic computations.

The interface has been coded in C++ and it can be interfaced directly by C++ applications or through the exposed bindings in Simulink® or MATLAB®. The ease of developing a control system with Simulink, together with the possibility to exploit simple debugging tools and existing toolboxes, render the use of the Simulink bindings the first choice to rapid prototype controllers. As an example, Figure 3.3 shows the implementation of a PD plus gravity compensation controller with the Simulink bindings of the Whole-Body Interface.

Part II

NONLINEAR CONTROL STRATEGIES FOR UNDERACTUATED FREE-FLOATING MECHANICAL SYSTEMS

4 ADAPTIVE CONTROL OF UNDERACTUATED MECHANICAL SYSTEMS

This chapter presents an extension to the classic Slotine's adaptive controller [99] to the underactuated case where we assume that the the control objective is the asymptotic stabilization of the collocated state space. Inspired by the back-stepping method, the key point is to view the sliding variable as the difference between the system velocity and an exogenous state, whose dynamics is considered as control input.

The new definition of the sliding variable allows us to extend directly the controller [99] to the underactuated case when the objective is the stabilization of the collocated dynamics.

Note that addressing the classical and well known drawbacks of adaptive control schemes (see, e.g., [2] and the references therein) are beyond the scope of the work hereby presented.

Before describing the main contribution, we introduce an additional property of the dynamical model of mechanical system presented in 2.3. We also rewrite the classical Slotine's adaptive control scheme in the new formulation which allows an easier extension to the underactuated case. We then present the simulation and the experimental results performed on an acrobot model.

4.1 PRELIMINARIES

The following results apply to a generic underactuated mechanical system of which the system described in (2.14) can be seen as a special case. In particular the position and orientation of the base of (2.14) are the *noncollocated* variables of the underactuated system. In this chapter we assume, differently from Section 2.3, that the configuration space $\mathbb{Q}$ is in $\mathbb{R}^n$. Of course the definition of the *collocated* and *noncollocated* variables depends on the particular system considered. To summarize, the configuration variable $q \in \mathbb{R}^n$ and

$$\begin{aligned} \nu &\equiv \dot{q} \in \mathbb{R}^n, \\ \dot{\nu} &\equiv \ddot{q} \in \mathbb{R}^n. \end{aligned}$$

An additional property of (2.14) is required and it is introduced in the following.

Property 4.1. *Linearity w.r.t. system base parameters. The dynamical model* (2.14) *can be expressed linearly with respect to the system base parameters* π*. Also, there exists a* regressor matrix $Y(\cdot) \in \mathbb{R}^{n\times p}$ *such that*

$$M(q,\pi)\ddot{q} + C(q,\dot{q},\pi)\xi + g(q,\pi) = Y(q,\dot{q},\xi,\ddot{q})\pi,$$

for any vector $\xi \in \mathbb{R}^n$.

The matrix $Y(\cdot)$ is the so-called Slotine-Li regressor. Observe that in view of the algebraic Property 4.1, the dynamics (2.14) can be compactly written by substituting ξ with $\dot{q}$, i.e.

$$Y(q,\dot{q},\dot{q},\ddot{q})\pi = \tau.$$

4.1.1 THE SLOTINE'S ADAPTIVE CONTROL - REVISITED

Before presenting our contribution, we want to describe the original work by Slotine on the adaptive control on fully actuated mechanical systems [99].

Let $r(t) \in \mathbb{R}^n$ denote a time-varying reference trajectory for the variables q. We also make the following assumption:

Assumption 4.1. *The reference trajectory* $r(t)$ *is bounded in norm on* $\mathbb{R}^+$*, and its first and second order time derivatives are well-defined and bounded on this set.*

We present below a revisited version of the scheme [99] that ensures the asymptotic stabilization of the tracking error

$$e := q - r \tag{4.1}$$

to zero without the knowledge of the system parameters π. The benefits of the following slightly different formulation will be clear in the next section where we present our contribution. First, define:

$$\tilde{\xi} := \xi - \dot{r}, \tag{4.2a}$$
$$s := \dot{q} - \xi, \tag{4.2b}$$
$$\tilde{\pi} := \hat{\pi} - \pi, \tag{4.2c}$$

where $\tilde{\pi}$ is the base parameters estimation error. By considering $\dot{\hat{\pi}}$ and $\dot{\xi}$ as auxiliary control inputs, fusing and reformulating the results [99] [102] lead to the following lemma.

Lemma 4.1. *Assume that Properties 2.1-2.3, 4.1 and Assumption 4.1 hold. Apply the following control laws to System (2.14)*

$$\tau = Y(q, \dot{q}, \xi, \dot{\xi})\hat{\pi} - Ks, \tag{4.3a}$$
$$\dot{\xi} = \ddot{r} - \Lambda_1 \dot{e} - \Lambda_2 e, \tag{4.3b}$$
$$\dot{\hat{\pi}} = -\Gamma Y^\top(q, \dot{q}, \xi, \dot{\xi})s, \tag{4.3c}$$

with $K, \Lambda_1, \Lambda_2 \in \mathbb{R}^{n\times n}$ diagonal, constant positive definite matrices, and $\Gamma \in \mathbb{R}^{p\times p}$ a constant positive definite matrix. Then, the following results hold.

i) The equilibrium point $\left(e, \tilde{\xi}, s, \tilde{\pi}\right) = (0_n, 0_n, 0_n, 0_p)$ of the closed loop dynamics $\left(\dot{e}, \dot{\tilde{\xi}}, \dot{s}, \dot{\tilde{\pi}}\right)$ is stable;

ii) For any initial condition $\left(e, \tilde{\xi}, s, \tilde{\pi}\right)(0)$, the trajectories of the closed loop dynamics are bounded, and the tracking error $e(t)$ converges to zero.

The proof is given in Appendix A.1. The main difference between the above formulation and that of [99] resides in the definition (4.2b), and in the fact that $\dot{\xi}$ is viewed as an auxiliary control input. The above Lemma shows that this standpoint does not affect stability, in terms of Lyapunov, and convergence.

Observe also that boundedness and convergence are independent from the initial condition $\tilde{\xi}(0)$ thanks to the additional term $\Lambda_2 e$ in (4.3b). This term plays the role of an integral action in the expression of (4.2b), and compensates for the initial condition $\tilde{\xi}(0)$.

For Lemma 4.1 to hold, it is assumed that System (2.14) is fully actuated. We now present an extension of Lemma 4.1 to the case where System (2.14) is underactuated.

4.2 COLLOCATED ADAPTIVE CONTROL

Assume that System (2.14) possesses only $m < n$ torque control inputs so that the first $k := n - m$ rows on the right hand side of Eq. (2.14) are identically equal to zero, i.e.

$$Y(q, \dot{q}, \dot{q}, \ddot{q})\pi = \begin{pmatrix} 0_k \\ \bar{\tau} \end{pmatrix}, \tag{4.4}$$

with $\bar{\tau} \in \mathbb{R}^m$ the control inputs. Now, partition the generalized coordinate vector q as follows

$$q := \begin{pmatrix} q_n \\ q_c \end{pmatrix}, \tag{4.5}$$

where $q_n \in \mathbb{R}^k$, $q_c \in \mathbb{R}^m$, and the suffixes "n" and "c" stand for *noncollocated* and *collocated*, respectively. Assume that the control objective is the asymptotic stabilization of the collocated coordinates q_c about reference trajectories $r(t) \in \mathbb{R}^m$, i.e. the stabilization of the tracking error

$$e := q_c - r \tag{4.6}$$

to zero. As before, we want to design control laws for this control objective without the knowledge of the parameters π.

Let us show the difficulties in attempting to apply Lemma 4.1 for controlling the collocated state space q_c. This Lemma assumes that Property 4.1 holds, i.e. the inverse dynamics of the controlled variables is linear with respect to the parameters π. In view of the system dynamics (4.4), this linearity still holds for the collocated state space q_c. Then, the stabilization of the tracking error e to zero may be achieved by applying the laws (4.3) as follows

$$\bar{\tau} = Y_c\left(q, \dot{q}, \begin{pmatrix}\dot{q}_n\\ \xi\end{pmatrix}, \begin{pmatrix}\ddot{q}_n\\ \dot{\xi}\end{pmatrix}\right)\hat{\pi} - Ks_c, \tag{4.7a}$$

$$\dot{\hat{\pi}} = -\Gamma Y_c^\top\left(q, \dot{q}, \begin{pmatrix}\dot{q}_n\\ \xi\end{pmatrix}, \begin{pmatrix}\ddot{q}_n\\ \dot{\xi}\end{pmatrix}\right)s_c, \tag{4.7b}$$

where

$$Y(q, \dot{q}, \xi, \ddot{q}) = \begin{pmatrix} Y_n(q, \dot{q}, \xi, \ddot{q})\\ Y_c(q, \dot{q}, \xi, \ddot{q})\end{pmatrix}, \quad s := \begin{pmatrix} s_n\\ s_c\end{pmatrix}, \tag{4.8}$$

with $s_n \in \mathbb{R}^k$, $s_c \in \mathbb{R}^m$, $Y_n \in \mathbb{R}^{k\times p}$, $Y_c \in \mathbb{R}^{m\times p}$, and $\dot{\xi} \in \mathbb{R}^m$ still governed by (4.3b). Note that $Y_c(\cdot)$ in (4.7) depends upon the accelerations of the noncollocated state space $\ddot{q}_n$. Therefore, if the measurement of this acceleration were available, one might apply the laws (4.3) for the adaptive control of e to zero.

The above approach, which is basically that of [29], does pose causality issues. In fact, the acceleration $\ddot{q}_n$ in Eq. (4.7) depends upon the control input $\bar{\tau}$ via the dynamic equation (2.14), i.e. $\ddot{q}_n = \ddot{q}_n(\bar{\tau})$. To avoid these causality concerns, one may think of substituting the acceleration $\ddot{q}_n$ in (4.4) with its expression deduced by the dynamical model (2.14). But in this case the obtained inverse dynamics

$$\begin{aligned}\bar{\tau} &= \bar{\tau}(\pi, q, \dot{q}, \ddot{q}_c)\\ &= (M_c - M_{nc}^\top M_n^{-1} M_{nc})\ddot{q}_c + (C_{cn} - M_{nc}^\top M_n^{-1} C_n)\dot{q}_n\\ &\quad + (C_c - M_{nc}^\top M_n^{-1} C_{nc})\dot{q}_c + (q_c - M_{nc}^\top M_n^{-1} q_n)\end{aligned}$$

is no longer linear with respect to the base parameters π, which destroys the stability and convergence arguments of Lemma 4.1.

The next theorem presents control laws that ensure the adaptive asymptotic stabilization of the collocated variables without acceleration feedback, thus avoiding causality concerns and circumventing the nonlinearity of the collocated inverse dynamics with respect to the base parameters.

Theorem 4.1. *Assume that Properties 2.1-2.3, 4.1 and Assumption 4.1 hold. Partition the variables ξ as follows:*

$$\xi := \begin{pmatrix} \xi_n \\ \xi_c \end{pmatrix}, \tag{4.9}$$

where $\xi_n \in \mathbb{R}^k$, $\xi_c \in \mathbb{R}^m$.

Apply the following control laws to System (4.4)

$$\bar{\tau} = Y_c(q, \dot{q}, \xi, \dot{\xi})\hat{\pi} - K s_c, \tag{4.10a}$$

$$\dot{\xi} = \begin{pmatrix} \dot{\xi}_n \\ \dot{\xi}_c \end{pmatrix} = \begin{pmatrix} \widehat{M}_n^{-1}\left[K_n s_n - Y_n\left(q, \dot{q}, \xi, \begin{pmatrix} 0_k \\ \dot{\xi}_c \end{pmatrix}\right)\hat{\pi}\right] \\ \ddot{r} - \Lambda_1 \dot{e} - \Lambda_2 e \end{pmatrix} \tag{4.10b}$$

$$\dot{\hat{\pi}} = -\Gamma Y^\top(q, \dot{q}, \xi, \dot{\xi}) s, \tag{4.11}$$

with $K, \Lambda_1, \Lambda_2 \in \mathbb{R}^{m\times m}$ and $K_n \in \mathbb{R}^{k\times k}$ diagonal, constant positive definite matrices, and the matrix $\widehat{M}_n$ defined as the kth *order leading principal minor of the mass matrix M evaluated with estimated base parameters, i.e.*

$$\widehat{M}_n := S_k M(q, \hat{\pi}) S_k^\top, \tag{4.12}$$

where the selector S_k is given by

$$S_k := \begin{pmatrix} I_k & 0_{k\times(n-k)} \end{pmatrix}. \tag{4.13}$$

Then, the following results hold.

i) The equilibrium point $\left(e, \tilde{\xi}_c, s, \tilde{\pi}\right) = (0_m, 0_m, 0_n, 0_p)$ of the closed loop dynamics $\left(\dot{e}, \dot{\tilde{\xi}}_c, \dot{s}, \dot{\tilde{\pi}}\right)$ is stable.

ii) Assume that the noncollocated velocities remain bounded, i.e. $\exists \delta > 0$ such that $|\dot{q}_n| < \delta \quad \forall t$. There exists a neighborhood $\mathcal{I}$ of the origin $(0_m, 0_m, 0_n, 0_p)$ such that if the initial condition $\left(e, \tilde{\xi}_c, s, \tilde{\pi}\right)(0)$ belongs to $\mathcal{I}$, then the tracking error $e(t)$ converges to zero.

The proof is given in Appendix A.2. The appeal of the invoked reformulation of the Slotine's adaptive control presented in Lemma 4.1 lies in the similarity between the control laws

(4.3) and (4.10)-(4.11). More precisely, in both cases the evolution of the variable ξ can be obtained by numerical integration of its dynamics $\dot{\xi}$. When the system possesses k unactuated degrees of freedom, it suffices to modify the first k elements of this dynamics – see Eq. (4.10b) – to still ensure stability and convergence of the collocated coordinates.

Note that the dynamics $\dot{\xi}_n$ in Eq. (4.10b) plays the role of an estimator for the noncollocated acceleration $\ddot{q}_n$ when the system's trajectories belong to a neighbourhood of the equilibrium point.

Convergence of the tracking error e to zero is guaranteed, however, when the noncollocated velocities $|\dot{q}_n|$ remain bounded. This requirement, which follows from the application of Barbalat's Lemma, is reminiscent of the condition on the stability of the zero dynamics in [93]. Let us remark that stability is here guaranteed independently from the boundedness of $\dot{q}_n$, which cannot be in general satisfied by an appropriate choice of the control input $\bar{\tau}$. In fact, the influence of this input on the noncollocated dynamics cannot be guaranteed to have general properties. Clearly, friction effects may play a role in guaranteeing the boundedness of $|\dot{q}_n|$, and consequently the convergence of the tracking error $e(t)$ to zero.

4.2.1 DESINGULARIZATION FOR A GLOBALLY DEFINED CONTROLLER

The local nature of the controls (4.10) is due to the fact that the matrix (4.12) may not be invertible far from the point $\tilde{\pi} = 0_p$. Observe that the invertibility of (4.12) in a neighbourhood of this point is guaranteed by Property 2.1, which implies that each leading principal minor of the mass matrix $M(q, \pi)$ is positive definite, and therefore invertible.

The non-invertibility of the matrix (4.12) is related to the *standard inertial parameters* associated with the estimated base parameters $\hat{\pi}$. The standard inertial parameters of a rigid body consist in a ten-dimensional vector composed of the mass, the three first moments of mass, and the six elements of the inertia matrix [46].

For instance, when an estimate $\hat{\pi}$ induces a negative mass of a rigid body composing the underlying mechanical system, the inertia matrix $M(q, \hat{\pi})$ may not be positive definite [115], thus eventually resulting in a ill-conditioned controller.

Now, let us remark that if

$$\det\Big(M_n(q(t), \hat{\pi}(t))\Big) > 0 \quad \forall t \tag{4.14}$$

independently of the initial conditions, the laws (4.10) ensure global convergence of the tracking error and global boundedness. This may not be always the case, however. To avoid a possible ill-conditioning of the laws (4.10), a desingularization policy must be defined when the above determinant gets close to zero.

The desingularization policy used in this paper exploits the following result on the inertia matrix $M(q,\pi)$.

Lemma 4.2. *Properties 2.1 and 4.1 imply that:*

$$\left[\partial_\pi \det\left(S_i M(q,\pi) S_i^\top\right)\right]\pi = i \det\left(S_i M(q,\pi) S_i^\top\right), \tag{4.15}$$

$\forall i \in \{1,\dots,n\}$ *and with* S_i *given by* (4.13)*. Then, the gradient with respect to* π *of the determinant of each leading principal minor of the mass matrix* $M(q,\pi)$ *has a norm different from zero, i.e. there exists* $\gamma > 0$ *such that*

$$\left|\partial_\pi \det\left(S_i M(q,\pi) S_i^\top\right)\right| > \gamma. \tag{4.16}$$

The proof is in Appendix A.3. The above lemma in turn implies that there exists a choice for $\dot{\hat{\pi}}$ such that the time derivative of (4.14) can be imposed at will. As a consequence, it is theoretically possible to modify the law (4.11) so as to ensure that the determinant of $\widehat{M}_n$ never decreases below a certain threshold. The next Proposition presents such a modification of the adaptation law $\dot{\hat{\pi}}$.

Proposition 4.1. *Consider the laws* (4.10) *with the adaptation law redefined as follows*

$$\dot{\hat{\pi}} = -\Gamma\left[Y^\top(q,\dot{q},\xi,\dot{\xi})s - \eta\delta\right], \tag{4.17}$$

with $\eta \in \mathbb{R}$ *and* $\delta \in \mathbb{R}^p$ *given by:*

$$\eta := \begin{cases} 0 & \text{if } \operatorname{tr}\left(\widehat{M}_n^{-1}\Upsilon\right) \geqslant 0 \text{ or } \det\left(\widehat{M}_n\right) > \varepsilon \\ -\frac{\operatorname{tr}(\widehat{M}_n^{-1}\Upsilon)}{\delta^\top\Gamma\delta} & \text{otherwise} \end{cases} \tag{4.18a}$$

$$\delta := \sum_{i=1}^{k} Y_{M_n}^\top(q,e_i)\widehat{M}_n^{-1}e_i, \tag{4.18b}$$

where $\varepsilon \in \mathbb{R}^+$,

$$Y_{M_n} := S_k\left[Y\left(q,0_n,0_n,\begin{pmatrix}e_i\\0_m\end{pmatrix}\right) - Y(q,0_n,0_n,0_n)\right] \tag{4.19a}$$

$$\Upsilon := (\upsilon_1,\dots,\upsilon_i,\dots,\upsilon_k), \tag{4.19b}$$

$$\upsilon_i := \left[\frac{\partial}{\partial q}(Y_{M_n}\hat{\pi})\right]\dot{q} - Y_{M_n}\Gamma Y^\top(q,\dot{q},\xi,\dot{\xi})s, \tag{4.19c}$$

and $e_i \in \mathbb{R}^k$ *denotes a vector of* k *zeros except for the* ith *coordinate, which is equal to one.*

Then, the following results hold.

i) If $\det\left(\widehat{M}_n\right) > 0$, *then*

$$|\delta| > 0 \quad \textit{and} \quad |\hat{\pi}| > 0.$$

ii) Assume that $\det\left(\widehat{M}_n\right)(0) > \varepsilon$. *Then,*

$$\det\left(\widehat{M}_n\right)(t) \geqslant \varepsilon \quad \forall t.$$

The proof is in Appendix A.4. This Proposition states that it is always possible to maintain the determinant of $\widehat{M}_n$ above a certain threshold ε. In fact, the desingularizing term η in (4.18) would be ill-conditioned only at $|\delta| = 0$, but this never occurs provided that $\det(\widehat{M}_n) > 0$ – see the result i). When compared to existing desingularization procedures, the main advantage of the above policy is that it does not affect the instantaneous value of the control torque $\bar{\tau}$, but only its derivative with respect to time. This characteristic is of a pivotal role in practice since it helps minimize the additional effort that the actuators must withstand close to the ill-conditioning point of the control laws.

In light of the above, the always-defined control laws are given by (4.10)-(4.17). Clearly, the larger the threshold ε, the larger the influence of the desingularizing term $\eta\delta$ on the results of Theorem 4.1. Consequently, this threshold must be tuned depending on the specific system. Assuming that one is given with best guesses $\bar{\pi}$ of the system's base parameters π, we suggest to set $\varepsilon = \det(M(\bar{q}, \bar{\pi}))$, with $\bar{q}$ a tunable parameter. For instance, if $r(t)$ converges to some desired values r_d, one may set $\bar{q} = (q_n^\top(0), r_d^\top)^\top$. Simulations and experimental results presented next show that the influence of this desingularizing term $\eta\delta$ does not significantly affect the practical stability and boundedness of the tracking error e.

Remark 4.1. We assumed that no external wrench acts on the dynamical system. The effects of an external measurable wrench f_k can be modelled as a disturbance

$$d := J_{C_k}^\top(q) f_k \in \mathbb{R}^n.$$

Now, partition

$$d = \begin{bmatrix} d_n \\ d_c \end{bmatrix}$$

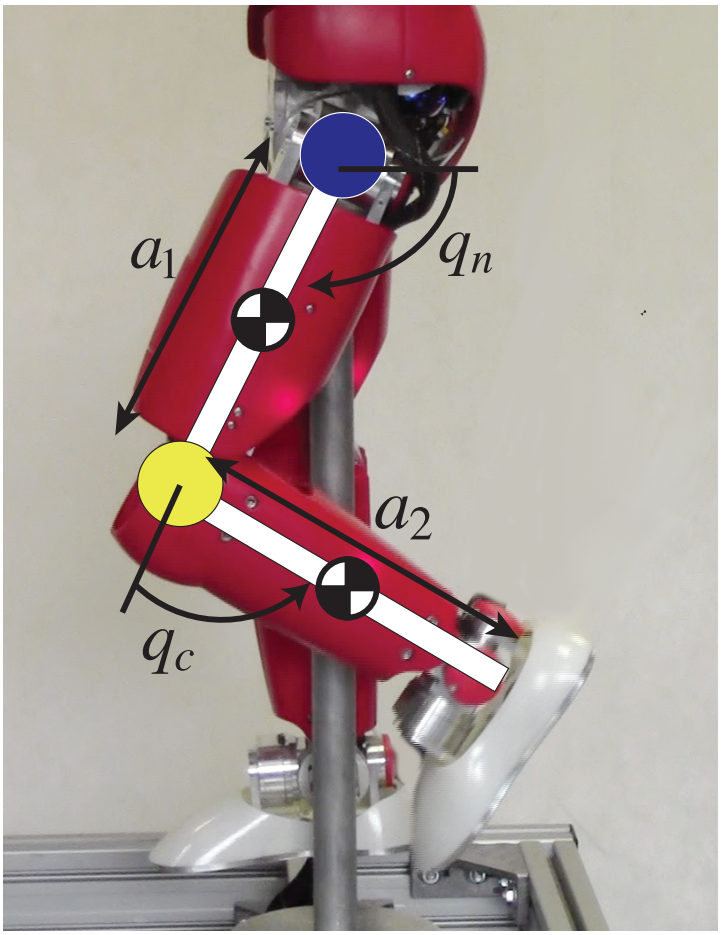


Figure 4.1: Two link manipulator obtained from the iCub's leg.

where $d_n \in \mathbb{R}^k$ and $d_c \in \mathbb{R}^m$. To retain stability and convergence of the collocated variables, it suffices to apply Eqs. (4.10) and (4.17) with:

$$\bar{\tau} = Y_c(q, \dot{q}, \xi, \dot{\xi})\hat{\pi} - Ks_c - d_c \ ,$$
$$\dot{\xi}_n = \widehat{M}_n^{-1}\left[K_n s_n + d_n - Y_n\left(q, \dot{q}, \xi, \begin{pmatrix} 0_k \\ \dot{\xi}_c \end{pmatrix}\right)\hat{\pi}\right] \ .$$

4.3 SIMULATIONS AND EXPERIMENTAL RESULTS

In this section, we test the control laws (4.10)-(4.17) first through simulations, and then through experiments carried out on a two-link manipulator with rotational joints.

The 2R manipulator is obtained from the *hip* – nonactuated joint – and the *knee* – actuated joint – of the iCub humanoid robot (see Figure 4.1). The robot's ankle is kept fixed with a position controller.

When applied to this case study, the laws (4.10)-(4.17) require the regressor $Y(\cdot)$ of a two-link manipulator. This regressor is computed with only viscous friction terms, which play a role when the controller is launched on the real robot. More precisely, the iCub platform is equipped with a low-level torque control loop that is in charge of stabilizing *any desired joint torque*. This loop is supposed to compensate for friction effects, but this compensation is never perfect. So, we added a friction terms $F_v\dot{q}$ in the regressor to account for imperfect viscous friction compensations by the low-level torque control. The base parameter vector is

$$\pi = \begin{bmatrix} \pi_1 & \pi_2 & \pi_3 & \pi_4 & \pi_5 & \pi_6 & \pi_7 & \pi_8 \end{bmatrix}^\top \in \mathbb{R}^8 \tag{4.20}$$

with

$$\begin{aligned} \pi_1 &= m_1 \\ \pi_2 &= m_1(l_1 - a_1) \\ \pi_3 &= I_1 + m_1(l_1 - a_1)^2 \\ \pi_4 &= m_2 \\ \pi_5 &= m_2(l_2 - a_2) \\ \pi_6 &= I_2 + m_2(l_2 - a_2)^2 \\ \pi_7 &= F_{v_1} \\ \pi_8 &= F_{v_2} \end{aligned}$$

where m_1 and m_2 are the masses of the two links, l_1 and l_2 are the positions of the links centers of mass, a_1 and a_2 are the known constant links lengths, I_1 and I_2 are the moments of inertia of the two links w.r.t. their center of mass. Finally F_{v_1} and F_{v_2} are the two joint viscous friction coefficients.

The corresponding regressor matrix $Y(q, \dot{q}, \xi, \dot{\xi})$

$$Y(q, \dot{q}, \xi, \dot{\xi}) = \begin{bmatrix} y_{11} & y_{12} & y_{13} & y_{14} & y_{15} & y_{16} & y_{17} & y_{18} \\ y_{21} & y_{22} & y_{23} & y_{24} & y_{25} & y_{26} & y_{27} & y_{28} \end{bmatrix} \in \mathbb{R}^{2\times 8}$$

m_1	2 kg	m_2	2 kg
I_1	0.2528 kg m²	I_2	0.2528 kg m²
l_1	0.75 m	l_2	0.75 m
a_1	1.5 m	a_2	1.5 m
F_{v_1}	1 N m s/rad	F_{v_1}	1 N m s/rad

Table 4.1: Physical parameters used in the simulations.

where the coefficients are defined as following:

$$
\begin{aligned}
y_{11} &= a_1^2\dot{\xi}_1 + a_1 g c_1 \\
y_{12} &= 2a_1\dot{\xi}_1 + g c_1 \\
y_{13} &= \dot{\xi}_1 \\
y_{14} &= (a_1^2 + 2a_1a_2c_2 + a_2^2)\dot{\xi}_1 + (a_1a_2c_2 + a_2^2)\dot{\xi}_2 \\
&\quad - a_2a_1s_2\dot{q}_2\xi_1 - a_1a_2s_2(\dot{q}_1 + \dot{q}_2)\xi_2 + a_1 g c_1 + a_2 g c_{12} \\
y_{15} &= 2(a_1c_2 + a_2)\dot{\xi}_1 + (a_1c_2 + 2a_2)\dot{\xi}_2 \\
&\quad - a_1s_2\dot{q}_2\xi_1 - a_1s_2(\dot{q}_1 + \dot{q}_2)\xi_2 + g c_{12} \\
y_{16} &= \dot{\xi}_1 + \dot{\xi}_2 \\
y_{17} &= \xi_1 \\
y_{18} &= 0 \\
y_{21} &= 0 \\
y_{22} &= 0 \\
y_{23} &= 0 \\
y_{24} &= (a_1a_2c_2 + a_2^2)\dot{\xi}_1 + a_2^2\dot{\xi}_2 + a_1a_2s_2\dot{q}_1\xi_1 + a_2 g c_{12} \\
y_{25} &= (a_1c_2 + 2a_2)\dot{\xi}_1 + 2a_2\dot{\xi}_2 + a_1s_2\dot{q}_1\xi_1 + g c_{12} \\
y_{26} &= \dot{\xi}_1 + \dot{\xi}_2 \\
y_{27} &= 0 \\
y_{28} &= \xi_2
\end{aligned}
$$

4.3.1 SIMULATIONS

Simulations are performed with the parameters listed in Table 4.1. The associated base parameters are

$$\pi = [2, -1.5, 1.3778, 2, -1.5, 1.3778, 1, 1]^\top.$$

The control input $\bar{\tau}$ is saturated at 73 [N m]. The initial conditions are

$$\xi(0) = q(0) = \dot{q}(0) = 0_2.$$

Figure 4.2 depicts the simulation result for the reference trajectory

$$r(t) = \frac{\pi}{2}\big(1 + \sin(\pi t)\big)$$

when the laws (4.10)-(4.17) are evaluated with

$$\begin{aligned} \Lambda_1 = \Lambda_2 = K &= 1_2 \\ \Gamma &= 1_8 \\ \varepsilon &= 1.5 \\ \hat{\pi}(0) &= [1.8, -1, 0.8, 2.6, -2, 1.7, 0.9, 1.1]\,. \end{aligned}$$

Convergence of the tracking error is achieved with an overshoot of 25°.

As for elements of comparison with existing control techniques, Figure 4.3 shows simulation results when applying the law (4.7) and the laws (4.10)-(4.17) with no feedforward term (i.e. $Y_c(q, \dot{q}, \xi, \dot{\xi})\hat{\pi} \equiv 0$), which result in a PID controller for all intents and purposes. The acceleration $\ddot{q}_n$ in (4.7) was estimated by using a filter of the form

$$\frac{s'}{(2\pi f s' + 1)'}$$

where s' is the Laplace variable, and f the cutoff frequency of the filter set at 10 [Hz]. By doing so, we basically compare our control strategy with that of [29]. Initial conditions, gains, and reference trajectory were kept equal to those associated with the simulation of Figure 4.2.

Interestingly enough, Figure 4.3 shows that the lack of the feedforward term $Y_c(q, \dot{q}, \xi, \dot{\xi})\hat{\pi}$ significantly worsens not only the steady state, but also the transient response. Clearly, increasing the PID's gains would reduce the error in this case, but this would result in amplifying eventual measurement errors and noises.

Figure 4.3 also shows that the law (4.7) evaluated with an estimated acceleration $\ddot{q}_n$ renders the variable q_c unstable. This instability is the combined effect of the torque saturation and the cutoff frequency of the filter used to estimate the acceleration $\ddot{q}_n$. Simulations we have performed tend to show that increasing the cutoff frequency reduce the likelihood of rendering the actuated variable unstable, but this may be problematic in practice because of well known issues such as high-frequency noise. Analogously, we verified that increasing the torque saturation would solve the instability problem, but this threshold may not be exceeded in practice.

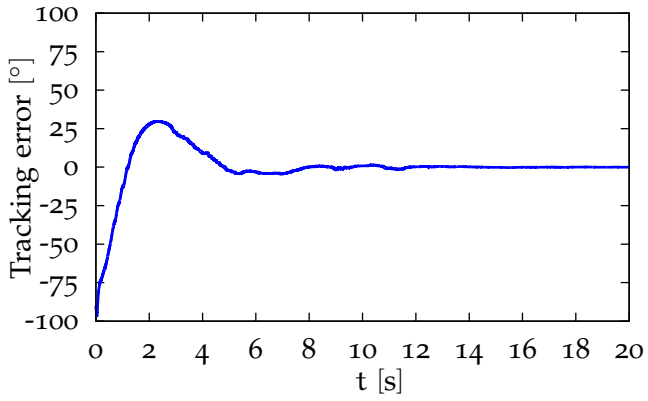


Figure 4.2: Performances of the control laws (4.10)-(4.17).

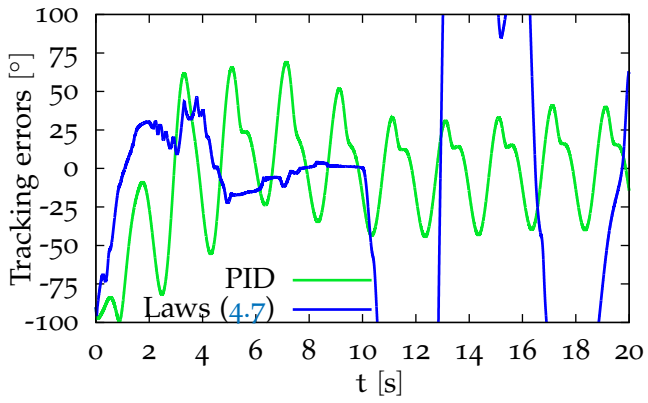


Figure 4.3: Performances of a PID and the control law (4.7).

4.3.2 EXPERIMENTS ON ICUB

We also applied the laws (4.10)-(4.17) to the aforementioned robot obtained from the iCub's leg. The reference trajectory was chosen as

$$r(t) = \frac{\pi}{4}\sin(2\pi f_r(t)t) - \frac{\pi}{3}\,,$$

with a piecewise constant frequency $f_r(t)$ given by

$$f_r(t) = \begin{cases} 0 \text{ Hz} & 0s \leqslant t < 20s, \\ 0.1 \text{ Hz} & 20s \leqslant t < 40s, \\ 0.2 \text{ Hz} & 40s \leqslant t < 58s, \\ 0.3 \text{ Hz} & 58s \leqslant t < 80s. \end{cases} \tag{4.21}$$

The control parameters are

$$\begin{aligned} \Lambda_1 = 6,\ \Lambda_2 = 15,\ K = 1, K_n = 1 \\ \Gamma = 0.2\ 1_8 \\ \varepsilon = 0.5 \\ e(0) = 40^\circ \\ \hat{\pi}(0) = [1.5,\ -0.1,\ 0.01,\ 2,\ -0.24,\ 0.08,\ 0.05,\ 0.05]\,. \end{aligned}$$

From top to bottom, Figures 4.4 and 4.5 depict the tracking error e, the estimated base parameters $\hat{\pi}$, the determinant of the matrix $M_n(q, \hat{\pi})$, the sliding variable s_c, the torque control input $\bar{\tau}$, and the velocity $\dot{q}_n$ of the noncollocated variable. Observe that the tracking error converges to zero for a constant reference r. Sharp variations of the tracking error and of the torque at the time instants $t = 20$ [s], $t = 40$ [s], and $t = 58$ [s] are due to the discontinuities of the reference trajectory $r(t)$. Note also that unmodeled friction effects and imperfect tracking of the low-level torque control loop reflect in $\dot{q}_n = 0$ close to $t = 20$ [s] (see Figure 4.5c). As the frequency f_r increases, the tracking error is kept relatively small despite a significant increase of the hip velocity (peak hip velocity at 40 [°/s]). The fact that the tracking error does not converge to zero is mainly due to the imperfect tracking of the low-level torque control loop implemented on the iCub platform.

Figure 4.4c shows the effects of the desingularization policy defined in Proposition 4.1. Once above the threshold ε, the determinant of $M_n(q(t), \hat{\pi}(t))$ never goes below this threshold. Observe that this desingularization action is of particular importance for high-frequency reference trajectories, where the coupling effects between the collocated and noncollocated joints are no longer negligible. Although stability and convergence are not guaranteed when the desingularization action is active, the estimated parameters remain bounded on $t \in (60, 80)$ [s] (see Figure 4.4b). We also want to remark that the parameters π chosen in (4.20) do no represent a minimum set. It is therefore not possible to compare the values of $\hat{\pi}$ obtained during the experiments with the real values, even if they had been available, or the values obtained by a classical estimation procedure.

4.4 DISCUSSION AND FUTURE WORK

This chapter presented an extension of the Slotine's adaptive control [99], which was developed for fully actuated manipulators, to stabilize the collocated space of an underactuated mechanical system. Stability and convergence of the collocated variables were shown by using Lyapunov and Barbalat arguments. Compared to existing results, our approach does not make use of any acceleration measurement, thus avoiding altogether causality concerns.

Applications of the presented approach include mechanical systems when the control task depends on the collocated variables only. Then, one can obtain the references for the collo-

cated variables associated with the control task, and stabilize them by means of the presented approach. The stabilization of the end effector of a manipulator attached to a space ship exemplifies this kind of applications [111].

The laws presented in this chapter render the associated equilibrium point stable, but convergence of the tracking errors is shown when the initial conditions belong to a neighbourhood of the equilibrium point. This local nature is due to the fact that the control laws make use of the invertibility of the system's inertia matrix along the estimated system's model. This matrix may not be invertible for physical inconsistent *base parameters* [115]. A possible extension of this work is thus to design an estimation dynamics such that the associated base parameters are always physical consistent. In this case, the presented control laws would guarantee global boundedness and convergence and the desingularization policy introduced in Proposition 4.1 would not be necessary anymore.

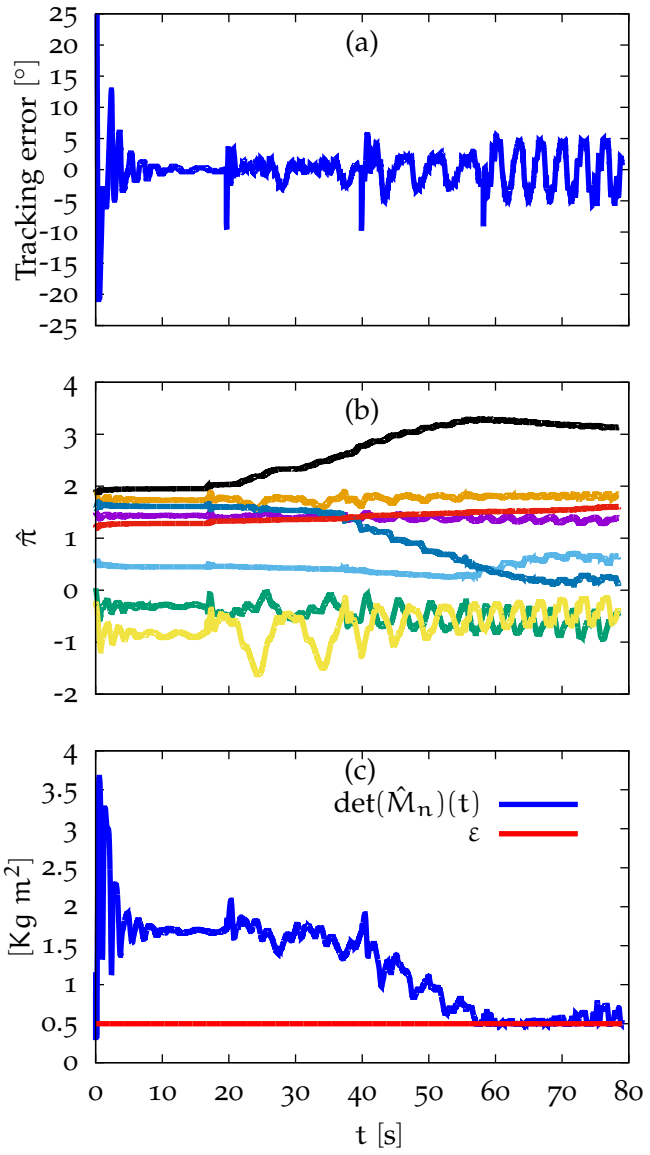


Figure 4.4: Plots associated with the experimental result. Figure b) shows the evolution of $\hat{\pi}$. Parameters are given with the following color order: purple, green, light blue, orange, yellow, dark blue, red, black, which correspond to $\hat{\pi}_1, \ldots, \hat{\pi}_p$.

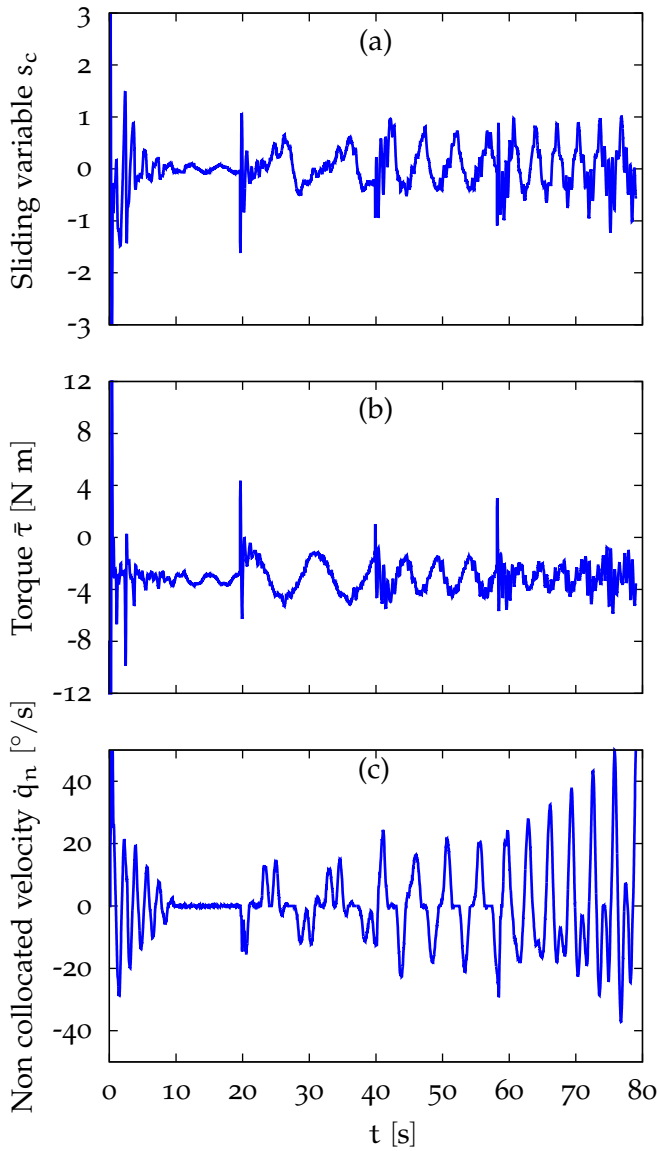


Figure 4.5: Plots associated with the experimental result.

5 PRIORITIZED OPTIMAL CONTROL

In this chapter we describe a novel control technique to control highly redundant mechanical systems. Prioritized Optimal Control tries to merge the advantages of state-of-the-art Prioritized Control and Optimal Control. This chapter first presents the problem formalization. We then show two different methods to solve the problem: a naive, yet simpler, method, and a more efficient one inspired by the Differential Dynamic Programming method.

5.1 PROBLEM FORMULATION

Let us consider a discrete-time nonlinear dynamical system

$$x_{i+1} = f(x_i, u_i), \quad \text{for } i = 0, \dots, N-1, \tag{5.1}$$

where $f(\cdot) : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^n$ is the dynamics function. Assume that the system has to perform K tasks, with task 1 having the highest priority, and task K the lowest. The k-th task is represented with an arbitrary cost function:

$$G^{(k)}(X, U) := \sum_{i=0}^{N-1} \phi^{(k)}(x_i, u_i) + \phi_N^{(k)}(x_N) \,, \tag{5.2}$$

where $\phi(\cdot) : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}$ is the running cost and $\phi_N(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}$ is the final cost and where the control and state trajectories are defined as

$$U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}, \quad X = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix}.$$

The control problem is to find the control input sequence U^* and state trajectory X^* that solve the following hierarchical optimal control problem, denoted by $\text{HOC}^{(k)}$:

$$\begin{aligned} \underset{U,X}{\text{minimize}} \quad & G^{(k)}(X,U) \\ \text{subject to} \quad & x_{i+1} = f(x_i,u_i), \quad \text{for } i = 0,\dots,N-1 \\ & x_0 \text{ fixed} \\ & G^{(j)}(X,U) = G^{(j)*} \quad \forall\, j < k\,, \end{aligned} \tag{5.3}$$

for $k = 1$ to K, where $G^{(j)*}$ is the optimum obtained by solving the $\text{HOC}^{(j)}$. We define the hierarchical optimal control problem composed of the K tasks as the solution of the cascade of $\text{HOC}^{(k)}$. This definition of "*cascade*" follows the construction made in [43] for hierarchical quadratic problems.

We now present two different approaches to solve the problem in (5.3). Both methods are based on the idea of iterative refinement. They start with an initial guess for the control solution which is iteratively refined by local approximation of the original nonlinear problem.

5.2 SOLVING THE PROBLEM BY CONSTRAINTS ELIMINATION

The hierarchical optimal control problem (5.3) can be solved by eliminating the dynamic constraints from the formulation.

The algorithm starts with an initial guess of the control input trajectory U, e.g. $\bar{U} \equiv 0$ as a trivial case. The dynamical system (5.1) in then "simulated", i.e. integrated, starting from the known initial state x_0, to obtain the corresponding state trajectory $\bar{X}$.

We first approximate the nonlinear problem by linearizing the dynamical system and by performing a second-order approximation of the nonlinear cost. Consider small variations of the state and the control (δx_i and δu_i) around the current trajectories. We can approximate the effect of these variations with a first-order Taylor expansion of the system dynamics:

$$\begin{aligned} \bar{x}_{i+1} + \delta x_{i+1} &\simeq f(\bar{x}_i,\bar{u}_i) + A_i \delta x_i + B_i \delta u_i \\ \delta x_{i+1} &\simeq A_i \delta x_i + B_i \delta u_i, \end{aligned} \tag{5.4}$$

where

$$A_i = \left.\frac{\partial f(x_i, u_i)}{\partial x_i}\right|_{\substack{x_i=\bar{x}_i \\ u_i=\bar{u}_i}} \in \mathbb{R}^{n\times n}$$
$$B_i = \left.\frac{\partial f(x_i, u_i)}{\partial u_i}\right|_{\substack{x_i=\bar{x}_i \\ u_i=\bar{u}_i}} \in \mathbb{R}^{n\times m}$$

Since $\delta x_0 = 0$, we can describe the whole evolution of the variational state in a single equation. We can define

$$\Delta X := \begin{bmatrix} \delta x_1 \\ \vdots \\ \delta x_N \end{bmatrix} \in \mathbb{R}^{nN}, \quad \Delta U := \begin{bmatrix} \delta u_0 \\ \vdots \\ \delta u_{N-1} \end{bmatrix} \in \mathbb{R}^{mN}$$

as the vectorization of the entire variational state and control trajectories and

$$\Gamma := \begin{bmatrix} B_0 & 0_{n\times m} & \cdots & 0_{n\times m} \\ A_1 B_0 & B_1 & \cdots & 0_{n\times m} \\ \vdots & & \cdots & \vdots \\ A_{N-1}\cdots A_1 B_0 & A_{N-1}\cdots A_2 B_1 & \cdots & B_{N-1} \end{bmatrix} \in \mathbb{R}^{nN\times mN}$$

as the matrix mapping the control trajectory to the state trajectory

We can thus write (5.4) for $i = 0, \cdots, N-1$ compactly as

$$\Delta X = \Gamma \Delta U.$$

5.2.1 PRIMARY TASK RESOLUTION

To simplify the derivation we assume that all the costs take this form:

$$G^{(k)}(X, U) = \frac{1}{2}\left\|c_k(X)\right\|^2 + \frac{1}{2}U^\top E_k U,$$

where $E_k \in \mathbb{R}^{mN\times mN}$ is a positive-semidefinite matrix, i.e. $E_k \geqslant 0$. This form is general enough for robotics applications: the state dependent cost $\left\|c_k(X)\right\|^2$ can represent any task-space tracking/reaching error, whereas the control-dependent cost $U^\top E_k U$ can penalize some form of effort. In practice, since the effort minimization consumes all the redundancy, E_k should be zero for all tasks but the last one.

Let us consider the effect that state and control variations have on the cost of task k. The cost variation can be written as:

$$\delta G^{(k)}(\Delta X, \Delta U) = G^{(k)}(\bar{X} + \Delta X, \bar{U} + \Delta U) - G^{(k)}(\bar{X}, \bar{U})$$

and for small variations, it can be approximated with its second order Taylor expansion in $X = \bar{X}$ and $U = \bar{U}$:

$$\begin{aligned}\delta G^{(k)}(\Delta X, \Delta U) \simeq c_k(\bar{X})^\top c_k' \Delta X + \bar{U}^\top E_k \Delta U + \tfrac{1}{2}\Delta U^\top E_k \Delta U \\ + \tfrac{1}{2}\Delta X^\top (c_k'^\top c_k' + c_k(\bar{X})^\top c_k'')\Delta X\end{aligned} \tag{5.5}$$

where $c_k' = \left.\frac{\partial c_k}{\partial X}\right|_{X=\bar{X}}$ and $c_k'' = \left.\frac{\partial^2 c_k}{\partial X^2}\right|_{X=\bar{X}}$.

We can now formulate an optimization problem to find the control that yields the minimum cost, subject to the dynamics, all written in the variational setting. In particular, for task $k = 1$ the optimization problem we solve is the following:

$$\begin{aligned}\delta G^{(1)*} = \min_{\Delta U \in \mathbb{R}^{mN}} \ & \delta G^{(1)}(\Delta X, \Delta U) + s_1 \|\Delta X\|^2 + r_1 \|\Delta U\|^2 \\ \text{st} \quad & \Delta X = \Gamma \Delta U,\end{aligned} \tag{5.6}$$

where $s_1 \in \mathbb{R}$ and $r_1 \in \mathbb{R}$ are strictly positive weights that penalize deviations from the current nominal solution $(\bar{X}, \bar{U})$. These penalties are fundamental because the linear approximation of the system dynamics is valid only in the vicinity of the current solution.

Since task 1 has the highest priority, this problem does not contain any constraints besides the system dynamics. To solve (5.6) we substitute the constraint into the cost function, and we set the derivative of the cost function with respect to ΔU equal to zero. Doing so we get:

$$S_1 \Delta U = d_1$$

where:

$$\begin{aligned} S_1 &= \Gamma^\top H_1 \Gamma + s_1 \Gamma^\top \Gamma + E_1 + r_1 1_{mN} \\ H_1 &= c_1'^\top c_1' + c_1(\bar{X})^\top C_1'' \\ d_1 &= -\Gamma^\top C_1^\top c_1(\bar{X}) - E_1 \bar{U} \end{aligned} \tag{5.7}$$

Since $r_1 > 0$, $E_1 \geqslant 0$ and by construction all the other terms are positive semidefinite, this system of equations has a unique solution, that is:

$$\Delta U_1^* = S_1^{-1} d_1$$

At this point we simulate the system using the new control trajectory $\bar{U} + \Delta U_1^*$ to obtain the new state trajectory $\hat{X} = \tilde{f}(x_0, \bar{U} + \Delta U_1^*)$. We then check whether the cost of the task has improved, i.e.:

$$G^{(1)}(\hat{X}, \bar{U} + \Delta U_1^*) < G^{(1)}(\bar{X}, \bar{U}) \tag{5.8}$$

If the cost has not improved, it means that we have moved into a region of the state/control space in which our approximations are no longer accurate. To improve the solution we iteratively increase the penalty weights (s_1, r_1) and recompute the solution of (5.6), until (5.8) is satisfied. Once we have found a ΔU_1^* that gives an improvement of the cost, we move to the following tasks.

5.2.2 SECONDARY TASKS RESOLUTIONS

Starting from the second task (i.e. $\forall\ k > 1$), the minimization is subject to the priority constraints, which impose to not alter the cost of the tasks with higher priority. In terms of variations we can express these constraints as:

$$\delta G^{(i)}(\Delta X, \Delta U) = \delta G^{(i)*} \qquad \forall i < k. \tag{5.9}$$

In general, nonlinear equality constraints needs to be approximated with a linear function in order to find analytical expressions of their solutions (as we did for the system dynamics). However, these nonlinear constraints have a special form that allows us to use quadratic approximations, which are more accurate. In fact, we exploit the fact that a convex quadratic function is equal to its minimum value if and only if its gradient is zero. This simple observation allows us to convert the quadratic approximations of the cost constraints (5.9) into linear constraints:

$$\frac{\partial \delta G^{(i)}}{\partial \Delta U}(\Delta X, \Delta U) = 0 \qquad \forall\ i < k.$$

Computing this derivative we get:

$$\underbrace{(\Gamma^\top H_k \Gamma + E_k)}_{T_k} \Delta U = d_k, \tag{5.10}$$

where H_k and d_k have already been defined in (5.7).

Typically we expect that, for all the tasks but the last one, the matrix $T_k \in \mathbb{R}^{mN \times mN}$ has a nontrivial nullspace, because $E_k = 0 \quad \forall\ k \neq K$. If that is the case, the constraint (5.10) admits infinite solutions:

$$\Delta U = T_k^+ d_k + \underbrace{(1_{mN} - T_k^+ T_k)}_{P_k} \Delta U_0,$$

where $\Delta U_0 \in \mathbb{R}^{mN}$ is an arbitrary vector and T_k^+ denotes the Moore-Penrose pseudoinverse of T_k.

We can now write the approximation of the problem (5.3) as

$$\begin{aligned} \delta G^{(k)*} = \min_{\Delta U \in \mathbb{R}^{mN}} \; & \delta G^{(k)}(\Delta X, \Delta U) + s_k \|\Delta X\|^2 + r_k \|\Delta U\|^2 \\ \text{s.t.} \quad & \Delta X = \Gamma \Delta U \\ & \Delta U = T_i^+ d_i + P_i \Delta U_0 \qquad \forall\, i < k. \end{aligned} \tag{5.11}$$

The problem consists in minimizing a quadratic function subject to linear equality constraints. It is thus possible to obtain its solution analytically. We eliminate the constraints by substituting them inside the cost function and then we set its derivative with respect to ΔU_0 equals to zero. As for the primary task, after computing the solution we must check that the cost has actually improved. Moreover, we check that the costs of the higher-priority tasks have not increased within a tolerance $\epsilon > 0$.

If the cost of either the current task or a higher-priority task has increased, then we must increase the penalty terms (s_k, r_k) and repeat the optimization.

To solve the whole cascade of minimizations we used Algorithm 1.

Algorithm 1 Resolution of the Linearized Hierarchy of Tasks.

procedure SOLVERSTEP($\bar{X}, \bar{U}, \Gamma, \{c\}, \{E\}, \{r\}, \{s\}$)
  $\Delta U \leftarrow 0$ ▷ Initialize variational control trajectory
  $P \leftarrow 1_{mN}$ ▷ Initialize nullspace projector
  for task $k = 1 \to K$ **do**
    $H \leftarrow C_k^\top C_k + c_k(\bar{X})^\top \frac{\partial C_k}{\partial X}$
    $T \leftarrow \Gamma^\top H \Gamma + E_k$
    $d \leftarrow -\Gamma^\top C_k^\top c_k(\bar{X}) - E_k \bar{U}$
  `WeightTuningLoop:`
    $S \leftarrow T + s_k \Gamma^\top \Gamma + r_k 1_{mN}$
    $\Delta U \leftarrow \Delta U + (SP)^+ (d - S\Delta U)$
    for task $i = 1 \to k$ **do**
      $\delta G^{(i)} \leftarrow$ computeCostVariation$(i, \bar{X}, \bar{U}, \Delta U)$
      if $\delta G^{(i)} > \epsilon$ **then**
        $(s_k, r_k) \leftarrow$ updatePenalties(s_k, r_k)
        go to `WeightTuningLoop`
      end if
    end for
    $P \leftarrow P - (TP)^+ TP$
  end for
  return ΔU
end procedure

5.2.3 PENALTY WEIGHT TUNING

The tuning of the penalty weights (s_k, r_k) is crucial for the fast convergence of the algorithm. Ideally, at each iteration one would like to find the values of (s_k, r_k) that result in the biggest reduction of the cost, but this search can be expensive. We can summarize our policy for tuning the penalty weights as follows. If the cost increases we multiply (s_k, r_k) times a factor $\mu > 1$ and repeat the optimization until we get an improvement in the cost. If the cost decreases we divide (s_k, r_k) by μ and recompute the solution. As long as the cost decreases, we continue decreasing (s_k, r_k), and we stop only when there is an increase in the cost. At each iteration we initialize (s_k, r_k) with the values that we found in the previous iteration.

5.2.4 STOP CRITERIA

We test convergence at the end of each main iteration, i.e. after each execution of Algorithm 1. After the iteration i, we have computed the state trajectory X_i and the control trajectory U_i. At this point, the algorithm stops if all the tasks have converged. A task has converged if at least one of these values is below the associated threshold:

- the absolute value of the cost $G^{(k)}(X_i, U_i)$;
- the relative cost improvement $\frac{|G^{(k)}(X_i,U_i)-G^{(k)}(X_{i-1},U_{i-1})|}{|G^{(k)}(X_{i-1},U_{i-1})|}$

5.2.5 ALGORITHM SUMMARY

We here summarize the Prioritized Optimal Control algorithm sketched in Algorithm 2. The input parameters are the initial state x_0, an initial guess for the control trajectory U_0, the state dependent costs c_k and the effort weights E_k for $k = 1, \cdots, K$. The initialization phase consists in initializing the penalty weights (used by the *SolverStep* procedure described in Algorithm 1) and computing the initial state trajectory with the associated costs. Then the main loop begins. Each iteration starts by computing the mapping matrix G and linearizing the nonlinear dynamics around the nominal trajectory (X, U), thus obtaining a time-varying linear dynamics. At this point the variational control trajectory are computed by using Algorithm 1. and the control and state trajectories (U, X) and the associated costs are updated. Finally the stopping criteria described in Section 5.2.4 are tested. If the criteria are met then the algorithm exits and

returns the computed control trajectory, otherwise it updates the costs and iterates the procedure.

Algorithm 2 Prioritized Optimal Control main iteration.

```
1: procedure PrioritizedOC(x_s, U_0, {c}, {E})
2:     {r}, {s} ← initializeWeights()
3:     U ← U_0
4:     X ← simulateTrajectory(x_s, U)
5:     {J} ← computeCosts(X, U, {c}, {E})
6:     loop
7:         {A_i}, {B_i}, {C_i} ← linearizeSystem(X,U)
8:         Γ ← computeΓ({A_i}, {B_i}, {C_i})
9:         ΔU ← SolverStep(X, U, Γ, {c}, {E}, {r}, {s})
10:        U ← U + ΔU
11:        X ← simulateTrajectory(x_s, U)
12:        {J_new} ← computeCosts(X, U, {c}, {E})
13:        if stopCriterion({J}, {J_new}) then
14:            Break
15:        end if
16:        {J} ← {J_new}
17:    end loop
18:    return U, {J}
19: end procedure
```

5.3 HIERARCHICAL DIFFERENTIAL DYNAMIC PROGRAMMING METHOD

We now introduce a different approach, based on Dynamic Programming and inspired from [61], to solve the Prioritized Optimal Control problem as formulated in (5.3). When comparing the two algorithms we will refer the one introduced in Section 5.2 as *POC* while the one we introduce in this section as *HDDP*.

5.3.1 ALGORITHM OUTLINE

Before delving into the mathematical details, let us introduce the principles of the algorithm. Similarly to the *POC* algorithm, each iteration consists of the following three phases:

1. Problem approximation.
2. Local control computation, or backward pass.
3. System simulation, or forward pass.

We start by approximating the dynamics and the cost along an initial nominal state-control trajectory. Then in the backward pass we compute the local control modification as the solution to a hierarchical optimal control problem for the approximated model. Because this control modification is based on a local model, we have to check how it performs on the real system (i.e. we integrate the dynamics with the new control trajectory and compute the new costs). The so-called line-search procedure takes care of reducing the magnitude of the control modification to compensate for the mismatch between approximated and real model. Finally we introduce a regularization procedure to solve two issues commonly arising in this class of algorithms: solutions far from the local validity of the model and ill-conditioning due to finite-precision arithmetic.

5.3.2 HDDP ALGORITHM: LINEAR PART

5.3.2.1 *Dynamic Programming*

We solve the problem (5.3) by applying the dynamic programming algorithm as introduced in Section 2.5.2. Let us start by defining the *cost-to-go* at step i for task k as:

$$G_i^{(k)}(x_i, U_i) := \sum_{j=i}^{N-1} \phi^{(k)}(x_j, u_j) + \phi_N^{(k)}(x_N) \ .$$

The optimal cost-to-go, or value function, is its minimum value:

$$V_i^{(k)}(x_i) := \min_{U_i} G_i^{(k)}(x_i, U_i) \ .$$

By applying Bellman's principle of optimality we get the recurrence equation of dynamic programming. The recurrence equation is:

$$V_i^{(k)}(x_i) := \min_{u_i} \phi^{(k)}(x_i, u_i) + V_{i+1}^{(k)}(f(x_i, u_i)) \ , \tag{5.12}$$

with $i = N-1, \cdots, 0$ initialized with $V_N^{(k)}(x_N) := \phi_N^{(k)}(x_N)$. To simplify the notation we will use the following definition:

$$\mathcal{V}_i^{(k)}(x_i, u_i) := \phi^{(k)}(x_i, u_i) + V_{i+1}^{(k)}(f(x_i, u_i)). \tag{5.13}$$

Solving the above equation in the nonlinear context is computationally infeasible even for low-dimensional state and control spaces (Bellman's curse of dimensionality). Instead, we iteratively solve local quadratic approximations of the value function.

5.3.2.2 *Hierarchical Dynamic Programming*

We start by considering a nominal control policy

$$\bar{U} := \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_{N-1} \end{bmatrix}$$

and the corresponding state trajectory

$$\bar{X} := \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_N \end{bmatrix}$$

resulting by applying the former control to the system (5.1) with initial condition x_0. We also denote the variation of the control and the state with respect to their nominal values as $\delta u_i := u_i - \bar{u}_i$ and $\delta x_i := x_i - \bar{x}_i$ respectively. With these definitions, we approximate (5.13) with its Taylor's series expansion truncated at the second order:

$$\begin{aligned} V_i^{(k)}(x_i, u_i) \approx{}& V_i^{(k)}(\bar{x}_i, \bar{u}_i) + Q_{x,i}^{(k)\top} \delta x_i + Q_{u,i}^{(k)\top} \delta u_i \\ &+ \frac{1}{2}\left(\delta x_i^\top Q_{xx,i}^{(k)} \delta x_i + \delta u_i^\top Q_{uu,i}^{(k)} \delta u_i\right) \\ &+ \delta x_i^\top Q_{xu,i}^{(k)} \delta u_i \,, \end{aligned} \tag{5.14}$$

where the coefficients are defined as:

$$\begin{aligned} Q_{x,i}^{(k)} &= \partial_x \phi^{(k)} + \partial_x f^\top \partial_x V_{i+1}^{(k)} \\ Q_{u,i}^{(k)} &= \partial_u \phi^{(k)} + \partial_u f^\top \partial_x V_{i+1}^{(k)} \\ Q_{xx,i}^{(k)} &= \partial_{xx} \phi^{(k)} + \partial_x f^\top \partial_{xx} V_{i+1}^{(k)} \partial_x f + \partial_x V_{i+1}^{(k)} \partial_{xx} f \\ Q_{uu,i}^{(k)} &= \partial_{uu} \phi^{(k)} + \partial_u f^\top \partial_{xx} V_{i+1}^{(k)} \partial_u f + \partial_x V_{i+1}^{(k)} \partial_{uu} f \\ Q_{xu,i}^{(k)} &= \partial_{xu} \phi^{(k)} + \partial_x f^\top \partial_{xx} V_{i+1}^{(k)} \partial_u f + \partial_x V_{i+1}^{(k)} \partial_{xu} f. \end{aligned} \tag{5.15}$$

All the derivatives are computed for $x_i = \bar{x}_i$ and $u_i = \bar{u}_i$.

Now that we have a quadratic model of $V_i^{(k)}(x_i, u_i)$, we can find the optimal control variation analytically:

$$\begin{aligned} \delta u_i^{(k)*} &= -Q_{uu,i}^{(k)\dagger} Q_{u,i}^{(k)} - Q_{uu,i}^{(k)\dagger} Q_{ux,i}^{(k)} \delta x_i + N_i^{(k)} \delta u_i^{(k+1)} \\ &= \delta \hat{u}_i^{(k)} - K_i^{(k)} \delta x_i + N_i^{(k)} \delta u_i^{(k+1)} \end{aligned} \tag{5.16}$$

where

$$\begin{aligned} \delta \hat{u}_i^{(k)} &:= -Q_{uu,i}^{(k)\dagger} Q_{u,i}^{(k)} \\ K_i^{(k)} &:= Q_{uu,i}^{(k)\dagger} Q_{ux,i}^{(k)} \,, \end{aligned}$$

$N_i^{(k)}$ is a projector onto $\mathcal{N}(Q_{uu,i}^{(k)})$ (i.e. the null space of $Q_{uu,i}^{(k)}$) and $\delta u_i^{(k+1)}$ is a free vector that will be chosen to minimize the task $k+1$. Typically in robotics the dimension of a task is much smaller than the number of DoFs of the robot, which ensures the existence of the null space.

ONE TASK RESOLUTION We start by solving the optimal control problem with only one task. In this simplified scenario the optimal control in (5.16) consists of the first two terms only, i.e. $\delta u_i^{(k+1)} \equiv 0$ for all i.

To obtain the value function at step i we substitute the control (5.16) into (5.14), thus obtaining the following quadratic form for the value function:

$$V_i(\delta x_i) = V_{s,i} + V_{x,i}^\top \delta x_i + \frac{1}{2}\delta x_i^\top V_{xx,i}\delta x_i \,, \tag{5.17}$$

where the scalar, linear and quadratic terms are defines as:

$$\begin{aligned} V_{s,i} &= Q_{u,i}^\top \delta\hat{u}_i + \frac{1}{2}\delta\hat{u}_i^\top Q_{uu,i}\delta\hat{u}_i \\ V_{x,i} &= Q_{x,i} - K_i^\top Q_{u,i} - K_i^\top Q_{uu,i}\delta\hat{u}_i + Q_{xu,i}\delta\hat{u}_i \\ V_{xx,i} &= Q_{xx,i} + K_i^\top Q_{uu,i}K_i - 2Q_{xu,i}K_i \end{aligned} \tag{5.18}$$

Equation (5.17) is solved backward in time starting from $i = N-1$ to 0 and initialized with the quadratic approximation of $\phi_N^{(k)}(x_N)$. As expected, for a single task, the solution of hierarchical differential dynamic programming coincides analytically with the solution of DDP [61].

MULTIPLE-TASK RESOLUTION When we solve the problem for a task $k > 1$ the control variable in no longer free. Indeed $\delta\hat{u}_i^{(j)}$ and $K_i^{(j)}$ have already been chosen for all tasks $j < k$. The control variable must then respect the following form:

$$\begin{aligned} \delta u_i =& [\delta\hat{u}_i^{(1)} + N_i^{(1)}\delta\hat{u}_i^{(2)} + \cdots + N_i^{(1)}\cdots N_i^{(k-2)}\delta\hat{u}_i^{(k-1)}] \\ & - [K_i^{(1)} + N_i^{(1)}K_i^{(2)} + \cdots + N_i^{(1)}\cdots N_i^{(k-2)}K_i^{(k-1)}]\delta x_i \\ & + N_i^{(1)}N_i^{(2)}\cdots N_i^{(k-1)}\delta u_i^{(k)} \\ =& \,\delta\bar{\hat{u}}_i^{(k-1)} - \bar{K}_i^{(k-1)}\delta x_i + \bar{N}_i^{(k)}\delta u_i^{(k)}, \end{aligned} \tag{5.19}$$

where we defined

$$\begin{aligned} \delta\bar{\hat{u}}_i^{(k-1)} :=& [\delta\hat{u}_i^{(1)} + N_i^{(1)}\delta\hat{u}_i^{(2)} + \cdots + N_i^{(1)}\cdots N_i^{(k-2)}\delta\hat{u}_i^{(k-1)}] \\ \bar{K}_i^{(k-1)} :=& [K_i^{(1)} + N_i^{(1)}K_i^{(2)} + \cdots + N_i^{(1)}\cdots N_i^{(k-2)}K_i^{(k-1)}] \\ \bar{N}_i^{(k)} :=& N_i^{(1)}N_i^{(2)}\cdots N_i^{(k-1)}. \end{aligned}$$

Substituting (5.19) into (5.14) we get updated values, denoted with $\overline{(\cdot)}$ for the coefficients in (5.15):

$$\begin{aligned}
\overline{Q}_{x,i}^{(k)} &= Q_{x,i}^{(k)} - \overline{K}_i^{(k-1)\top} Q_{u,i}^{(k)} + \overline{Q}_{xu,i}^{(k)} \delta\bar{\hat{u}}_i^{(k-1)} \\
\overline{Q}_{u,i}^{(k)} &= Q_{u,i}^{(k)} + Q_{uu,i}^{(k)} \delta\bar{\hat{u}}_i^{(k-1)} \\
\overline{Q}_{xx,i}^{(k)} &= Q_{xx,i}^{(k)} + \overline{K}_i^{(k-1)\top} Q_{uu,i}^{(k)} \overline{K}_i^{(k-1)} - 2Q_{xu,i}^{(k)} \overline{K}_i^{(k-1)} \\
\overline{Q}_{uu,i}^{(k)} &= \overline{N}_i^{(k-1)} Q_{uu,i}^{(k)} \overline{N}_i^{(k-1)} \\
\overline{Q}_{xu,i}^{(k)} &= Q_{xu,i}^{(k)} - \overline{K}_i^{(k-1)\top} Q_{uu,i}^{(k)} \,.
\end{aligned} \tag{5.20}$$

With the above redefinition we can compute the optimal control $\delta u_i^{(k)*}$ for the task k using (5.16) with the substitution $Q \leftarrow \overline{Q}$, i.e. with the following control:

$$\delta u_i^{(k)*} = -\overline{Q}_{uu,i}^{(k)\dagger} \overline{Q}_{u,i}^{(k)} - \overline{Q}_{uu,i}^{(k)\dagger} \overline{Q}_{ux,i}^{(k)} \delta x_i + \overline{N}_i^{(k)} \delta u_i^{(k+1)}.$$

Similarly, the value function at step i has the same form as the one in (5.17) provided that its coefficients in (5.18) are computed with $\overline{Q}$, that is:

$$\begin{aligned}
V_{s,i} &= \overline{Q}_{u,i}^{\top} \delta\bar{\hat{u}}_i + \frac{1}{2} \delta\bar{\hat{u}}_i^{\top} \overline{Q}_{uu,i} \delta\bar{\hat{u}}_i \\
V_{x,i} &= \overline{Q}_{x,i} - K_i^{\top} \overline{Q}_{u,i} - K_i^{\top} \overline{Q}_{uu,i} \delta\bar{\hat{u}}_i + \overline{Q}_{xu,i} \delta\bar{\hat{u}}_i \\
V_{xx,i} &= \overline{Q}_{xx,i} + K_i^{\top} \overline{Q}_{uu,i} K_i - 2\overline{Q}_{xu,i} K_i.
\end{aligned}$$

Equations (5.20) and (5.19) already outline the correct procedure to compute the control and the value-function update during the backward pass. It is clear that two sweeps on the task hierarchy are required: i) from $k = 1$ to K to compute the feedforward terms, the feedback gain matrices and the null-space projectors; ii) from $k = K$ to 1 to update the value function. Algorithm 3 summarizes the procedure.

5.3.3 HDDP ALGORITHM: NONLINEAR HEURISTIC

The procedure described in the previous section operates on a local approximation of the original problem (5.3). It is thus necessary to introduce some procedures to ensure that the solutions obtained on the local model yield a descent of the cost in 5.3. This section presents the line search and the regularization procedure applied in the algorithm.

Algorithm 3 Hierarchical Linear Solver

Initialize $V_{x,N}^{(k)}$ and $V_{xx,N}^{(k)}$ $\forall$ task k
for $i = N-1$ to 0 **do**
  for $k = 1$ to K **do**
    Update $\delta\bar{\hat{u}}_i^{(k-1)}$, $\bar{K}_i^{(k-1)}$ and $\bar{N}_i^{(k-1)}$
    Compute $Q_{x,i}^{(k)}$, $Q_{u,i}^{(k)}$, $Q_{xx,i}^{(k)}$, $Q_{uu,i}^{(k)}$ and $Q_{xu,i}^{(k)}$
    Compute $\bar{Q}_{x,i}^{(k)}$, $\bar{Q}_{u,i}^{(k)}$, $\bar{Q}_{xx,i}^{(k)}$, $\bar{Q}_{uu,i}^{(k)}$ and $\bar{Q}_{xu,i}^{(k)}$
    $\delta\hat{u}_i^{(k)} = -\bar{Q}_{uu,i}^{(k)\dagger}\bar{Q}_{u,i}^{(k)}$
    $K_i^{(k)} = \bar{Q}_{uu,i}^{(k)\dagger}\bar{Q}_{ux,i}^{(k)}$
    $N_i^{(k)} \leftarrow$ projector onto $\mathcal{N}(\bar{Q}_{uu,i}^{(k)})$
  end for
  for $k = K$ to 1 **do**
    Compute $V_{x,i}^{(k)}$ and $V_{xx,i}^{(k)}$
  end for
  $\delta\hat{u}_i = \delta\bar{\hat{u}}_i^{(1)}$, $K_i = \bar{K}_i^{(1)}$
end for
return $\{\delta\hat{u}_i, K_i, i = 0, \ldots, N-1\}$

5.3.4 REGULARIZATION PROCEDURE

Regularization procedures help attenuating numerical issues arising from the finite-precision arithmetic and the local validity of the approximated model.

We introduce two regularization procedures. The first one, which resembles the Levenberg modification for the nonlinear least-squares problem, penalizes state deviations from the nominal state trajectory. A parameter $\lambda^{(k)}$ regulates the intensity of the regularization, which is applied in (5.15) to the definitions of $Q_{uu,i}^{(k)}$ and $Q_{xu,i}^{(k)}$:

$$\begin{aligned} Q_{uu,i}^{(k)} &= \partial_{uu}\phi^{(k)} + \partial_u f^\top(\partial_{xx}V_{i+1}^{(k)} + \lambda^{(k)}I_n)\partial_u f \\ Q_{xu,i}^{(k)} &= \partial_{xu}\phi^{(k)} + \partial_x f^\top(\partial_{xx}V_{i+1}^{(k)} + \lambda^{(k)}I_n)\partial_u f \,, \end{aligned}$$

where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix.

The second regularization consists in damping the pseudoinverses [57] with a parameter $\mu^{(k)}$. Note that the null-space projectors must be computed with the undamped pseudoinverses to ensure the proper hierarchy propagation.

5.3.5 LINE-SEARCH PROCEDURE

Another important feature of the proposed algorithm consists in adopting a custom line-search procedure, which reduces the

optimization step computed on the quadratic approximation. More in details, the control-policy update rule (5.16) consists in a feedback $K_i(x_i - \bar{x}_i)$ and a feed forward $\delta\hat{u}_i^{(k)}$ term. The latter in particular, might significantly degrade the policy optimality and should therefore be carefully chosen by a suitable line-search procedure, complicated by the hierarchical nature of the algorithm.

The line search adopted in our algorithm is the following: a set of constants $\nu^{(k)} \in [0, 1]$, one for each task, is chosen and the control:

$$u_i = \bar{u}_i + \sum_{k=1}^{K} \nu^{(k)} \delta\hat{u}_i^{(k)} - K_i(x_i - \bar{x}_i)$$

is applied to the system (5.1). The policy for selecting the step size $\nu^{(k)}$ consists in initializing $\nu^{(k)} = 0$ for all the tasks. We then start from the highest-priority task traversing the whole hierarchy down to the last task. At a generic task k we set $\nu^{(k)} = 1$ and we progressively decrease it until all the costs decrease in a lexicographic order, i.e. a decrease of the cost for the task k must not lead to an increase of the cost of any tasks $j < k$.

5.3.6 ALGORITHM SUMMARY

We now summarize the algorithm proposed to solve a cascade of hierarchical optimal control problems. An iteration of the main algorithm is composed of the following phases:

1. Problem approximation.
2. Local control computation, or backward pass.
3. System simulation, or forward pass.

Starting from an initial nominal trajectory we approximate the system and the costs and we compute a control modification. In the forward pass, we simulate the system and compute the new cost for every task. In this phase we perform the line-search procedure. In case of no improvements, we increase the regularization parameters and repeat the backward pass. If at least one task has improved, we decrease its regularization parameter and accept the iteration.

Convergence is tested at the end of each iteration. The convergence criteria is similar to the one in Section 5.2.4: it consists in an absolute criterion and a relative one. We assume that the algorithm has converged if the cost is lower than the absolute tolerance value. Alternatively, the relative improvement between

two successive iterations must be smaller than a relative tolerance value.

Algorithm 4 summarizes in pseudo code the hierarchical dynamic programming algorithm.

Algorithm 4 Hierarchical Differential Dynamic Programming

Require: Given x_0 and $\bar{U}$
Ensure: $\bar{U}, K_i^*$
  $K_i^* \leftarrow 0$
  $\bar{X} \leftarrow$ FORDWARDDYNAMICS$(x_0, \bar{U}, K^*)$
  $G^{(k)} \leftarrow$ COMPUTECOST$(\bar{X}, \bar{U})$
  Initialize regularization parameters: $\lambda \leftarrow \lambda_0$, $\mu \leftarrow \mu_0$
5: **loop**
    for $k = 1, \cdots, K$ and $i = 0, \cdots, N$ **do**
      Compute $\partial_{(\cdot)}\phi^{(k)}$, $\partial_x f$ and $\partial_u f$
    end for
    $\{\delta\hat{u}_i^{(k)}, K_i\} \leftarrow$ LINEARSOLVER$(\partial_{(\cdot)}\{\phi^{(k)}, f\}, \mu, \lambda)$
10: $\{U_{new}, G_{new}^{(k)}\} \leftarrow$ LINESEARCH$(\delta\hat{u}_i^{(k)}, K_i)$
    if all$(G_{new}^{(k)} - G^{(k)} > 0)$ **then**
      increase λ and μ
      goto backward pass
    else
15:   decrease $\lambda^{(k)}$ and $\mu^{(k)}$ for improved tasks
    end if
    $\bar{U} \leftarrow U_{new}$, $G^{(k)} \leftarrow G_{new}^{(k)}$, $K_i^* \leftarrow K_i$
    if converged **then**
      break
20: **end if**
  end loop
  return $\{\bar{U}, K_i^*\}$

5.4 COMPUTATIONAL COST ANALYSIS

The main advantages of the algorithm presented in Section 5.3 w.r.t to one presented in Section 5.2 lies in its computational complexity. We now analyze and compare the computational cost of the two algorithms focusing on the linear part.

5.4.1 HDDP COMPUTATIONAL COMPLEXITY ANALYSIS

We start by considering a single iteration of the linear part of the algorithm. At each iteration we compute the coefficients $\bar{Q}_{(\cdot)}^{(k)}$. This step consists of a series of matrix-matrix multiplica-

tions and matrix-vector multiplications ($O(r^3)$, where r is the largest dimension of the matrices involved). To compute the pseudoinverse and the null-space projector we have to perform an orthogonal decomposition of the matrix $\overline{Q}_{uu}^{(k)}$, e.g. we chose the SVD decomposition. Depending on the algorithm the cost can vary, but it is proportional to $O(m^3)$, being $\overline{Q}_{uu}^{(k)} \in \mathbb{R}^{m \times m}$. The value-function update step consists of matrix-matrix and matrix-vector multiplications too. Its cost is still in the order of $O(r^3)$, with r defined as before. To summarize, and keeping only the relevant terms, the above computations are performed for each of the K tasks and for all the N time steps thus yielding the total cost of $O(N\ K\ m^3)$.

5.4.2 HDDP AND POC COMPARISON

POC did not take advantage of the intrinsic sparsity of the optimal control problem. Indeed at every iteration a matrix $S \in \mathbb{R}^{mN \times mN}$ is formed and then decomposed with the SVD algorithm. Its computational cost amounts at $O(N^3\ m^3)$. The total algorithmic cost is dominated by the above decomposition. We can thus neglect the cost due to matrix multiplications. The total complexity of the linear part of the algorithm is $O(N^3\ K\ m^3)$.

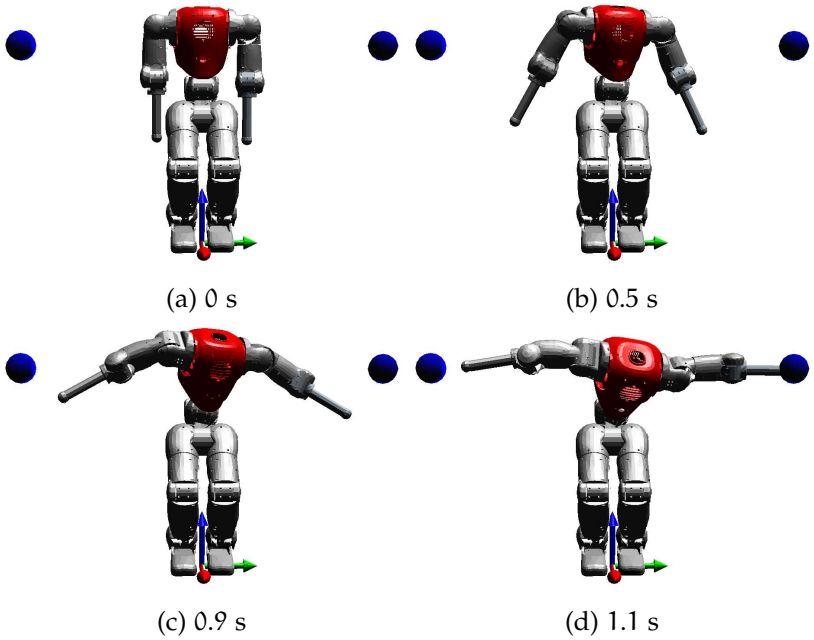
Comparing the two costs the difference in complexity is clear: POC is cubic in the number of time steps while HDDP is linear.

5.5 SIMULATION RESULTS

This section presents the results of the application of Priopritized Optimal Control in simulation. We present a first set of tests where we compare our algorithm with classical optimal control, i.e. by weighting all the tasks together in a single cost functional, and with instantaneous prioritized control. In the second series of tests we compare the performances of *POC* and *HDDP* on the same task.

5.5.1 EXPERIMENTAL SETUP

All tests have been conducted on a workstation with an Intel Xeon quad core at 3.2 GHz with 8 GB of RAM. We tested the three algorithms on a customized version of the Compliant huManoid (CoMan) simulator [11]. The base of the robot was fixed because we used only its upper body, which counts 11 DoFs: 4 in each arm and 3 in the torso. All the algorithms and the dynamics computation have been coded in Matlab 2012b.



(a) 0 s (b) 0.5 s

(c) 0.9 s (d) 1.1 s

Figure 5.1: Screenshots of the simulation. The two blue balls represent the targets of the left (high priority) and right (low priority) end-effectors of the robot.

5.5.2 TEST DESCRIPTION

We controlled the humanoid robot in order to reach three cartesian points with three different parts of its body, while minimizing the effort (i.e. the norm of joint torques). In order of decreasing priority, we controlled the position of the left-arm end-effector, right-arm end-effector and the top of the torso. The three tasks were individually feasible, but impossible to achieve at the same time. This allowed us to check whether the task priorities were satisfied: we expected the left end-effector to perfectly reach its target, while we expected significant errors on the right end-effector and the top torso.

In the first two tests, we discretized the equations of motion of the system by using a first-order explicit Euler scheme with a fixed time step of 20ms. We set the time horizon to 1s, which resulted in a total of $N = 50$ time steps. In all tests, we initialized the control trajectory with gravity compensation torques, so that the robot maintained the initial state for the whole time horizon. Fig. 5.1 reports some screenshots of the simulation, showing also two reference points.

We tested our approach on a customized version of the Compliant huManoid (CoMan) simulator [11]. The base of the robot was fixed because we used only its upper body, which counts 11 DoFs: 4 in each arm and 3 in the torso. We integrated the

Table 5.1: Values of the parameters of the algorithm used in the tests.

Parameter	Value	Parameter	Value
Initial s_k	1	Absolute Cost Threshold	10^{-6}
Initial r_k	10^{-2}	Relative Cost Threshold	10^{-2}
μ	1.5	Pseudoinverse Tolerance	10^{-5}
ϵ	10^{-6}		

equations of motion with the Simulink variable step integrator *ode23t* (relative tolerance 10^{-3}, absolute tolerance 10^{-6}). Table 5.1 lists the values used during our tests for all the parameters of the algorithm.

5.5.3 COMPARISON WITH CLASSICAL OPTIMAL CONTROL

In this test we compared the performances of our algorithm with those of classical optimal control. To approximate strict priorities with classical optimal control we minimize the following cost function:

$$g(X, U) = w^2 g_1(X, U) + w g_2(X, U) + g_3(X, U), \tag{5.21}$$

where $w \in \mathbb{R}$ is a user-selected weight parameter. Table 5.2 shows the results obtained with different values of w. By increasing w, the error of the primary task got lower, while the errors of the other tasks became higher. However, increasing w did not yield a clear trend on the secondary tasks. This outlines the difficulty of tuning the task weights. We also verified that, whenever we changed the tasks, we had to repeat the weight-tuning procedure.

During the tests, we noticed different problems as we incremented w. First of all, the number of iterations needed to converge increased (see Table 5.2). Second, we had to increase the initial value of the penalty weights (s, r) in the linear approximation of the system. Failing to do that resulted in an early termination of the optimization problem for numerical reasons. Third, we had to adapt the tolerances used in the convergence criteria. Since the cost function is a weighted sum of the task errors, its value has not a consistent unit of measurement.

Conversely, our method managed to obtain an acceptable error on the first task (0.5 mm), while obtaining significantly smaller errors on the second and third tasks. The number of iterations needed to converge was lower, but one iteration of our algorithm is in general more expensive, so this comparison is not fair.

Table 5.2: Comparison between classical Optimal Control and Prioritized Optimal Control. Each column shows error norm for three different choices of the weight w and the number of iterations needed to compute the solution. The last column shows the same results for our algorithm.

	Classical Optimal Control			Prioritized
	$w = 10^2$	$w = 10^3$	$w = 10^4$	O.C.
Task 1 Err. [μm]	530	54	27	507
Task 2 Err. [mm]	142	307	290	107
Task 3 Err. [mm]	359	426	425	305
Iterations	22	54	75	18

5.5.4 COMPARISON WITH PRIORITIZED CONTROL

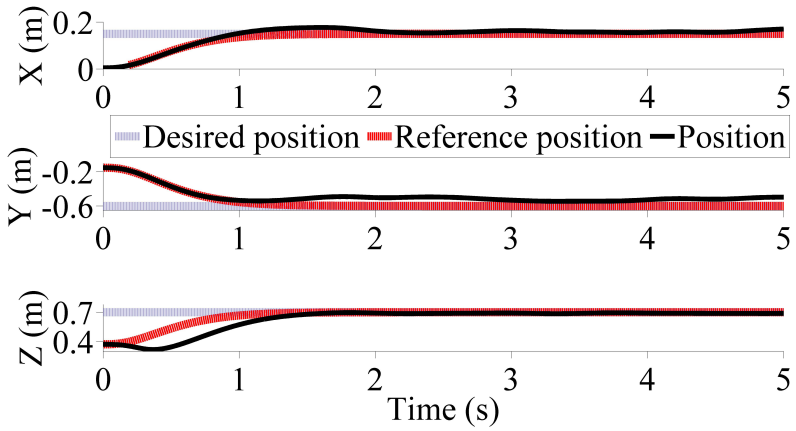
In this test we used an implementation of the prioritized control formulation presented in [82]. To prevent deviations from the desired trajectory and to ensure disturbance rejection, the controller computed the desired task-space accelerations $\ddot{x}^d \in \mathbb{R}^3$ with a proportional-derivative feedback control law:

$$\ddot{x}^d = \ddot{x}_r - K_d(\dot{x} - \dot{x}_r) - K_p(x - x_r),$$

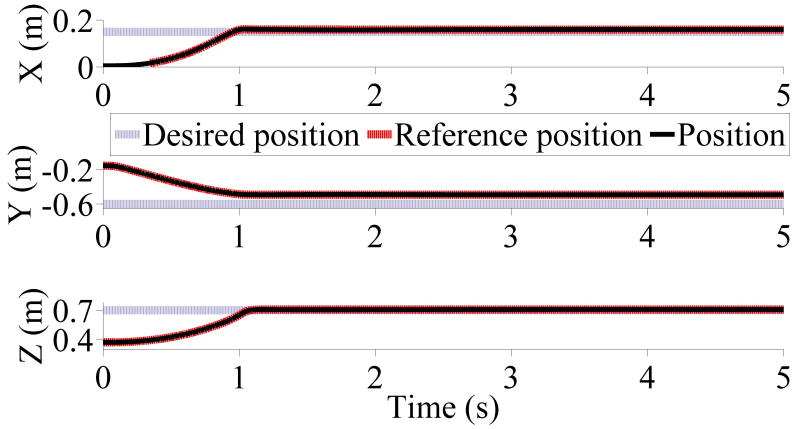
where $x_r(t), \dot{x}_r(t), \ddot{x}_r(t) \in \mathbb{R}^3$ are the position, velocity and acceleration reference trajectories, whereas K_d and $K_p \in \mathbb{R}^{3\times3}$ are diagonal matrices acting as derivative and proportional gains, respectively. We set all the diagonal entries of K_p to $10s^{-2}$, and all the diagonal entries of K_d to $5s^{-1}$. The controller also includes a simple trajectory generator [79], which provides approximately-minimum-jerk trajectories.

Prioritized control tends to make robots unstable when trying to move close to a singularity. A common approach to avoid instability is to use damped pseudoinverses [10], which however degrade tracking performances. In this test the robot reached a singular configuration, because it had to completely stretch both arms to get as close as possible to the targets. Using Moore-Penrose pseudoinverses resulted in instability, despite the low control period of 1 ms. With prioritized control, we had to set the damping factor to 0.1 to avoid instability. Despite the high damping factor (typically we set it to 0.02), the robot exhibited an oscillatory behavior around the goal configuration. Fig. 5.2 depicts the trajectory of the right hand of the robot, clearly showing the oscillations. The left end-effector manifested a similar behaviour.

Our algorithm generates open-loop trajectories, which cannot be used without a stabilizer. We decided to use the same



(a) Prioritized control. The reference trajectory is a quasi-minimum jerk trajectory, designed to reach the desired target in about 1 s



(b) Prioritized control + Prioritized optimal control. The reference trajectory is found by our algorithm with a time horizon of 1 s

Figure 5.2: Test 2. Comparison of right hand reaching between prioritized control (a) and "prioritized control + prioritized optimal control" (b).

prioritized controller as stabilizer for the trajectories computed by our method. In this case we could safely lower the damping factor of the pseudoinverses to 0.01, because the reference trajectories were always feasible. The improvement in the resulting behaviour was remarkable: the controller was able to track the trajectories with negligible errors. The robot reached the goal configuration and Fig. 5.2 shows no oscillation around the final positions for the right end-effector. Also the left end-effector and the top of the torso were stable.

5.5.5 PERFORMANCE TEST: POC VS HDDP

The last test we present is a performance test between the two algorithms described in this chapter, namely POC and HDDP.

The reaching cost functions contain only a final cost, i.e. it takes the form $G^{(k)}(X, U) \equiv \phi_N^{(k)}(x_N)$, being the squared norm of the distance between the end-effector and the target position:

$$\phi_N^{(k)}(x_N) = \|h(x_N) - \bar{h}\|^2 ,$$

where $h(x) : \mathbb{R}^n \mapsto \mathbb{R}^3$ is the function mapping the state variable x to a cartesian position, and $\bar{h}$ is its reference value.

We tested three different scenarios: i) two conflicting reaching tasks for the left and right arm end-effectors (Test 1); ii) three conflicting reaching tasks, using both arms and the torso (Test 2); iii) two compatible reaching tasks for the left and right arm end-effectors (Test 3). In all tests we added an effort task at the bottom of the hierarchy, which *removes* any left redundancy. We repeated the tests 100 times changing the original task targets of a value randomly sampled from a uniform distribution between 0 and 5cm. Table 5.3 reports the mean value of the final costs and of the total computation time for the three scenarios.

For both algorithms the absolute and relative tolerance for the stopping criteria have been set to $10^{-8}[m^2]$ and 10^{-4}, respectively.

Table 5.3 shows that, as far as the final error is concerned, HDDP manages to achieve a slightly better performance in all three scenarios. Unexpectedly, in Test 3, in which the tasks are compatible, POC yields some errors, especially in the secondary task, while HDDP manages to achieve almost zero errors.

From the computational standpoint HDDP is faster then POC in every test as we expected. Note that the CPU time is not directly proportional to the number of iterations in the HDDP algorithm. This is mainly due to the line-search procedure: big values of $\nu^{(k)}$ mean few forward-dynamics simulations, resulting in faster iterations.

5.6 DISCUSSION AND FUTURE WORK

In this chapter we proposed a new approach for the control of highly redundant robotic systems. The main advantage of this approach, drawing from Prioritized Control, is the possibility to define the tasks to be accomplished in an ordered sequence of increasing priority.

We then propose two different methods to solve the problem. While the second one is more efficient from the computational point of view, the first one can be more easily extended to in-

Table 5.3: Comparison between POC and HDDP. Task errors are in $[m^2]$. Values are average of 100 trials.

		Error $[m^2]$			Iter.	CPU Time
		Task 1	Task 2	Task 3		
Test 1	POC	$4.07\ 10^{-7}$	$1.59\ 10^{-2}$	N.A.	13	139s
	HDDP	$1.87\ 10^{-8}$	$1.22\ 10^{-2}$	N.A.	411	107s
Test 2	POC	$5.03\ 10^{-7}$	$1.39\ 10^{-2}$	$2.87\ 10^{-2}$	28	511s
	HDDP	$1.42\ 10^{-8}$	$1.24\ 10^{-2}$	$2.24\ 10^{-2}$	382	136s
Test 3	POC	$6.73\ 10^{-7}$	$1.44\ 10^{-5}$	N.A.	21	206s
	HDDP	$4.75\ 10^{-9}$	$3.98\ 10^{-9}$	N.A.	281	187s

corporate inequalities constraints: it suffices to substitute the analytical solution to the minimization problem with a hierarchical QP solver.

In this regard the introduction in the formulation of inequalities, such as joint limits and torque limits, is fundamental for its application to a real system.

A contact-aware forward dynamic routine is also necessary, being humanoid robots in contact with the environment. From this point of view, the two algorithms are totally agnostic on the kind of forward-dynamic implementation, thus their modification is not foreseen.

Part III

MOMENTUM-BASED TORQUE CONTROL FOR WHOLE-BODY BALANCING

6 BALANCING IN THE CONTEXT OF WHOLE-BODY CONTROL

6.1 WHOLE-BODY CONTROL OF FREE FLOATING MECHANICAL SYSTEMS

Whole-body control focuses in controlling the behaviour of the robot as a single entity. This means that all the possible constraints should be simultaneously taken into account when finding a solution to a user objective. Control objectives are usually specified as an output function associated with the dynamical system (2.14). More specifically, the objective can be expressed as the regulation to zero of some n-dimensional continuous and twice differentiable function called *task function* [91].

A possible approach to solve a whole-body control problem is to formulate it as a constrained optimization problem. In a general form we can state it as

$$\begin{aligned} &\underset{x}{\text{minimize}}\ f(x) \\ &\text{subject to dynamic constraints} \\ &\qquad\qquad \text{contact constraints} \\ &\qquad\qquad \text{equality constraints} \\ &\qquad\qquad \text{inequality constraints,} \end{aligned} \tag{6.1}$$

where $x \in \mathbb{X}$ and $f : \mathbb{X} \to \mathbb{R}$ is the cost function.

Problem (6.1) denotes a general optimization problem. In the context of whole-body control the cost function $f(x)$ is in general a quadratic function of the optimization variables.

How we choose the constraints, the cost, and the optimization variables highly depends on the various approaches. A further level of separation between the various approaches is if the cost function $f(x)$ includes all the possible tasks, usually momenta and motion tasks, or if they are explicitly solved in a hierarchical approach.

Our whole-body control architecture has the following control objectives:

i) Control of the robot momenta through the contact forces.

ii) Stability of the zero dynamics.

Each control objective is represented by a corresponding task, and they are organized in a task hierarchy. The control of the robot momenta has the highest priority, while the stability of the zero dynamics has the lowest.

6.1.1 FIRST TASK: CONTROL OF ROBOT MOMENTA

The control of the equilibrium of the robot has been associated in literature with the control of the linear and angular momentum of the system.

The linear and angular momentum of the robot have been defined in Section 2.2.3 and they are related to the external forces acting on the system. We can summarize (2.10) and (2.12) compactly as

$$\dot{H} = \begin{bmatrix} \dot{P} \\ \dot{L}_G \end{bmatrix} = \sum_{k=1}^{n_c} {}^{c}X^{*}_{\mathcal{C}_k} f_k + mg \tag{6.2}$$

where $g \in \mathbb{R}^6$ is the gravity acceleration vector, f_k is the k-th wrench applied at the frame $\mathcal{C}_k$, see discussion after (2.14), expressed at the frame $\mathcal{C}_k$ and ${}^{c}X^{*}_{\mathcal{C}_k} \in \mathbb{R}^{6\times6}$ is the transformation matrix between the frame $\mathcal{C}_k$ and the frame of the center of mass, that is

$${}^{c}X^{*}_{\mathcal{C}_k} := \begin{bmatrix} 1_3 & 0_{3\times3} \\ S(p_{\mathcal{C}_k} - p_G) & 1_3 \end{bmatrix}.$$

Note that $f_k = \begin{bmatrix} F_k^\top & \mu_k^\top \end{bmatrix}^\top$, i.e. it is composed of a pure force and a pure torque at the contact. The right hand side of equation (6.2) thus describes all the external forces acting on the system. In the following, differently from (2.14) where all the external forces were considered, we define n_c to be the number of controlled wrenches. All the other possible external wrenches are seen as disturbances which must be compensated for by the controller.

The control objective is the asymptotic stabilization of a desired momenta dynamics (6.2). We start by grouping the wrenches in one variable, stacking them one on top of the others, i.e.

$$f := \begin{bmatrix} f_1 \\ \vdots \\ f_{n_c} \end{bmatrix} \in \mathbb{R}^{6n_c}. \tag{6.3}$$

By defining

$${}^{c}X^{*}_{f} := \begin{bmatrix} {}^{c}X^{*}_{\mathcal{C}_1} & \cdots & {}^{c}X^{*}_{\mathcal{C}_{n_c}} \end{bmatrix} \in \mathbb{R}^{6\times6n_c}$$

we can rewrite (6.2) as

$$\dot{H} = {}^{c}X_{f}^{*} f + mg. \tag{6.4}$$

Given a reference momenta H^{r}, we can define the momenta error as

$$\tilde{H} := H - H^{r}.$$

As a consequence a common choice for the desired momenta is defined in the following expression:

$$\dot{H}^{d} := \dot{H}^{r} - K_{p}^{h}\tilde{H} - K_{i}^{h}\int_{0}^{t}\tilde{H}\,ds \tag{6.5}$$

where the matrices K_{p}^{h} and $K_{i}^{h} \in \mathbb{R}^{6\times 6}$ should be symmetric and positive definite. It is important to note that classical choices for K_{i}^{h} in Eq. (6.5) are given by

$$K_{i}^{h} := \begin{bmatrix} K_{i}^{h,L} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} \end{bmatrix} \tag{6.6}$$

i.e. the integral correction term at the angular momentum level is equal to zero. $K_{i}^{h,L} \in \mathbb{R}^{3\times 3}$ is a symmetric positive definite matrix.

The above choice for K_{i}^{h} practically results in the following equivalent choice for H^{d}

$$\dot{H}^{d} := \begin{bmatrix} m\ddot{x}_{g}^{r} - K_{i}^{h,L} m(x_{g} - x_{g}^{r}) - K_{p}^{h,L} m(\dot{x}_{g} - \dot{x}_{g}^{r}) \\ -K_{p}^{h,\omega}(L_{G} - L_{G}^{d}) \end{bmatrix}.$$

This means that the linear momentum is controlled in order to follow a desired reference trajectory for the center of mass. Furthermore, the robot angular moment is, in general, controlled to zero, i.e. $L_{G}^{d} \equiv 0$. Some remarks about the control of the angular momentum are done in the conclusions of this chapter.

If we consider as control variable the interaction forces f, note that (6.4) possesses a different number of solutions depending on the number of contacts. Indeed, the number of rows of the matrix ${}^{c}X_{f}^{*}$ is fixed to 6, while the number of columns is a multiple of 6. Note also that by construction the matrix ${}^{c}X_{f}^{*}$ is always full-row rank. We thus have two different cases depending on the number of contacts:

1. $n_{c} = 1$. The matrix ${}^{c}X_{f}^{*}$ is invertible, and only one solution exists.

2. $n_{c} > 1$. ${}^{c}X_{f}^{*}$ is wide and thus (6.4) admits infinite solutions.

The optimization problem associated with the first task — the control of the robot momenta — is then the following:

$$\begin{aligned} \underset{f}{\text{minimize}} \; & \left\| \dot{H}^d - \dot{H} \right\|^2 \\ \text{subject to } & \dot{H} = Xf + mg \\ & \text{friction constraints} \\ & \text{center of pressure constraints} \\ & \text{positivity of the normal forces.} \end{aligned} \tag{6.7}$$

The contact forces must then satisfy the hard constraints consisting in their physical feasibility. Note that the solution is in the least squares sense in case $n_c = 1$.

6.1.2 SECOND TASK: STABILITY OF THE ZERO DYNAMICS

The second task in our hierarchy consists in the so called *postural task*, which is responsible of stabilizing the zero dynamics.

Consider the dynamical system (2.14). We can compactly write the contribution due to all the external forces by stacking the wrenches and the Jacobians. The force variable $f \in \mathbb{R}^{6n_c}$ has already been defined in Eq. (6.3). We define now the Jacobian J by stacking all the corresponding Jacobians, i.e.

$$J := \begin{bmatrix} J_{\mathcal{C}_1} \\ \vdots \\ J_{\mathcal{C}_{n_c}} \end{bmatrix} \in \mathbb{R}^{6n_c \times n+6}, \quad \begin{bmatrix} J_b & J_j \end{bmatrix} = \begin{bmatrix} J^b_{\mathcal{C}_1} & J^j_{\mathcal{C}_1} \\ \vdots & \\ J^b_{\mathcal{C}_{n_c}} & J^j_{\mathcal{C}_{n_c}} \end{bmatrix}. \tag{6.8}$$

The result of the postural task are the torques to be applied to the constrained mechanical system (2.14). Kinematic constraints due to rigid contacts are represented by imposing that the velocity of the frame associated with the k-th contact, $\mathcal{C}_k$, is zero, i.e. by Eq. (2.18). The constraints are then differentiated once as in (2.19) and stacked together. The minimization problem can be formulated as

$$\begin{aligned} \underset{\tau}{\text{minimize}} \; & \|\tau - \varphi\|^2 \\ \text{subject to } & M(q)\dot{\nu} + C(q,\nu)\nu + g(q) = B\tau + J^\top f \\ & J(q)\dot{\nu} + \dot{J}(q)\nu = 0 \\ & f = f^* \\ & \varphi = \text{to be chosen} \end{aligned} \tag{6.9}$$

where and f^* is the contact wrench which minimizes (6.7).

We still have to choose the variable φ. Nonetheless, note that the last constraint, which is the postural task, is satisfied in the

least squares sense, i.e. after the dynamics, contacts and the task hierarchy constraints are satisfied.

The choice of the variable φ is usually performed as if the task is the only task in the hierarchy. Consider the system (2.14) and let's separate the base and joints space dynamics, i.e.

$$\begin{cases} M_b \dot{\nu}_b + M_{bj} \ddot{q}_j + h_b - J_b^\top f = 0 \\ M_{bj}^\top \dot{\nu}_b + M_j \ddot{q}_j + h_j - J_j^\top f = \tau \end{cases}$$

where $\nu_b \in \mathbb{R}^6$ is the linear and angular velocity of the base frame $\mathcal{B}$. We can obtain the expression of $\dot{\nu}_b$ from the first equation:

$$\dot{\nu}_b = -M_b^{-1} \left[M_{bj} \ddot{q}_j + h_b - J_b^\top f \right]$$

and plug it into the second equation thus obtaining the joint space dynamics:

$$\begin{aligned} &(M_j - M_{bj}^\top M_b^{-1} M_{bj}) \ddot{q}_j + (h_j - M_{bj}^\top M_b^{-1} h_b) \\ &- (J_j^\top - M_{bj}^\top M_b^{-1} J_b^\top) f = \tau \, . \end{aligned}$$

By defining

$$\begin{aligned} \widehat{M}_j(q) &:= (M_j(q) - M_{bj}^\top(q) M_b^{-1}(q) M_{bj}(q)) \\ \widehat{h}_j(q, \nu) &:= (h_j(q, \nu) - M_{bj}^\top(q) M_b^{-1}(q) h_b(q, \nu)) \\ \widehat{J}_j(q) &:= (J_j(q) - J_b(q) M_b^{-1}(q) M_{bj}(q)) \end{aligned}$$

we can rewrite the joint space dynamics more compactly as

$$\widehat{M}_j \ddot{q}_j + \widehat{h}_j - \widehat{J}_j^\top f = \tau \, . \tag{6.10}$$

The control objective of the postural task is to impose a desired trajectory $q_j^r(t)$ on the joint space dynamics. The dynamics in Eq. (6.10) is now fully actuated and we can apply classical control methods for fully actuated manipulators. In particular we present here two common alternatives for the choice of the variable φ.

6.1.2.1 *Computed Torque approach*

One of the most diffuse methods in model-based control of manipulators is the *Computed Torque* approach. This method is based on the separation of the control architecture in two nested loop. The inner one is responsible of compensating the nonlinear terms of the dynamics, while the outer one, usually a Proportional-Integral-Derivative (PID) action, handles the tracking of the desired reference.

Given Eq. (6.10) we can easily design the inner loop. In fact the choice of

$$\tau = \widehat{h}_j - \widehat{J}_j^\top f + \widehat{M}_j u$$

with $u \in \mathbb{R}^n$ the new control input, when applied to the system (6.10) results in the following linear dynamical system:

$$\ddot{q}_j = u \, .$$

u can now be chosen to track the reference trajectory, for example by implementing a PID action, i.e.

$$u = \ddot{q}_j^r - K_d^q(\dot{q}_j - \dot{q}_j^r) - K_p^q(q_j - q_j^r) - K_i^q \int_{t_0}^{t} q_j - q_j^r \, ds$$

where K_p^q, K_d^q and $K_i^q \in \mathbb{R}^{n\times n}$ are suitably chosen positive definite gain matrices.

One drawback of the computed torque approach is that the dynamical model of the nonlinear system must be known with great precision. Indeed, the stability of the closed loop system strongly depends on the fact that the inner control loop compensate for all the nonlinear affects. If this hypothesis were not satisfied, stability would be no longer guaranteed.

6.1.2.2 *Proportional-Derivative plus Compensation approach*

The fact that model knowledge strongly affects the performances and the stability of the computed torque approach, as highlighted above, has suggested the development of *Computed-Torque-like* methodologies. In these control methods, some of the dynamic information provided by the model are discarded. Of course, performances are degraded w.r.t. the full computed torque method, but less hypothesis are needed to guarantee stability.

One of the simplest yet used control architecture in this family is the *PD plus gravity compensation*. As the name suggests, the control action is simply given by:

$$\tau = \widehat{g}_j - K_d^q \dot{q}_j - K_p^q(q_j - q_j^r)$$

where K_p^q and $K_d^q \in \mathbb{R}^{n\times n}$ are suitably chosen positive definite gain matrices and where we defined

$$\widehat{g}_j(q) := \widehat{h}_j(q, 0)$$

the gravity compensation term.

In our control architecture we also compensate for the contact forces, thus applying the following control torques:

$$\tau = \widehat{g}_j - \widehat{J}_j^\top f - K_d^q \dot{q}_j - K_p^q(q_j - q_j^r) \, . \tag{6.11}$$

The power of this method resides in the fact that asymptotic stability for an equilibrium configuration $(q_j^r, 0)$, which can be proved by means of Lyapunov analysis, does not necessitate of information on the mass and Coriolis matrix.

In the optimization problem (6.9) we choose φ to resemble a PD plus gravity and forces compensation control, i.e.

$$\varphi = \widehat{g}_j - \widehat{J}_j^\top f - K_d^q \dot{q}_j - K_p^q (q_j - q_j^r) \tag{6.12}$$

but any other choices, such as the computed torque control law, is applicable.

6.1.2.3 *Exploiting the postural task for cartesian motion tasks*

In literature motion tasks are usually specified as separate layers in the hierarchy, at a higher priority of the postural task. However, we can notice that the postural task formulation leads to a joint space tracking task, that is, the postural task is responsible to track a desired joint trajectory reference signal.

We can thus exploit this feature to avoid the introduction of explicit cartesian tasks. Position references in cartesian space are mapped to desired joint positions by inverting the kinematic relationship:

$$x_{t_j} = h_{t_j}(q)$$

with $x_{t_j} \in \mathbb{R}^3 \times SO(3)$ and $h_{t_j} : \mathbb{R}^3 \times SO(3) \times \mathbb{R}^n \to \mathbb{R}^3 \times SO(3)$. The same procedure can be done for velocity or acceleration references in cartesian space. All the tasks are then merged together and any conflict it may occur is solved beforehand. As output of this process we obtain position, velocity and acceleration references of the joint variables, i.e. q_j^r, $\dot{q}_j^r$ and $\ddot{q}_j^r$, to be supplied as inputs to the postural task.

6.1.3 PUTTING EVERYTHING TOGETHER: THE HIERARCHICAL CONTROL

Solutions to the hierarchy composed of the previously described two tasks can be solved in different ways. In some sense this is what differentiate the various works presented in Section 1.2.2.

Examine now the dynamic equation (2.14)

$$M(q)\dot{\nu} + C(q,\nu)\nu + g(q) = B\tau + \sum_{k=1}^{n_c} J_{c_k}^\top f_k$$

and the constraints equations (2.19)

$$J_{\mathcal{C}_k}(q)\dot{\nu} + \dot{J}_{\mathcal{C}_k}(q)\nu = 0.$$

If we substitute the expression of the accelerations as obtained from (2.14), i.e.

$$\dot{\nu} = M^{-1}(B\tau + J^\top f - C\nu - g)$$

into (2.19) we obtain the following equation:

$$\begin{aligned} JM^{-1}\left(B\tau + J^\top f - C\nu - g\right) + \dot{J}\nu = 0, \\ (JM^{-1}B)\tau = JM^{-1}(C\nu + g - J^\top f) - \dot{J}\nu. \end{aligned} \tag{6.13}$$

Equation (6.13) specifies the instantaneous relation between the contact forces and the joint torques.

The function $\tau = \tau(f)$ in (6.13) states that given a choice of the joint torques τ we can uniquely determine the contact forces f. On the contrary, a choice of contact forces f can be generated by an infinite choice of joint torques, depending on the number of contacts and the number of degrees of freedom of the system. Indeed, if we examine the matrix

$$JM^{-1}B \in \mathbb{R}^{6n_c \times n}$$

and we assume that J is full row rank, we have two possible cases:

i) $6n_c \geqslant n$. There are not enough degrees of freedom to satisfy the constraints, or they perfectly match. Either the solution is unique or is defined in the least square sense. In both cases there is no redundancy.

ii) $6n_c < n$. The mechanical system has more degrees of freedom than the constraints. There exists an infinity of choices for the joint torques to generate the desired contact forces.

Usually for a humanoid robot the second case is the most common.

We can now consider the solution of the first task, i.e. the control of the robot momenta. We have two different approaches depending on the number of contacts.

6.1.3.1 *Single Contact case*

If $n_c = 1$, i.e. there is only one contact, we solve directly (6.7) by means of a QP solver. We thus obtain a value for f^* which we can use directly into (6.13):

$$(JM^{-1}B)\tau = JM^{-1}(C\nu + g - J^\top f^*) - \dot{J}\nu. \tag{6.14}$$

Solving for τ we obtain the following expression

$$\tau = (JM^{-1}B)^{\dagger}JM^{-1}(Cv + g - J^{\top}f^{*}) - \dot{J}v + N_{\tau}\varphi \tag{6.15}$$

where $(JM^{-1}B)^{\dagger}$ is the Moore Penrose pseudoinverse of $(JM^{-1}B)$ and $N_{\tau} \in \mathbb{R}^{n\times n}$ is the projector onto the null space of $(JM^{-1}B)$ if it exists. Finally $\varphi \in \mathbb{R}^{n}$ is the free variable which we use to satisfy the second task, i.e. we choose the PD plus gravity and forces compensation as in Eq. (6.12).

In our approach we do not enforce torque or joint limits. If that were the case, we can solve (6.14) together with the additional constraints by means of a QP solver.

6.1.3.2 *Multiple Contacts case*

If $n_c > 1$, then the constraint (6.4) admits infinite solutions. Before imposing the inequality constraints, we parametrize the solution as

$$\begin{aligned} f^{*} &= {}^{c}X_{f}^{*\dagger}(\dot{H}^{d} - mg) + N_{f}\gamma \\ &= f_{1} + N_{f}\gamma. \end{aligned} \tag{6.16}$$

where ${}^{c}X_{f}^{*\dagger}$ is the Moore-Penrose pseudoinverse of ${}^{c}X_{f}^{*}$, N_f is a projector onto the null space of ${}^{c}X_{f}^{*}$ and $\gamma \in \mathbb{R}^{6n_c}$ is the associated free variable. Note that

$$f_1 = {}^{c}X_{f}^{*\dagger}(\dot{H}^{d} - mg)$$

is the minimum norm solution. It is quite common in literature to force the minimum norm of the contact forces. If this is the case problem (6.7) can be solved directly enforcing the inequality constraints. Equation (6.14) then follows directly. As we motivate in the next chapter, we perform a different choice for the contact forces.

Consider the expression of φ defined in Eq. (6.12) and, for notation simplicity, rewrite it as

$$\varphi = \widehat{g}_j - \widehat{J}_j^{\top} f + \overline{\varphi}$$

where $\overline{\varphi} = -K_d^q \dot{q}_j - K_p^q(q_j - q_j^r)$. We can now solve (6.13) for τ:

$$\begin{aligned} (JM^{-1}B)\tau &= JM^{-1}(Cv + g - J^{\top}f) - \dot{J}v \\ \tau &= (JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g - J^{\top}f) - \dot{J}v\right] + N_{\tau}\varphi \\ &= (JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g - J^{\top}f) - \dot{J}v\right] \\ &\quad + N_{\tau}(\widehat{g}_j - \widehat{J}_j^{\top} f + \overline{\varphi}) \end{aligned} \tag{6.17}$$

If we group the force terms we obtain the following expression:

$$\begin{aligned}\tau = &-\left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right] f \\ &+ (JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g) - \dot{J}v\right] + N_{\tau}(\widehat{g}_j + \overline{\varphi})\end{aligned}$$

We now substitute the expression of the force from (6.16) into the above equation, obtaining:

$$\begin{aligned}\tau &= -\left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right](f_1 + N_f\gamma) \\ &\quad + (JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g) - \dot{J}v\right] + N_{\tau}(\widehat{g}_j + \overline{\varphi}) \\ &= -\left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right] N_f\gamma \\ &\quad + (JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g) - \dot{J}v\right] + N_{\tau}(\widehat{g}_j + \overline{\varphi}) \\ &\quad - \left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right] f_1 \\ &= C\gamma + d\end{aligned} \tag{6.18}$$

where we defined

$$C := -\left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right] N_f$$

and

$$\begin{aligned}d := &(JM^{-1}B)^{\dagger}\left[JM^{-1}(Cv + g) - \dot{J}v\right] + N_{\tau}(\widehat{g}_j + \overline{\varphi}) \\ &- \left[(JM^{-1}B)^{\dagger}JM^{-1}J^{\top} + N_{\tau}\widehat{J}_j^{\top}\right] f_1\end{aligned}$$

We can now formulate the following optimization problem

$$\begin{aligned}&\underset{\gamma}{\text{minimize}} \left\|\tau(\gamma)\right\|^2 \\ &\text{subject to } \tau(\gamma) = C\gamma + d \\ &\qquad\qquad \text{friction constraints} \\ &\qquad\qquad \text{center of pressure constraints} \\ &\qquad\qquad \text{positivity of the normal forces.}\end{aligned} \tag{6.19}$$

Note that the inequality constraints must be reformulated on the new optimization variable γ as defined in (6.16). Starting from the friction cone constraints in (2.22), the original inequality constraint

$$A^{f_c} f < 0_{4n_c}$$

can be rewritten as

$$A^{f_c} N_f \gamma + A^{f_c} f_1 < 0_{4n_c}. \tag{6.20}$$

The torsional friction constraints in (2.23), i.e.

$$A^{\tau} f < 0_{2n_c}$$

becomes

$$A^{\tau} N_f \gamma + A^{\tau} f_1 < 0_{2n_c}. \tag{6.21}$$

The center of pressure constraints in (2.20), i.e.

$$A^{cop} f < 0_{4n_c}$$

can be transformed into

$$A^{cop} N_f \gamma + A^{cop} f_1 < 0_{4n_c}. \tag{6.22}$$

Finally, the positivity of the normal force constraints in (2.24)

$$A^{p} f < 0_{n_c}$$

is now transformed into

$$A^{p} N_f \gamma + A^{p} f_1 < 0_{n_c}. \tag{6.23}$$

The optimization problem (6.19) can now be stated as

$$\begin{aligned} &\underset{\gamma}{\text{minimize}} \ \left\| \tau(\gamma) \right\|^2 \\ &\text{subject to } \tau(\gamma) = C\gamma + d \\ &\qquad\qquad (6.20),\ (6.21) \\ &\qquad\qquad (6.22) \\ &\qquad\qquad (6.23) \end{aligned} \tag{6.24}$$

By plugging γ^*, defined as the value of γ which minimizes (6.24), into (6.18) we obtain the function

$$\tau^* = \tau(\gamma^*)$$

i.e. the final expression for the torques to be applied to the robot.

6.2 OBSERVATIONS

We conclude this chapter by presenting some observations on the control architecture presented. Note that these observations apply also to the state-of-the-art controllers described in the opening of the chapter. Addressing these issues will be subject of future works.

6.2.1 CONTROL OF THE ANGULAR MOMENTUM

Control of the angular momentum has been motivated by observations on humans. Nevertheless recent studies [31, 106] suggest that angular momentum is not regulated to zero, but is controlled depending on the tasks to be achieved or the walking gait.

From a control perspective, the regulation of the angular momentum to zero, or its control to a predefined, task-dependent value, does not imply that a particular pose, e.g. the initial one, is achieved. This can result in the robot keeping an "unusual" pose, just because of the regulation of the angular momentum. To overcome this issue some authors [76] close the loop on a specific link of the robot, usually the torso.

6.2.2 STABILITY OF THE ZERO DYNAMICS

The zero dynamics (see [38] and [37, Ch.6]) is the dynamical system that characterizes the internal behaviours of a system once initial conditions and inputs are chosen in such a way as to constrain the output to be identically zero. As far as we know, there not exist in literature a stability analysis of the zero dynamics for momentum-based controllers for free-floating mechanical systems. Chapter 8 shows numerical evidence that momentum-based controllers may lead to unstable zero dynamics. We then propose a modification to the control scheme that ensure asymptotic stability.

7 EXPLOITING THE FORCE REDUNDANCY: TORQUE MINIMISATION AS OPTIMALITY CRITERION

In the previous chapter we showed that if the number of contacts is greater than one, there exists an infinite choice of the contact forces which satisfy the robot momenta equations (6.4). Differently from most of the approaches found in literature where the minimum force norm solution is imposed, we choose to exploit the redundancy of the contact forces such as the internal joint torques are minimized.

This chapter first defines the concept of *sensitivity* of the center of pressure at a static configuration. We then introduce a useful change of basis which simplifies the analysis on the full dynamical model. Analytical results are thus obtained on a simpler model, i.e. a planar model of a four-bar linkage. We conclude the chapter by presenting experiments on the iCub humanoid robot which validate the results on the full dynamical robot.

7.1 SENSITIVITY OF THE STATIC CENTER OF PRESSURE

This section discusses a property of any center of pressure associated with rigid, flat contacts and quasi-static robot configurations. We shall see in Section 7.3 that this property – called *static center of pressure sensitivity* – can be used as an element of evaluation of balancing controllers for humanoids.

7.1.1 THE SENSITIVITY OF THE STATIC CENTER OF PRESSURE IN A CASE STUDY: THE FOUR-BAR LINKAGE CASE

To provide the reader with an introduction to the *sensitivity* of the static center of pressure, let us focus on Figure 7.1. This picture depicts a four-bar linkage standing on two flat, rigid contacts. In this configuration, the mechanical system possesses

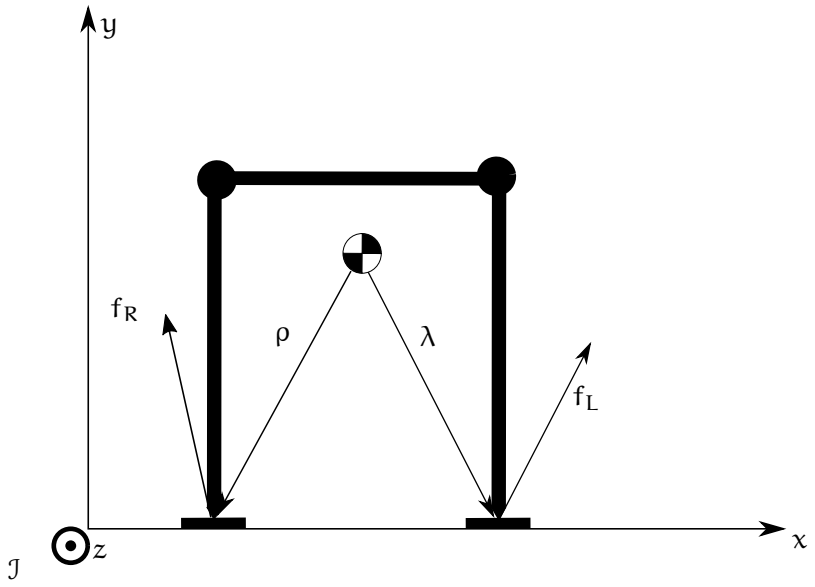


Figure 7.1: Planar four-bar linkage exploiting two rigid contact to stand

one degree of freedom, i.e. the variable ξ, which represents the minimal coordinate of the constrained mechanical system. At the equilibrium configuration, the system is *hyperstatic* because the equilibrium conditions of the Newton-Euler equations are not sufficient to determine all the internal forces – including the joint torques τ. Since the joint torques are assumed to be a control input, however, one can assume that their value is given, with the only requirement that the system remains at the equilibrium configuration.

Assume that the joint torques depend only on the minimal coordinate ξ. Then, the center of pressures at each *static* robot configuration, i.e. $\dot{\xi} = 0$, depend only on the minimal coordinate ξ. This poses the following interesting question.

What is the rate-of-change of the center of pressures over the quasi-static constrained robot motion?

Different control laws for making the four-bar linkage balance may be associated with different rate-of-changes of the static center of pressure over the constrained robot motion. Clearly, the higher this rate-of-change, the higher the likelihood to hit the boundaries of the support polygon of the foot, which causes breaking the associated contact and, eventually, a fall of the robot. The above rate-of-change of the center of pressure is what we call *static center-of-pressure sensitivity*, and the following generalizes this concept to any multi-body system constrained by some rigid, flat contacts.

7.1.2 THE FORMAL DEFINITION

Assume that the robot makes n_c rigid contacts with the environment, and that no other external wrench acts on the robot. Let consider the Jacobian J obtained by stacking all contact Jacobians $J_{\mathcal{C}_k}$ as in (6.8).

Consequently, all rigid contacts imply the constraint (2.18) on the floating base system (2.14). As we did in Section 6.1.2 the associated constraints can be differentiated and written as:

$$\dot{J}\nu + J\dot{\nu} = 0. \tag{7.1}$$

As we did in Section 6.1.2 we combine the above constraints with Eq. (2.14). By defining

$$f := \begin{bmatrix} f_1 \\ \vdots \\ f_{n_c} \end{bmatrix}$$

we obtain the same expression in (6.13), which in turn defines the explicit expression of all contact forces, i.e.

$$f = (JM^{-1}J^\top)^{-1}\left[JM^{-1}\left(C(q,\nu)\nu + G(q) - B\tau\right) - \dot{J}\nu\right]. \tag{7.2}$$

The hidden assumption for the above equality to hold is that J is full row rank. Once the obtained expression of f is substituted into Eq. (2.14), one obtains the following equations of motion:

$$M(q)\dot{\nu} + J^\top(JM^{-1}J^\top)^{-1}\dot{J}\nu = N\left[B\tau - C(q,\nu)\nu - G(q)\right] \tag{7.2a}$$

$$N = 1_{n+6} - J^\top(JM^{-1}J^\top)^{-1}JM^{-1}. \tag{7.2b}$$

Evaluating Eq. (7.2a) at the equilibrium condition

$$(\nu, \dot{\nu}) = (0, 0)$$

yields

$$N\left[B\tau - G(q)\right] = 0. \tag{7.3}$$

Eq. (7.3) defines the values that the joint torques must take to satisfy the equilibrium condition $(\nu, \dot{\nu}) = (0, 0)$. Then, we make the following assumption.

Assumption 7.1. *The n_c holonomic constraints (2.17), restrain the configuration space $\mathbb{Q}$ into the set $\mathbb{R}^{n+6-6n_c}$, i.e. there exists a mapping $\chi : \mathbb{R}^{n+6-6n_c} \to \mathbb{Q}$ such that*

$$\forall q \in \mathbb{Q} \quad \exists! \quad \xi \in \mathbb{R}^{n+6-6n_c} \quad : \quad q = \chi(\xi). \tag{7.4}$$

Furthermore, the joint torques τ at the equilibrium configuration $(\nu, \dot{\nu}) = (0, 0)$ depend only on the robot constrained configuration space, i.e. $\tau = \tau(\xi)$ with $\xi \in \mathbb{R}^{n+6-6n_c}$. □

In view of the above assumption, the contact wrenches at the equilibrium configuration, i.e.

$$f^e = \begin{pmatrix} f_1^e \\ \vdots \\ f_{n_c}^e \end{pmatrix} = (JM^{-1}J^\top)^{-1}JM^{-1}(G - B\tau), \tag{7.5}$$

depend only on the robot constrained motion, i.e. $f^e = f^e(\xi)$. In light of the above, we can now define the *static center-of-pressure sensitivity* as follows.

Definition 7.1. *Let* $\sigma_k \in \mathbb{R}^2$ *denote the center of pressure associated with the* k*-th rigid contact at the equilibrium* $(\nu, \dot{\nu}) = (0, 0)$*, i.e.*

$$\sigma_k = \frac{1}{e_3^\top f_k^e} \begin{pmatrix} -e_5^\top f_k^e \\ e_4^\top f_k^e \end{pmatrix}. \tag{7.6}$$

The static center of pressure sensitivity η_k *is defined as the rate-of-change of* (7.6) *over the constrained robot motion, i.e.*

$$\eta_k := \frac{\partial \sigma_k}{\partial \xi}. \tag{7.7}$$

□

In general, the *static center-of-pressure sensitivity* is a matrix belonging to $\mathbb{R}^{2\times n+6-6n_c}$, and characterizes how fast the center of pressure associated to a rigid, planar contact moves depending on the constrained robot motion. This sensitivity may then characterize the stability of a contact associated with a robot configuration. Of particular importance are the cases when the sensitivity η_k tends to infinity. Clearly, these configurations must be avoided by any controller associated with the robot, since they may cause abrupt system responses and, eventually, a robot fall. Furthermore, the proposed sensitivity may be used to compare different balancing controllers, since lower values of η_k are associated with smaller variations of the center of pressures, which may increase the likelihood of not breaking the contact.

7.2 DECOUPLING THE JOINT AND BASE FRAME ACCELERATIONS

This section presents a new expression of the equations of motion (2.14). In particular, the next lemma presents a change of coordinates in the state space (q, ν) that transforms the system dynamics (2.14) into a new form where the mass matrix is block diagonal, thus decoupling joint and base frame accelerations.

Lemma 7.1. *Consider the equations of motion given by (2.14). Perform the following change of state variables:*

$$q := q, \quad \bar{\nu} := T(q)\nu, \tag{7.8}$$

with

$$T := \begin{bmatrix} {}^{c}X_b & T_s \\ 0_{n\times 6} & 1_n \end{bmatrix}, \tag{7.9a}$$

$$T_s := {}^{c}X_b M_b^{-1} M_{bj} \tag{7.9b}$$

$${}^{c}X_b := \begin{bmatrix} 1_3 & -S({}^{\mathcal{I}}p_G - {}^{\mathcal{I}}p_{\mathcal{B}}) \\ 0_{3\times 3} & 1_3 \end{bmatrix}. \tag{7.9c}$$

Then, the equations of motion with state variables (q, ν) can be written in the following form

$$\bar{M}(q)\dot{\bar{\nu}} + \bar{C}(q,\bar{\nu})\bar{\nu} + \bar{g} = B\tau + \sum_{k=1}^{n_c} \bar{J}_{\mathcal{C}_k}^{\top} f_k \tag{7.10}$$

with

$$\bar{M}(q) = T^{-\top} M T^{-1} = \begin{bmatrix} \bar{M}_b(q) & 0_{6\times n} \\ 0_{n\times 6} & \bar{M}_j(q_j) \end{bmatrix}, \tag{7.11a}$$

$$\begin{aligned} \bar{C}(q,\bar{\nu}) &= T^{-\top}(M\dot{T}^{-1} + CT^{-1}) \\ &= \begin{bmatrix} 0_{3\times 3} & 0_{3\times 3} & 0_{3\times n} \\ 0_{3\times 3} & \bar{C}_b(q,\nu) & \bar{C}_{bj}(q,\nu) \\ 0_{3\times 3} & -\bar{C}_{bj}^{\top}(q,\nu) & \bar{C}_j(q,\nu) \end{bmatrix}, \end{aligned} \tag{7.11b}$$

$$\bar{g} = T^{-\top} g = mge_3, \tag{7.11c}$$

$$\bar{J}_{\mathcal{C}_k}(q) = J_{\mathcal{C}_k}(q)T^{-1} = \begin{bmatrix} \bar{J}_{\mathcal{C}_k}^{b}(q) & \bar{J}_{\mathcal{C}_k}^{j}(q_j) \end{bmatrix}.$$

We can further decompose $\bar{M}_b(q)$ as

$$\bar{M}_b(q) = \begin{bmatrix} m1_3 & 0_{3\times 3} \\ 0_{3\times 3} & I(q) \end{bmatrix}$$

and $\bar{J}_{\mathcal{C}_k}^{b}(q)$ as

$$\bar{J}_{\mathcal{C}_k}^{b}(q) = \begin{bmatrix} 1_3 & -S({}^{\mathcal{I}}p_{\mathcal{C}_k} - {}^{\mathcal{I}}p_G) \\ 0_{3\times 3} & 1_3 \end{bmatrix}.$$

The above lemma points out that the mass matrix of the transformed system (7.10) is block diagonal, i.e. the transformed

base acceleration is independent from the joint acceleration. More precisely, the transformed robot's velocity $\bar{\nu}$ is given by

$$\bar{\nu} = \begin{pmatrix} {}^{\mathcal{I}}\dot{p}_G \\ {}^{\mathcal{I}}\omega_G \\ \dot{q}_j \end{pmatrix} \tag{7.12}$$

where ${}^{\mathcal{I}}\dot{p}_G$ is the velocity of the center-of-mass of the robot, and ${}^{\mathcal{I}}\omega_G$ is the so-called *average angular velocity*[1] [40, 20, 75]. Observe also that the Coriolis term $\bar{C}(q, \bar{\nu})$ is *quasi* block diagonal, and that the gravity term $\bar{g}$ is constant and influences the acceleration of the center-of-mass only.

A transformation similar to (7.8) was presented in [25]. The transformation presented here, however, is from the state (q, ν) into a mixed state, which is composed of the robot's momentum and velocity. Thus it does not represent classical transformations of the state space.

7.3 FOUR-BAR LINKAGE MODEL

This section studies the effects of minimizing joint torques of a planar four-bar linkage when standing on two, flat, rigid contacts.

As the experimental results, later in this chapter, show, the chosen case study is representative of a humanoid robot standing on two feet. More precisely, by using the contributions in Sections 7.1 and 7.2, what follows shows that minimizing the joint torques when the Newton-Euler equations are at the equilibrium can bring benefits in terms of the *static center-of-pressure sensitivity*, thus enforcing the stability of the contact.

7.3.1 MODELLING

Consider the four-bar linkage shown in Figure 7.2, where l and d denote the lengths of the leg and of the upper rod, respectively. The inertial frame $\mathcal{I}$ is chosen so as the y axis opposes gravity, the z axis exits from the plane, and the x axis completes the right-hand base. Being a planar model, only translations in the $x - y$ plane and rotations about the z axis are allowed. Let ${}^{\mathcal{I}}p_b \in \mathbb{R}^2$ denote the position of the origin of the frame $\mathcal{B}$ – which is located at the center of the upper rod– expressed in the

1 The term ${}^{\mathcal{I}}\omega_G$ is also known as the *locked angular velocity* [60].

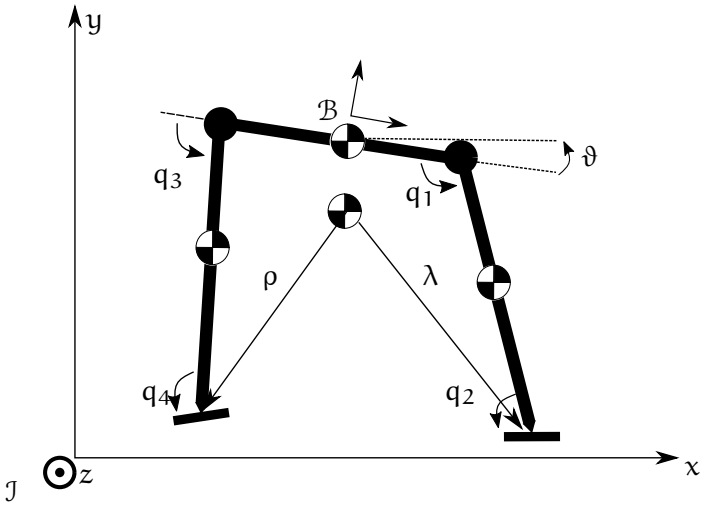


Figure 7.2: Free-floating planar four-bar linkage

inertial frame. The configuration space is then parametrized by the following coordinates

$$q = ({}^{\mathcal{J}}p_b, \theta, q_1, q_2, q_3, q_4) \in \mathbb{R}^7.$$

To derive the equations of motion, we use the Lagrange formalism with the assumption that any link composing the four-bar linkage can be approximated as a point-mass located at the center of the link. In addition, we also assume that the mass of the left leg equals the mass of the right leg.

We present here only the terms of the dynamical model that are necessary to the remainder of the section. In particular, partition the mass matrix M as follows

$$M = \begin{bmatrix} M_b & M_{bj} \\ M_{bj}^\top & M_j \end{bmatrix}.$$

Let $\sin(q_i) = s_i$ and $\cos(q_i) = c_i$. One can verify that

$$M_{bj} = \frac{m_l}{2} \begin{bmatrix} ls_1 & 0 & ls_3 & 0 \\ -lc_1 & 0 & -lc_3 & 0 \\ \frac{l^2 s_1^2 - lc_1(d-lc1)}{2} & 0 & \frac{l^2 s_3^2 + lc_3(d+lc_3)}{2} & 0 \end{bmatrix} \tag{7.13}$$

where m_l is the mass of either leg.

To represent a classical balancing context of humanoid robots, we assume that the mechanism makes contact with the environment at the two extremities. Also, we assume that the distance between the contacts equals the length d of the upper

rod – see Figure 7.1. Then, the rigid constraints acting on the system takes the form (7.1), with

$$J = \begin{bmatrix} J_{\mathcal{C}_1} \\ J_{\mathcal{C}_2} \end{bmatrix} \in \mathbb{R}^{6\times 7},$$

$J_{\mathcal{C}_1}$ and $J_{\mathcal{C}_2}$ the Jacobians of the two extremities, and $\nu = \dot{q}$. The associated contact wrenches are

$$f = \begin{bmatrix} f_L \\ f_R \end{bmatrix} \in \mathbb{R}^6.$$

As a consequence of these constraints, the mechanism possesses one degree of freedom when standing on the extremities. With the aim of evaluating the sensitivity of the static center-of-pressure, observe that the mapping $\chi(\xi)$ in (7.4) is given by:

$$\begin{pmatrix} {}^{\mathcal{J}}p_b \\ \theta \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} = \begin{pmatrix} \frac{1}{2}de_1 + l\sin(\xi)e_2 + l\cos(\xi)e_1 \\ 0 \\ \xi \\ \pi - \xi \\ \xi \\ \pi - \xi \end{pmatrix}. \tag{7.14}$$

7.3.2 CHOICE OF THE CONTACT FORCES

Remark that in order to evaluate the sensitivity of the static center of pressure, we need an expression for the contact forces at the equilibrium configuration – see Definition 7.1. What follows proposes an analysis of the sensitivity of the static center of pressures evaluated with contact wrenches obtained from two criteria.

1. The contact wrenches ensure the equilibrium of the centroidal dynamics written in the plane – i.e. three equations. The three dimensional redundancy of the contact wrenches minimize the norm of the contact wrenches $\|f\|$.
2. The contact wrenches ensure the equilibrium of the centroidal dynamics. The three dimensional redundancy of the contact wrenches minimize the norm of the joint torques $\|\tau\|$.

More precisely, being the balance of the external wrenches acting on the mechanism, the Newton-Euler equations (6.2) implies the following:

$$0 = {}^{c}X_L^{*\,L}f_L + {}^{c}X_R^{*\,R}f_R - mge_2 = {}^{c}X_f^{*}f - mge_2 \tag{7.15}$$

where mge_2 is the wrench due to gravity and ${}^cX_f^* := \begin{bmatrix} {}^cX_L^* & {}^cX_R^* \end{bmatrix}$,

$$ {}^cX_L^* = \begin{bmatrix} 1_2 & 0_{2\times 1} \\ (S\lambda)^\top & 1 \end{bmatrix}, {}^cX_R^* = \begin{bmatrix} 1_2 & 0_{2\times 1} \\ (S\rho)^\top & 1 \end{bmatrix}, $$

$\lambda = \begin{bmatrix} \lambda_1 & \lambda_2 \end{bmatrix}^\top \in \mathbb{R}^2$ and $\rho = \begin{bmatrix} \rho_1 & \rho_2 \end{bmatrix}^\top \in \mathbb{R}^2$ the vectors connecting the center of mass to the left and right feet frames respectively.

Since ${}^cX_f^* \in \mathbb{R}^{3\times 6}$ is always full row rank by construction, (7.15) admits infinite solutions, i.e.

$$ f = {}^cX_f^{*\dagger} mge_2 + N_f \gamma \tag{7.16} $$

Then, the criterion 1) corresponds to the choice $\gamma = 0$. The criterion 2), instead, involves the evaluation of the joint torques associated with the wrenches f, and then the exploitation of γ to minimize the joint torques.

7.3.2.1 *Minimum contact wrench norm solution*

By setting $\gamma = 0$ in Eq. (7.16), one can show that the wrench acting on the right foot is given by

$$ {}^Rf_R = \frac{1}{5mg(d^2+4)} \begin{bmatrix} 0 \\ \frac{20+5d^2+6ld\cos(\xi)}{2} \\ 6ld\cos(\xi) \end{bmatrix}. $$

The force acting on the left foot is similar to the above expression. If d is small, e.g. the distance between the two feet is smaller than one meter, then the above expression points out that one has an *almost* constant vertical force – equal to $mg/2$ – independently of ξ, i.e. independently of the center of mass position.

Figure 7.3 shows the static CoP of the left and right foot in correspondence of the CoM motion. We can notice how σ_L and σ_R moves together with the CoM leading to the breakage of the contact.

To evaluate the the static center of pressure sensitivity, we can differentiate the CoP expression w.r.t ξ, that is

$$ \begin{aligned} \eta_R = \frac{\partial \sigma_R}{\partial \xi} &= \frac{\partial}{\partial \xi}\left(\frac{12l\cos(\xi)}{5d^2 - 6ld\cos(\xi) + 20}\right) \\ &= -60l\sin(\xi)\frac{d^2+4}{(5d^2 - 6ld\cos(\xi) + 20)^2} \end{aligned} $$

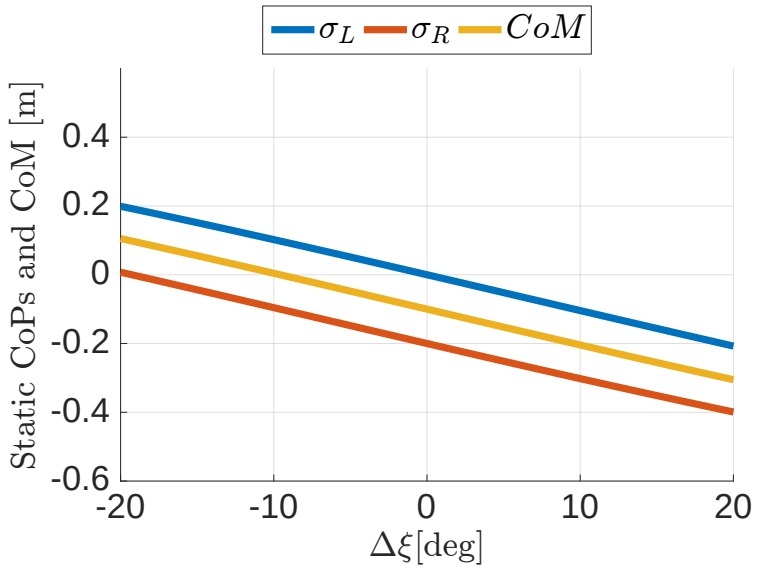


Figure 7.3: Left and right foot CoPs for the minimum wrenches norm solution. The CoM is plotted in yellow

7.3.2.2 *Minimum joint torque norm solution*

Let us consider again (7.15) and impose the following expression for the contact wrenches:

$$\begin{aligned} {}^{L}f_L &= {}^{c}X_L^{*-1}\frac{mg}{2}e_2 + {}^{c}X_L^{*-1}\Delta \\ {}^{R}f_R &= {}^{c}X_R^{*-1}\frac{mg}{2}e_2 - {}^{c}X_R^{*-1}\Delta \end{aligned} \tag{7.17}$$

where $\Delta \in \mathbb{R}^3$ is a free variable. We can notice that with the above choice, (7.15) is always satisfied independently of the choice of Δ. In what follows, the redundancy Δ is used to minimize the joint torques. The equations of motion, however, are written with the formalism presented in Lemma 7.1, which simplifies the process of obtaining the value of Δ that minimizes the joint torques. So, from now on, the mass matrix, the Coriolis term, and the Jacobians are overlined.

The relation between the joint torques and the contact wrenches is given by Eq. (7.5). So, by inverting this relation, we evaluate the effects of the contact wrench redundancy Δ on the joint torques at the equilibrium configuration, i.e.

$$\tau = (\bar{J}_j\bar{M}_j^{-1})^{\dagger}\bar{J}\bar{M}^{-1}(mge_3 - \bar{J}^{\top}f). \tag{7.18}$$

We can now substitute the definition of f as in (7.17) into the obtained expression of $\tau = \tau(f)$ in (7.18), i.e.

$$\begin{aligned} \tau = (\bar{J}_j\bar{M}_j^{-1})^{\dagger}\bar{J}\bar{M}^{-1}[&(2 - \bar{J}_L^{\top}\,{}^{c}X_L^{*-1} - \bar{J}_R^{\top}\,{}^{c}X_R^{*-1})\frac{mg}{2}e_3 \\ &-(\bar{J}_L^{\top}\,{}^{c}X_L^{*-1} - \bar{J}_R^{\top}\,{}^{c}X_R^{*-1})\Delta]. \end{aligned} \tag{7.19}$$

Now, it is easy to verify that

$$\bar{J}_L^\top {}^cX_L^{*-1} + \bar{J}_R^\top {}^cX_R^{*-1} = \begin{bmatrix} 2\,1_6 \\ J_{L,j}^\top {}^cX_L^{*-1} + J_{R,j}^\top {}^cX_R^{*-1} - 2\frac{M_{bj}^\top}{m}\Lambda^\top \end{bmatrix}$$
$$\bar{J}_L^\top {}^cX_L^{*-1} - \bar{J}_R^\top {}^cX_R^{*-1} = \begin{bmatrix} 0_{6\times 6} \\ J_{L,j}^\top {}^cX_L^{*-1} - J_{R,j}^\top {}^cX_R^{*-1} \end{bmatrix}.$$

Observe also that

$$P^\top e_3 = \begin{bmatrix} 0_{3\times 3} & -S(r) \\ 0_{3\times 3} & \frac{I}{m} - 1_3 \end{bmatrix} e_3 = 0_6 \Rightarrow \Lambda^\top e_3 = 1_6$$

then (7.19) becomes

$$\tau = -(\bar{J}_j \bar{M}_j^{-1})^\dagger \bar{J}_j \bar{M}_j^{-1} \left[(J_{L,j}^\top {}^cX_L^{*-1} + J_{R,j}^\top {}^cX_R^{*-1} - 2\frac{M_{bj}^\top}{m}) \frac{mg}{2} e_3 + (J_{L,j}^\top {}^cX_L^{*-1} - J_{R,j}^\top {}^cX_R^{*-1})\Delta \right]. \quad (7.20)$$

If we consider the four-bar linkage model, we can notice that the number of rows of J_j are greater than the DoFs of the system and thus the product of the first two terms simplifies into the identity matrix. As a consequence, the vector Δ that minimizes the joint torques is given:

$$\Delta = -\frac{mg}{2} \left(J_{L,j}^\top X_L^{-1} - J_{R,j}^\top X_R^{-1} \right)^\dagger \left(J_{L,j}^\top X_L^{-1} + J_{R,j}^\top X_R^{-1} - 2\frac{M_{bj}^\top}{m} \right) e_2 \quad (7.21)$$

which, by using the actual expression for the Jacobians, boils down to

$$\Delta = \begin{bmatrix} 3l\cos^2(\xi)/10d\sin(\xi) \\ 3l\cos(\xi)/10d \\ d/4 \end{bmatrix}. \quad (7.22)$$

We can now compute the center of pressures σ_L and σ_R. As before we focus only on the right foot static CoP knowing that the same analysis can be done equally on the other foot.

Taking the forces expression in (7.17), the static CoP of the right foot is

$$\sigma_R = \frac{-\frac{1}{2}\rho_1 - \rho_2\Delta_1 + \rho_1\Delta_2 - \Delta_3}{\frac{1}{2} - \Delta_2}. \quad (7.23)$$

By plugging the expression of Δ we found before we obtain the expression of the static center of pressure as a function of the free variable ξ. Figure 7.4 shows its behaviour.

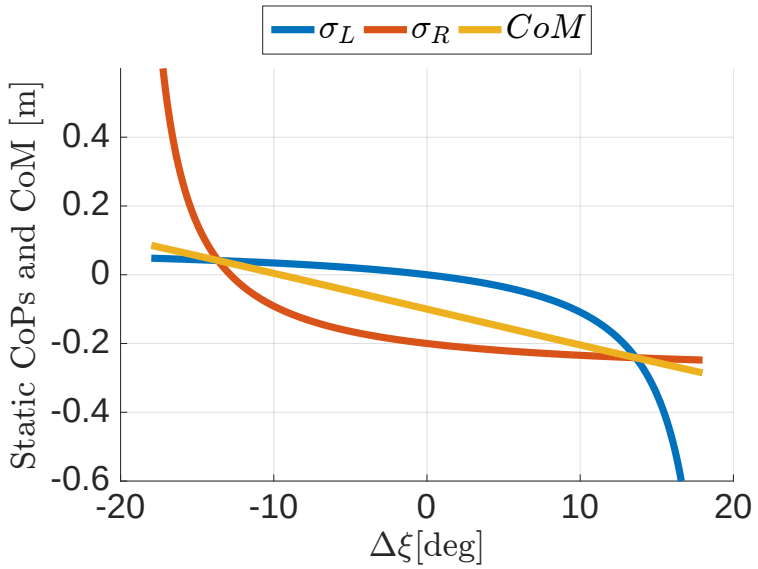


Figure 7.4: Left and right foot CoPs, and CoM during robot motion. The asymptot is in correspondence of the forces going to zero

Interestingly, while the solution exhibits more stable CoPs around the centered position, we can notice a divergent behaviour around ± 18[deg]. This is due to the fact that with this solution, the force on one foot goes to zero as the weight is moved toward the other foot.

We want to note that in a real application the introduction of inequalities constraints on the CoPs, as we presented in the previous chapter, are thus mandatory. Figure 7.4 shows the effects of choosing Δ such that the torques are minimized. As expected the CoP are limited to remain inside the boundaries of the support polygon of the foot.

We can compare the static centers of pressure for the two different force solution applied to the system by computing their sensitivity.

We can differentiate the CoP expression w.r.t ξ, that is

$$\eta_R = \frac{\partial \sigma_R}{\partial \xi} = \frac{\partial}{\partial \xi}\left(\frac{3ld\cos(\xi)}{10d - 6l\cos(\xi)}\right)$$
$$= -\frac{15ld^2\sin(\xi)}{2(5d - 3l\cos(\xi))^2}$$

7.3.3 STATIC COP SENSITIVITY ANALYSIS

We now analyse the sensitivity of the static CoP for the four-bar linkage model when we apply the two different forces solution in (7.16) and (7.20).

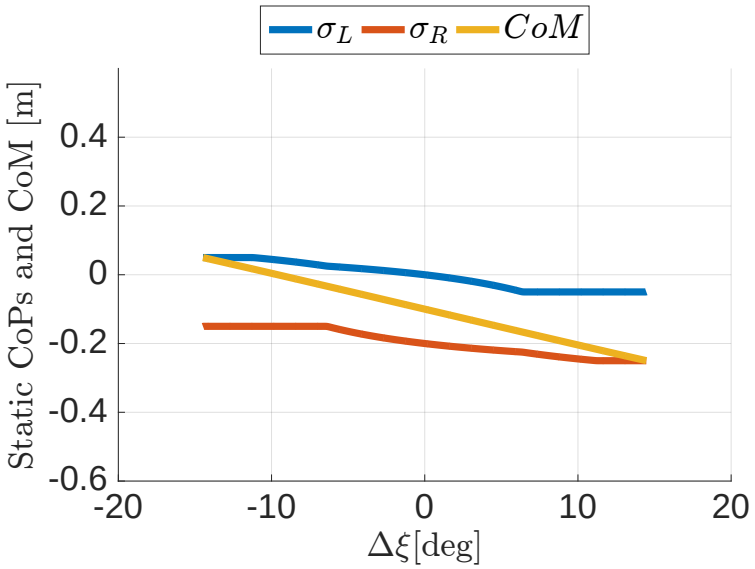


Figure 7.5: Left and right foot CoPs, and CoM during robot motion. CoPs remains limited because of the presence of the QP solver

If we evaluate the sensitivity at the initial configuration, i.e. $\xi = \frac{\pi}{2}$, we obtain that $\eta_R = -0.5941$m/rad for the minimum wrenches norm and $\eta_R = -0.3$m/rad for the minimum torque norm solution. As it is confirmed by the numerical evaluation in Figure 7.6 the second solution is twice less sensitive than the first one.

7.4 EXPERIMENTAL EVALUATION ON THE ICUB HUMANOID ROBOT

In this section, we present the experimental validation performed on the iCub humanoid robot (see Chapter 3 for a description of the platform) of the analysis previously presented on the planar four-bar linkage system. The control torques have been chosen as described in Equation (6.18).

The contact forces are chosen as in (7.16). Each scenario has a different choice for the γ term, i.e. we change how we exploit the redundancy of the forces. In particular we perform the following tests:

i) Minimum wrenches norm solution, i.e. γ

ii) Minimization of joint torques (i.e. (6.19)) without inequality constraints

iii) Minimization of joint torques (i.e. (6.19)) with constraints on the CoPs.

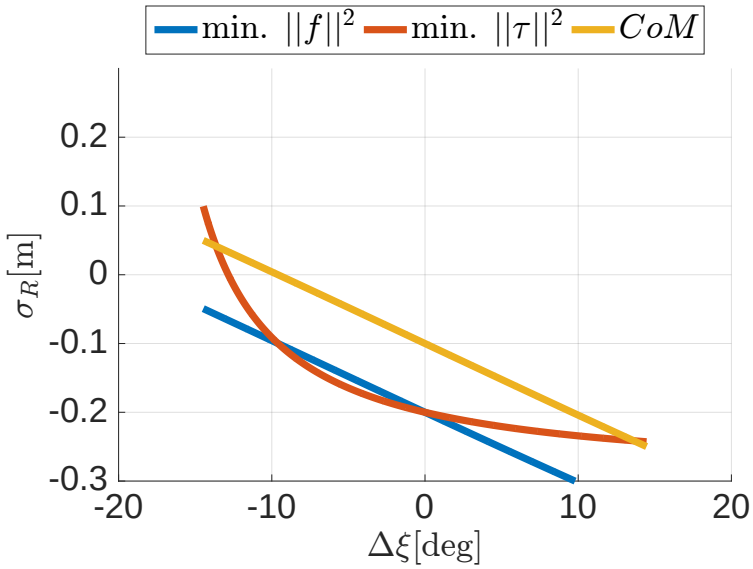


Figure 7.6: Comparison between the right foot CoPs for two different forces choice. The CoM is also plotted in yellow

In all the experiments we moved the robot with very low velocity so as to simulate the quasi-static scenario from the initial, symmetric configuration to the configuration where the CoM lies on the left foot frame origin. Note also that in this case the static center-of-pressure can be approximated with the real center of pressure.

We first tested the minimum wrenches norm solution, i.e. $\gamma = 0$ in (6.18). As we showed in the analysis we expect that the vertical components of the forces do not change so much from the half weight of the robot. This is confirmed also on the real robot. As we can see in Figure 7.7 the forces start symmetrical. When the CoP starts moving toward the left foot we notice that both the forces changes of only a limited quantity.

We then tested the second control law, i.e. we exploited the redundancy of the forces to optimize for the joint torques as formulated in (6.19). Figure 7.8 shows the forces during the experiment, while Figure 7.9 shows the CoPs for the left and right feet together with the CoM during the experiment. We can notice that at $t = 4s$ the CoP of the right foot touches the boundaries of the support of the foot. This corresponded to a fall of the robot.

Finally we added all the inequality constraints specified in (6.19) and we solved the optimization problem by means of a QP solver. Figure 7.10 shows the CoPs during the robot motion. We can observe that the solver manages to limit the CoP inside

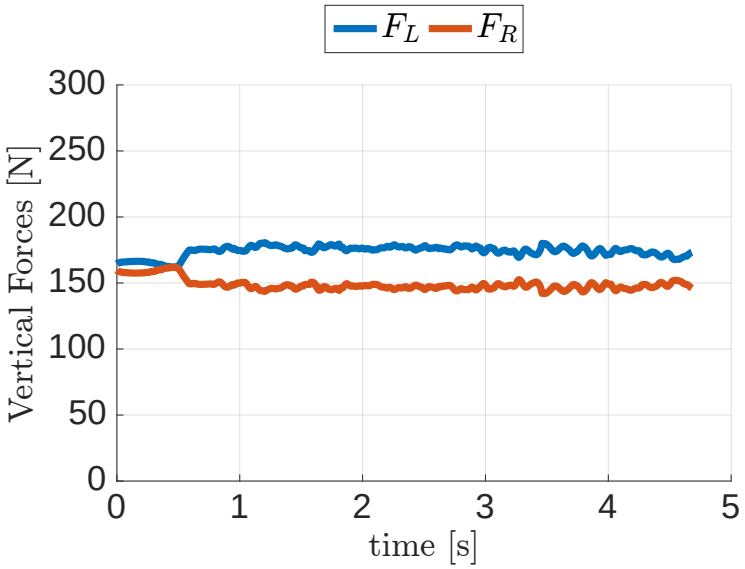


Figure 7.7: Left and right foot forces for the minimum norm of the forces solution

the foot limit. Forces are similar to the ones in Figure 7.8 and are thus not reported.

We finally compare the CoPs of the right foot when we apply the minimum wrenches norm solution and the minimum torque solution, and the results are plotted in Figure 7.11. We also plotted in dashed the sensitivity of the two solutions. Also on the real robot, the sensitivity of the solution with $\gamma = 0$ is greater than the other one.

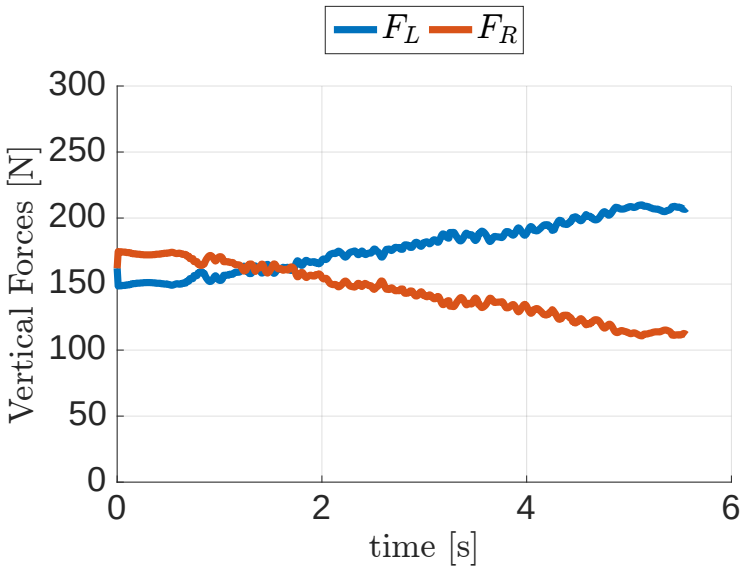


Figure 7.8: Left and right foot forces for the minimum norm of the torques solution

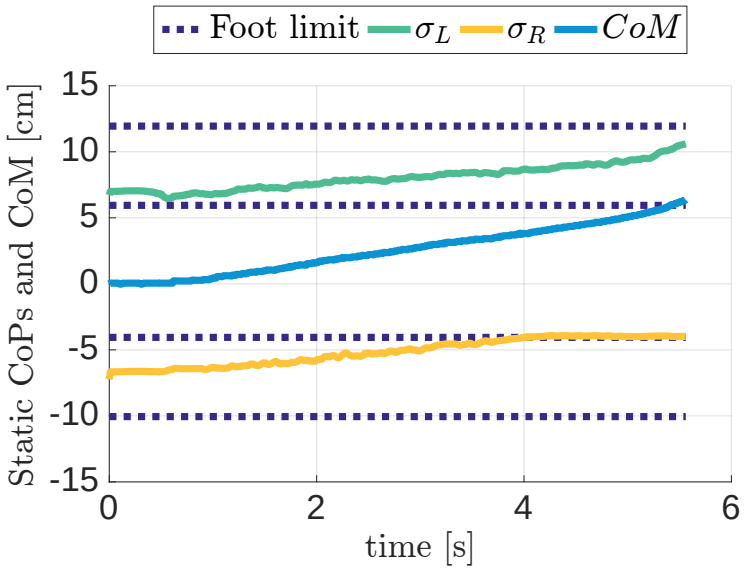


Figure 7.9: CoPs and CoM when the minimum torque solution without the use of a QP solver

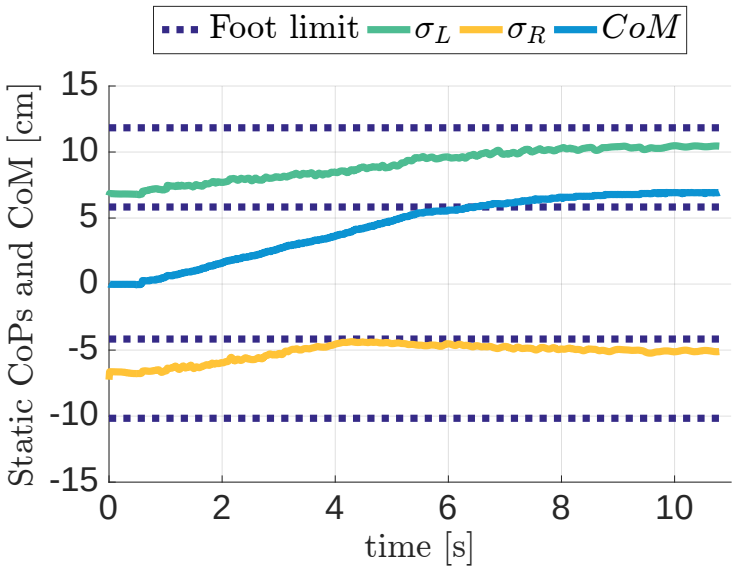


Figure 7.10: CoPs and CoM when the minimum torque solution with the QP is applied

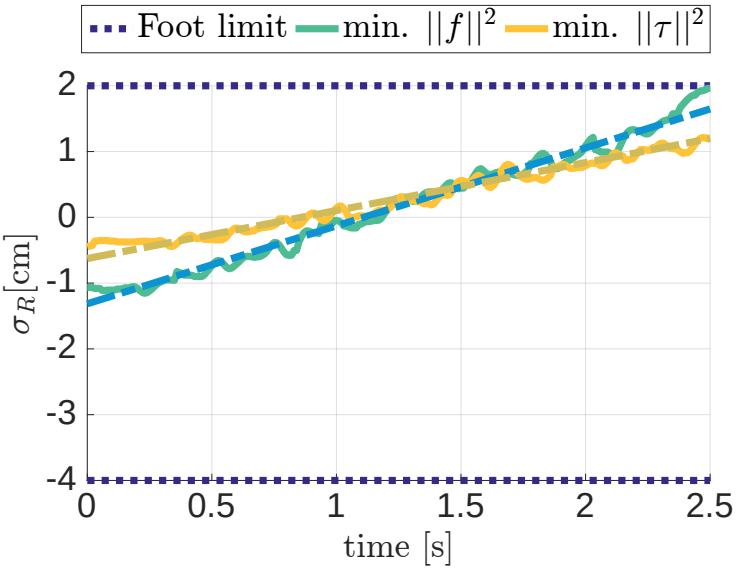


Figure 7.11: CoPs of right foot for two wrenches choice. In dashed the linear regression showing the different sensitivities

8

STABILITY ANALYSIS OF THE ZERO DYNAMICS

As we remarked in Section 6.2.2 it does not exist, as far as the authors know, a proof that the so called *postural task*, in the momentum-based controllers, ensures stability of the zero dynamics.

The task, in the presented form, has been proposed as an empirical "recipe" to avoid divergent behaviours of the joint space dynamics. There are some results on the stability of the zero dynamics for fully actuated manipulators when controlled by stack-of-tasks-based controllers but the underlining hypotheses do not make the proposed analysis applicable to the controller presented in this work [77].

The analysis is performed on a dynamical system where we assume the mass matrix is block diagonal. If this was not the case, it is possible to perform a change of the configuration and velocity (q, ν) variables as we described in Section 7.2. Note that this assumption is not necessary for the results presented, but it simplifies the expression of the terms in the dynamic equation. Because of the above assumption, we drop the overlined notation in Section 7.2.

In order to perform an analytical analysis of the stability of the zero dynamics, we neglect the inequality constraints in the optimization problem (6.7) so as to obtain an analytical expression of its solution. The resulting control laws we apply to the dynamical system are thus the following:

$$\begin{aligned} \tau &= (J_j M_j^{-1})^{\dagger} (J M^{-1} (h - J^{\top} f) - \dot{J}\nu) + N_{\tau}\varphi \\ f &= J_b^{-\top} \left(\dot{H}^d + mge_3 \right) \\ \varphi &= h_j - J_j^{\top} f - K_p^q (q_j - q_j^d) - K_d^q \dot{q}_j. \end{aligned} \tag{8.1}$$

with N_τ the projector onto the nullspace of $J_j M_j^{-1}$. Notice that the above choice for the contact forces implies the choice of $\gamma = 0$ in Eq. (6.16), i.e. we choose the minimum norm of the contact wrenches solution. In fact, the choice of γ has no influence on the stability analysis of the zero dynamics as we now show.

Consider the joint space dynamics of the system in the decoupled form, i.e. as in (7.10):

$$M_j \ddot{q}_j + h_j - J_j^\top f = \tau.$$

By applying the control laws (8.1), with the definition

$$v := K_p^q (q_j - q_j^d) + K_d^q \dot{q}_j \quad \Rightarrow \quad \varphi = h_j - J_j^\top f - v,$$

one obtains the following expression:

$$M_j \ddot{q}_j = (J_j M_j^{-1})^\dagger (J_b M_b^{-1} (h_b - J_b^\top f) - \dot{J} \nu) - N_\tau v.$$

Notice that, because of the specific choice of the base in (7.10), the following result holds

$$J_b^\top N_f = 0,$$

and, as a consequence, one shows that the variable γ does not have any influence in the joint space dynamics.

Lastly, to perform the analytical analysis, we do an additional assumption:

Assumption 8.1. *Only one frame associated with a robot link has a constant position-and-orientation with respect to the inertial frame.*

Without loss of generality, it is assumed that the only constrained frame is that between the environment and one of its feet. This hypothesis is necessary in order to obtain an expression for the set of minimum coordinates of the constrained dynamical system. Note that it is not trivial to identify this set in case of different type or number of constraints. Nevertheless, the simulations and experimental results are obtained both on the single foot and two feet scenarios.

8.1 NUMERICAL EVIDENCE OF UNSTABLE ZERO DYNAMICS

This section shows evidence in simulation that the zero dynamics obtained by applying the control laws (8.1) with the choice of K_i^h in Eq. (6.6), lead to unstable zero dynamics. Because of the presence of friction which acts as a dissipative element on the real robot, the divergent behaviour is more easily seen on simulation.

8.1.1 SIMULATION ENVIRONMENTS

Two different simulation setups have been exploited to perform the numerical validation.

8.1.1.1 *Custom setup*

The first simulation setup used to perform the validation is a custom setup that is in charge of integrating the dynamics (7.10) when it is subject to the constraint (2.19). We parametrize $SO(3)$ by means of a quaternion representation

$$\mathcal{Q} \in \mathbb{R}^4 .$$

The resulting system state, which is integrated through time, is then:

$$\chi := \begin{bmatrix} p_{\mathcal{B}}^\top & \mathcal{Q}^\top & q_j^\top & \dot{p}_{\mathcal{B}}^\top & \omega_{\mathcal{B}}^\top & \dot{q}_j^\top \end{bmatrix}^\top ,$$

whose derivative is given by

$$\dot{\chi} = \begin{bmatrix} \dot{p}_{\mathcal{B}} \\ \dot{\mathcal{Q}} \\ \dot{q}_j \\ \dot{\nu} \end{bmatrix} .$$

This formulation corresponds to the one described in Remark 2.3.1.

The constraints (2.19), as well as $|\mathcal{Q}| = 1$, are then enforced during the integration phase, and additional correction terms are thus necessary. The evolution of the system is then obtained by integrating the constrained dynamical system with MATLAB *ode15s* numerical integrator.

8.1.1.2 *Gazebo setup*

The Gazebo simulator [51] is the other simulation setup used for our tests on the stability of the zero dynamics. Of the different physic engines that can be used with Gazebo, we choose the Open Dynamics Engine (ODE). Differently from the previous simulation environment, Gazebo allows one for more flexibility. Indeed, we only have to specify the model for the robot, and the constraints arise naturally during the simulation. Furthermore, Gazebo integrates the dynamics with a fixed step semi-implicit Euler integration scheme.

An advantage of using Gazebo w.r.t. the custom integration scheme previously presented consists in the ability to test in simulation the same control software used on the real robot thanks to its integration with the YARP middleware [66]. While the use of the custom scheme allows a more precise simulation of the dynamical system, effects of the discretization of the control laws are more difficultly perceived.

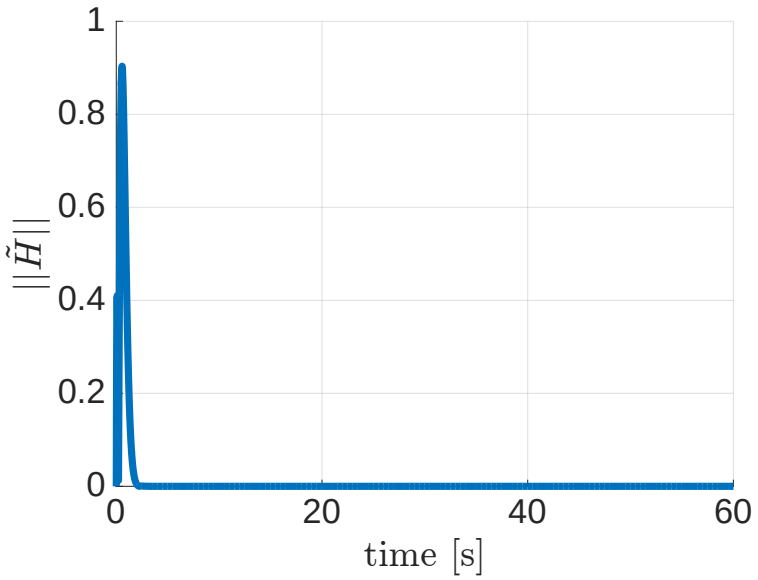


Figure 8.1: Time evolution of the robot momenta error when standing on one foot and when the control law (8.1) is applied. Simulation run with the custom environment.

8.1.2 UNSTABLE ZERO DYNAMICS

To show that the momentum-based control strategies may lead to unstable zero dynamics, we control the robot linear momentum so as to follow a desired center of mass trajectory, i.e. a sinusoidal reference along the y coordinate with amplitude 0.05m and frequency 0.3Hz.

8.1.2.1 *Tests on the robot balancing on one foot in the custom simulation setup*

In this section we present simulation results obtained by applying the control laws (8.1) with K_i^h as in (6.6). Hence, we assume that the left foot is attached to ground, and no other external wrench applies to the robot.

Figure 8.1 shows typical simulation results of the convergence to zero of the robot momentum error, thus meaning that the contact forces ensure the stabilization of $\tilde{H}$ towards zero. Figure 8.2, instead, depicts the norm of the joint position error $\left\| q_j - q_j^d \right\|$. This figures shows that the norm of the joint angles increases while the robot momentum is kept equal to zero. This is a classical behaviour of unstable zero dynamics.

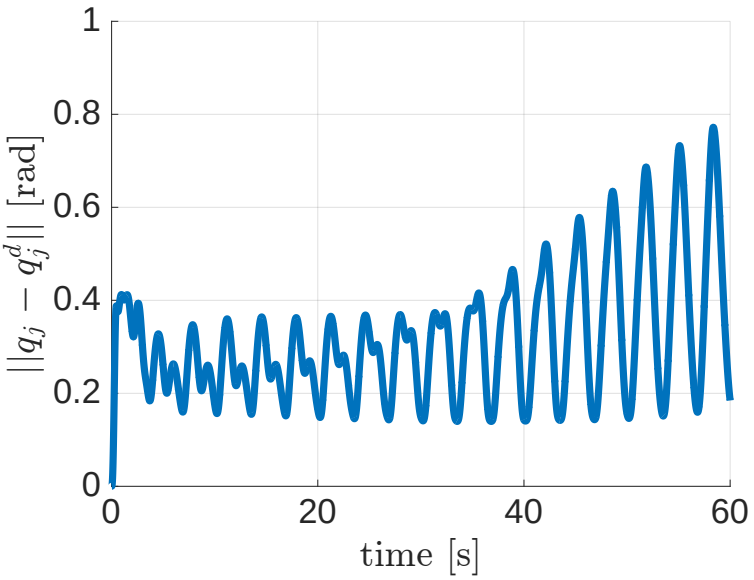


Figure 8.2: Time evolution of the norm of the position error $\left\|q_j - q_j^d\right\|$ when the robot is standing on one foot and when the control law (8.1) is applied. Simulation run with the custom environment.

8.1.2.2 *Tests when the robot balances on two feet in the Gazebo simulation setup*

The tests on the stability of the zero dynamics have been carried out on also in the case the robot stands on both feet. In this case the control applied is the one presented in Chapter 6.

Figure 8.3 depicts the norm of the joint position error $\left\|q_j - q_j^d\right\|$ when the above control algorithm is applied, and the simulation is run in the Gazebo setup. It is clear from this figure that the instability of the zero dynamics is observed also in the case the robot stands on two feet.

8.2 STABILITY ANALYSIS

To circumvent the problems related to the stability of the zero dynamics discussed in the previous section, what follows proposes a modification of the control laws (8.1) that allows us to show stable zero dynamics of the constrained, closed loop system. In particular, the following result holds.

Lemma 8.1. *Assume that Assumption (8.1) holds, and that the robot possesses more than six degrees of freedom, i.e.* $n \geqslant 6$. *In addition, assume also that* $H^r = 0$. *Let*

$$(q_j, \dot{q}_j) = (q_j^d, 0) \tag{8.2}$$

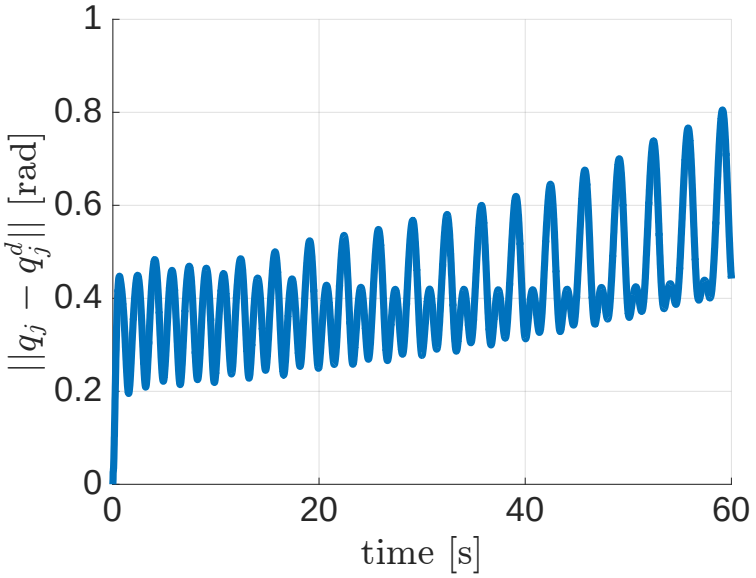


Figure 8.3: Time evolution of the norm of the position error $\left\| q_j - q_j^d \right\|$ when the robot is standing on two feet and when the control presented in Chapter 6 is applied. Simulation run with the Gazebo environment.

denote the equilibrium point associated with the constrained, closed loop system, and assume that the matrix $J_j(q)M_j^{-1}(q)$ *is full row rank in a neighbourhood of* (8.2).

Apply the control laws (8.1) *with*

$$K_i^h > 0 \tag{8.2a}$$
$$K_p^q = \bar{K}_p^q N_\tau M_j \tag{8.2b}$$
$$K_d^q = \bar{K}_d^q N_\tau M_j \tag{8.2c}$$

where $\bar{K}_p^q, \bar{K}_d^q \in \mathbb{R}^{n\times n}$ *are two constant, positive definite matrices.*

Then, the equilibrium point (8.2) *of the constrained, closed loop dynamics is asymptotically stable.*

The proof is in Appendix A.5. The above Lemma shows that the asymptotic stability of the equilibrium point (8.2) of the constrained, closed loop dynamics can be ensured by choosing a positive definite matrix K_i^h, and by modifying the gains of the postural task. Clearly, the asymptotic stability of the equilibrium point (8.2) implies that the zero dynamics is asymptotically stable.

The fact that the gain matrix K_i^h must be positive definite conveys the necessity of closing the control loop with *position-angular* terms at the momentum level. Some authors, in fact, intuitively close the angular momentum loop by using the orientation of the robot torso [76].

The proof of Lemma 8.1 exploits the fact that the minimum coordinates of the robot configuration space when Assumption 8.1 holds is given by the joint angles q_j. The analysis focuses on the closed loop dynamics of the form

$$\ddot{q}_j = f(q_j, \dot{q}_j) \tag{8.3}$$

which is then linearized around the equilibrium point (8.2). By means of a Lyapunov analysis, one shows that the equilibrium point is asymptotically stable. One of the main technical difficulties when linearizing Eq. (8.3) comes from the fact that the closed loop dynamics depends on the integral of the robot momentum (see Eq. (8.1)), i.e.

$$\int_0^t H(q_j(s), \dot{q}_j(s))\, ds$$

The partial derivative w.r.t. the state $(q_j, \dot{q}_j)$ is, in general, not obvious because of the dependence of the integrand function upon the state $(q_j, \dot{q}_j)$. It is known, however, that the momentum associated with a multi-body system is linear versus the system velocity, i.e.

$$H = J_G(q)\nu$$

where $J_G(q) \in \mathbb{R}^{6\times n+6}$ is the so-called *centroidal momentum matrix* [74]. As a consequence of Assumption 8.1, one can express the momentum H only as function of the state $(q_j, \dot{q}_j)$, which yields

$$H = \bar{J}_G(q_j)\dot{q}_j \tag{8.4}$$

where $\bar{J}_G(q) \in \mathbb{R}^{6\times n}$. Eq. (8.4) implies that the integral of the momentum along the constrained motions is given by

$$\int_0^t H(q_j(s), \dot{q}_j(s))\, ds = \int_{q_j(0)}^{q_j(t)} \bar{J}_G(q_j)\, dq_j.$$

Consequently, one has

$$\frac{\partial}{\partial \dot{q}_j(t)} \int_0^t H(q_j(s), \dot{q}_j(s)) ds = 0 \tag{8.4a}$$

$$\frac{\partial}{\partial q_j(t)} \int_0^t H(q_j(s), \dot{q}_j(s)) ds = \bar{J}_G(q_j(t)) \tag{8.4b}$$

The expression (8.5) points out that in a neighbourhood of the equilibrium point $(q_j, \dot{q}_j) = (q_j^d, 0)$, the integral of the momentum may be approximated as

$$\int_0^t H(q_j(s), \dot{q}_j(s))\, ds \approx \bar{J}_G(q_j^d)(q_j - q_j^d). \tag{8.5}$$

Hence, Eq. (8.5) may also be used as a feedback control action instead of the integral of the momentum in Eq. (8.1), thus avoiding eventual nonstationary effects of the robot motion due to the integral of measurement noise.

8.3 SIMULATIONS AND EXPERIMENTAL RESULTS

This section shows simulation and experimental results obtained by applying the control laws (8.1) and the ones presented in Chapter 6 with the modifications presented in Lemma (8.1). To show the improvements of the control modification in this Lemma, we apply the same reference signal of Section 8.1, which revealed unstable zero dynamics. Hence, the desired linear momentum is chosen so as to follow a sinusoidal reference on the center of mass. Also, control gains are kept equal to those used for the simulations presented in Section 8.1.

8.3.1 SIMULATION RESULTS

Figures 8.4 and 8.5 show the norm of the joint errors $\left\| q_j - q_j^d \right\|$ when the robot stands on either one or two feet, respectively. These two simulations are performed with the custom and Gazebo simulation environment, respectively. As expected, the zero dynamics is now stable, and no divergent behaviour of the robot joints is observed.

8.3.2 RESULTS ON THE ICUB HUMANOID ROBOT

We tested the control algorithm in Chapter 6 with the modification presented in Lemma (8.1) on the iCub humanoid robot (see Chapter 3 for a description of the platform).

Figures 8.6 and 8.7 show the joint position error $\left\| q_j - q_j^d \right\|$ and the center of mass error. Though the center of mass does not converge to the desired value, all signals are bounded, and the control modification presented in Lemma (8.1) does not pose any barrier for the implementation of the control algorithm in Chapter 6 on a real platform.

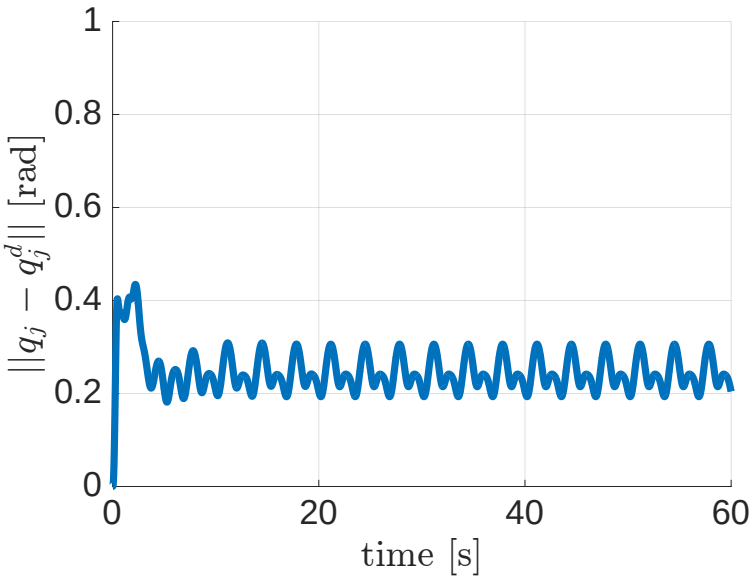


Figure 8.4: Time evolution of the norm of the position error $\left\|q_j - q_j^d\right\|$ when the robot is standing on one foot and when the control law (8.1) with the modification presented in Lemma (8.1) is applied. Simulation run with the custom environment.

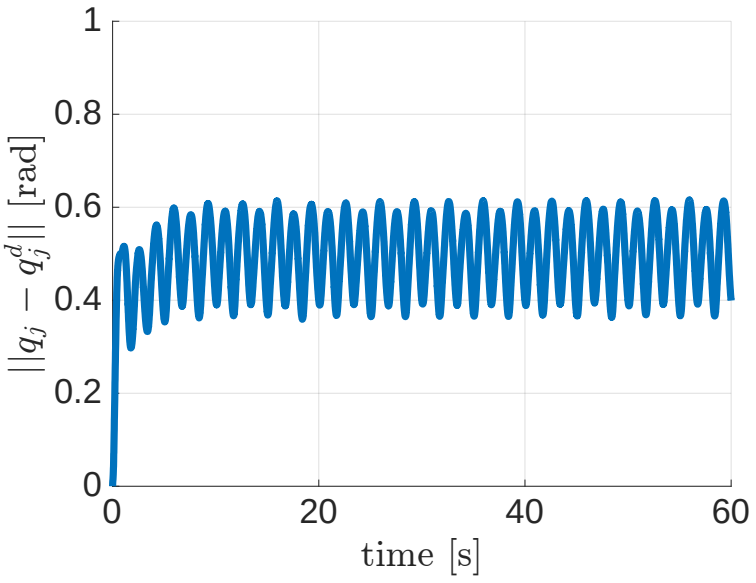


Figure 8.5: Time evolution of the norm of the position error $\left\|q_j - q_j^d\right\|$ when the robot is standing on two feet and when the control law in Chapter 6 with the modification presented in Lemma (8.1) is applied. Simulation run with the Gazebo environment.

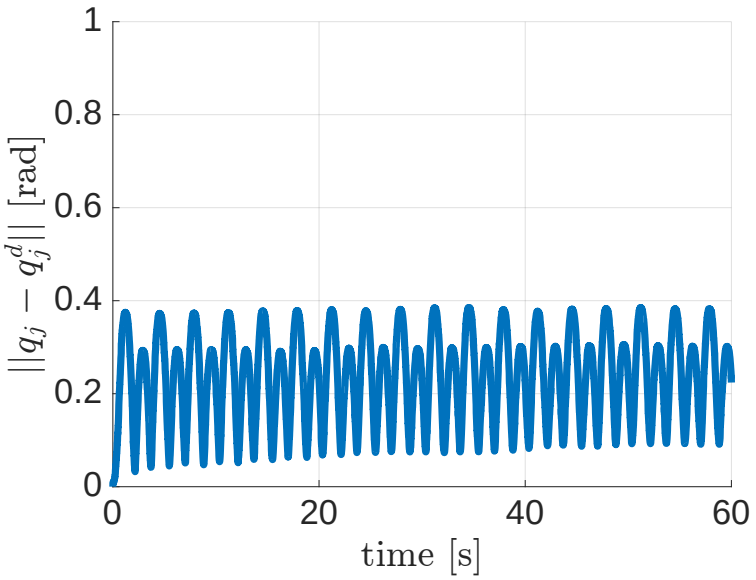


Figure 8.6: Time evolution of the norm of the position error $\left\|q_j - q_j^d\right\|$ when the robot is standing on two feet and when the control law in Chapter 6 with the modification presented in Lemma (8.1) is applied. Experiment run on the humanoid robot iCub.

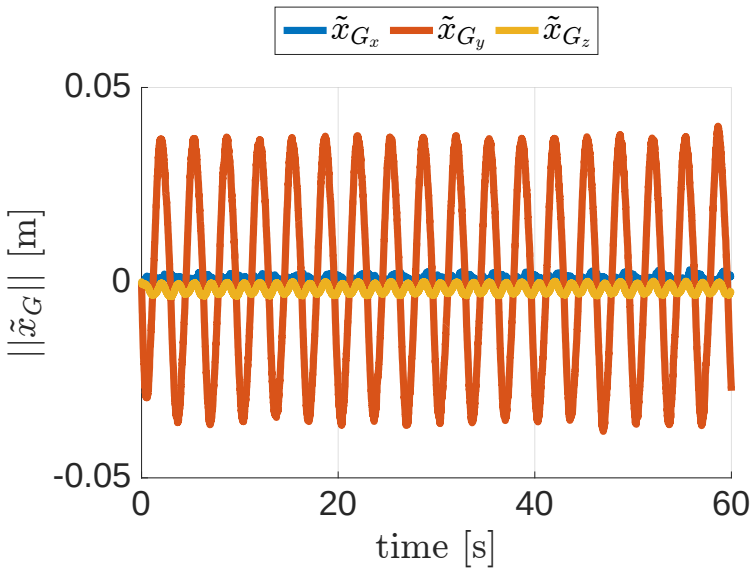


Figure 8.7: Time evolution of the robot center-of-mass error when standing on two feet foot and when the control law in Chapter 6 with the modification presented in Lemma (8.1) is applied. Experiment run on the humanoid robot iCub.

9 DISCUSSION AND CONCLUSIONS

Chapter 6 presented the whole-body torque control architecture implemented on the iCub humanoid robot for balancing purposes. The control objectives are the control of the robot momenta and the stability of the zero dynamics. These objectives are put into a two tasks hierarchy, where the former has the highest priority. Cartesian motion tasks are mapped to joint trajectories, solving eventual conflicts beforehand, and the resulting references are used in the zero dynamics stability task.

Examining the control of the robot momenta, it is well known that, in case the number of contacts with the environment is greater than one, the choice of the contact wrenches is not unique. While often neglected in literature, this choice has a critical impact on the robot performances while balancing. Chapter 7 analysed two different choices of the contact wrenches when the robot balances on two feet. Specifically, we examined the minimum wrench norm solution, usually applied in literature, and the minimum joint torques norm solution. The analytical analysis is performed on a simplified model, but experimental validation on the real robot has also been shown to prove the validity of the approach.

Finally, in Chapter 8 we presented a stability analysis of the zero dynamics. The analysis performed on the stack of tasks method showed that the presence of the integral term on the robot momenta ensures stability of the zero dynamics. The analysis and stability proof has been performed on the single-contact case, but it has been validated for both one and two contacts on simulation and on the real robot.

We conclude by describing the different scenarios in which the presented control has been successfully employed. We first tested the control to make the robot balance on its two feet. By properly specifying the desired robot momenta, it is straightforward to control the trajectory of the center of mass, e.g. to keep the initial center of mass position or to make the center of mass follow a sinusoidal reference.

Secondly, we tested the addition of cartesian tasks and their integration in the two-layer hierarchy. In particular, as high-

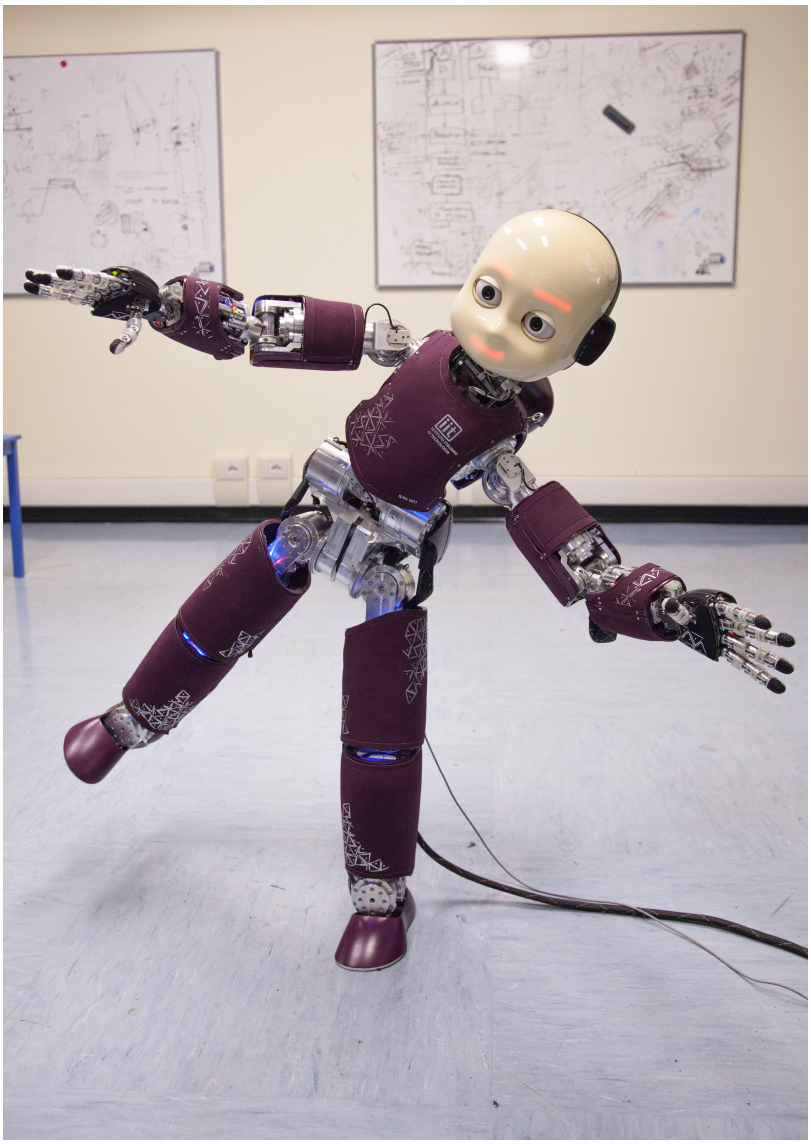


Figure 9.1: iCub while balancing on its left foot. Complex movements of the joints can also be added.

lighted in Chapter 6, cartesian tasks yield joint trajectory references for the zero dynamics stability task. In the scenario, cartesian trajectories for the two hands end-effectors of the robot were mapped into joint trajectories by using an inverse kinematics solver. The robot successfully tracked the end-effectors references provided by the inverse kinematics module while balancing on its two feet.

We then moved forward and tested the capabilities of the balancing while standing on only one foot as shown in Figure 9.1. Additional joints trajectories were added, thus rendering iCub capable of performing fast and complex movements while keeping its balance on only one foot. Starting from the initial, sym-

metrical, configuration on two feet, iCub moves its weight on one leg before switching from two to one foot balancing. A critical point during the execution of this scenario is the change occurring in the constraints set. Indeed, the constrained dynamical system changes between the two considered configurations, i.e. from two feet to one foot balancing. As a consequence, during the contact switch, it may be possible that discontinuities occur in the contact forces. To properly render the dynamical transition as smooth as possible, we adopt two strategies. Firstly, as the analysis in Chapter 7 showed, the use of the joint torque minimization has a great impact on the resulting vertical component of the contact forces, i.e. they progressively reduce as the weight moves from one leg to the other. Secondly, we introduce a smoothing coefficient in the contact Jacobians to account for the remaining force components. Note that the scenario just described is preparatory to implement a torque controlled walking, but this application is outside the scope of the present thesis.

Part IV

CONCLUSIONS AND FUTURE WORK

10 CONCLUSIONS AND FUTURE WORK

Free-floating mechanical systems, such as humanoid robots, do pose interesting issues to the research community. The fact that they are underactuated significantly complexify the associated control problem. The high number of degrees of freedom of humanoid robots, while enabling them to perform multiple tasks at the same time, introduces also some difficulties to be tackled.

In this thesis we presented different approaches to deal with the control of underactuated free-floating mechanical systems. We now summarize the results and we conclude the thesis with a discussion on future work.

10.1 SUMMARY

Part II described two different control strategies. Adaptive control is one of the available control techniques to cope with model uncertainties. We extended (c.f. Chapter 4) the classical adaptive control by Slotine by including the possibility to control a subset of the configuration variables of an underactuated mechanical system. In particular we focused on the control of the collocated variables. The control laws presented are not defined globally because the inversion of the estimated mass matrix is required, which in general is not positive definitive, or even invertible. We showed that a desingularization policy can be employed, but the stability results are not guaranteed when it is acting.

Chapter 5 has been devoted to extend the optimal control formulation to include strict task prioritization. In literature prioritized control is a quite established technique. Born for representing cartesian velocity tasks, it has been extended to also consider the dynamics of mechanical systems. On the contrary, optimal control is naturally specified as a single task problem, where multiple tasks are gathered together into a single cost. Prioritized optimal control tries to merge the benefits of strict prioritization into the optimal control formalism. Considering that a humanoid robot can execute simultaneously multi-

ple tasks, possibly in conflict, task weight tuning can become a time-consuming and error prone procedure.

Part III presented a whole-body control formulation applied to free-floating mechanical systems. More in detail Chapter 6 described the balancing problem and the formalization of the control objectives. Specifically, the control objectives are the control of a desired momenta and of the zero dynamics. These two objectives are represented as control tasks and organized into a two layers hierarchy. The whole hierarchy is then transcribed into an optimization problem and the resulting quadratic program is solved by means of a dedicated solver. Chapter 7 analysed how the choice of the contact forces, and as a consequence of the resulting centers of pressures, while balancing on two feet is crucial for the robot performances and must be accurately considered. The analytical analysis has been performed on a simplified model and subsequently validated with experiments on the real robot. Finally, Chapter 8 presented a stability analysis of the zero dynamics for momentum-based controllers. We showed how the classical choice of closing the loop on the angular momentum only, i.e. without considering an *angular-position* term, may lead to unstable zero dynamics. Conversely, we proved that the addition of that term guaranteed the asymptotic stability of the zero dynamics.

10.2 DISCUSSION AND FUTURE WORK

The adaptive control method presented in Chapter 4 can be applied to underactuated mechanical systems, of which the free-floating systems are a particular case. For example, the stabilization of the end-effector trajectory of a manipulator attached at a space ship can be done with the introduced method. If the external forces and contact points are measurable, the method can be applied, in principle, also to humanoid robots in contact with the environment. From the theoretical point of view the main extension of this work is improving the estimation dynamics. In particular, if the dynamics of the base parameters guaranteed their physical consistency, the desingularization policy would not be needed, and the control laws would be always applicable, and by consequence, the stability analysis presented in the chapter.

The prioritized optimal control presented in Chapter 5 has proved itself in the simulations we performed. There are still two main limitations which must be addressed, namely, the inclusion of inequality constraints in the formulation and the use

of a dynamics formulation with contacts. From the theoretical point of view, the algorithm proposed does not make any assumptions on the presence of contacts in the dynamic function, but from a practical viewpoint contacts can create discontinuities during the forward integration of the dynamics, and their effect should be examined with care. Regarding the inequality constraints instead, their presence is necessary for the applicability of the algorithm. For example, in a minimum time reaching task, torque limits should be taken into account, otherwise the algorithm may generate high, unfeasible, torques.

The whole-body control presented in Part III has been applied as a balancing controller for the iCub humanoid robot in different scenarios. The same controller, in fact, has been used to i) balance on two feet; ii) balance on a single foot while performing complex movements, and iii) balance while following additional cartesian tasks. The presented thesis (c.f. Chapter 7 and Chapter 8) goes in the direction of analysing the properties of the solution obtained by the QP solver. In particular we focused on the choice of the contact forces and their consequence on the performances of the robot and on the analysis of the stability of the zero dynamics. Nonetheless some theoretical aspects still remain and should be addressed. Specifically, a critical point needed to prove the stability of constrained dynamical system is to define the minimum set of coordinates identifying its evolution. While this can be straightforward in case of one contact, the extension to multiple contacts is not trivial and must be considered carefully. Furthermore, the stack of tasks approach strongly resembles a cascade of dynamical systems. It is the authors' opinion that the stability of the whole system can be proved by using the more general, and more elegant, framework of stability of interconnected systems.

Part V

APPENDIX

A PROOFS

A.1 PROOF OF LEMMA 4.1

Consider the following candidate Lyapunov function

$$V := \frac{1}{2}\Big[s^\top M s + \tilde{\pi}^\top \Gamma^{-1}\tilde{\pi} + 2e^\top K\Lambda_1 e + 2(\tilde{\xi} + \Lambda_1 e)^\top \Lambda_1 K\Lambda_2^{-1}(\tilde{\xi} + \Lambda_1 e)\Big]. \tag{A.1}$$

We can verify that (A.1) is a Lyapunov candidate function. Indeed,

$$V = 0 \iff (e, \tilde{\xi}, s, \tilde{\pi}) = (0_n, 0_n, 0_n, 0_p).$$

Note also that since K, Λ_1, and Λ_2 are diagonal matrices, then the products $K\Lambda_1$ and $\Lambda_1 K\Lambda_2^{-1}$ are diagonal and positive definite matrices, thus

$$V(e, \tilde{\xi}, s, \tilde{\pi}) > 0.$$

We can now compute the time derivative of (A.1).

$$\begin{aligned}
\dot{V} &= s^\top M\dot{s} + \frac{1}{2}s^\top \dot{M}s + \tilde{\pi}^\top \Gamma^{-1}\dot{\tilde{\pi}} + 2e^\top K\Lambda_1\dot{e} \\
&\quad + 2(\tilde{\xi} + \Lambda_1 e)^\top \Lambda_1 K\Lambda_2^{-1}(\dot{\tilde{\xi}} + \Lambda_1\dot{e}) \\
&= s^\top \left[M\ddot{q} - M\dot{\xi} + \frac{1}{2}\dot{M}s\right] - \tilde{\pi}^\top Y^\top(q, \dot{q}, \xi, \dot{\xi})s + 2e^\top K\Lambda_1\dot{e} \\
&\quad + 2(\xi - \dot{r} + \Lambda_1 e)^\top \Lambda_1 K\Lambda_2^{-1}(-\Lambda_2 e) \\
&= s^\top \left[\tau - (M\dot{\xi} + C\xi + g) - Cs + \frac{1}{2}\dot{M}s - Y(\cdot)\tilde{\pi}\right] \\
&\quad + 2e^\top K\Lambda_1\dot{e} + 2(\dot{r} - \Lambda_1 e - \xi)^\top \Lambda_1 Ke \\
&= s^\top \left[Y(\cdot)\hat{\pi} - Ks - Y(\cdot)\pi - Y(\cdot)\tilde{\pi}\right] \\
&\quad + 2e^\top K\Lambda_1\dot{e} + 2(\dot{r} - \Lambda_1 e - \xi)^\top \Lambda_1 Ke
\end{aligned}$$

where we used Properties 2.2, 4.1, the controls (4.3a)-(4.3c), and the definitions in (4.2). We thus obtain following expression

$$\dot{V} = -s^\top Ks + 2e^\top K\Lambda_1\dot{e} + 2(\dot{r} - \Lambda_1 e - \xi)^\top \Lambda_1 Ke. \tag{A.2}$$

From Eq. (4.3b) observe that the variable ξ can be obtained by integration, i.e.

$$\xi(t) = \dot{r} - \Lambda_1 e - \Lambda_2\zeta,$$

with

$$\zeta := \int_0^t e(z)dz + \Lambda_2^{-1}\alpha,$$

and $\alpha = \dot{r}(0) - \xi(0) - \Lambda_1 e(0) \in \mathbb{R}^n$, i.e. all initial conditions. By substituting

$$s = \dot{q} - \xi = \dot{e} + \Lambda_1 e + \Lambda_2 \zeta$$

in the first term on the right hand side of (A.2) one has:

$$\dot{V} = -(\dot{e} + \Lambda_2\zeta)^\top K(\dot{e} + \Lambda_2\zeta) - e^\top \Lambda_1 K \Lambda_1 e. \tag{A.3}$$

As a consequence, $\dot{V} \leqslant 0$. Then, the stability of

$$(e, \tilde{\xi}, s, \tilde{\pi}) = (0_n, 0_n, 0_n, 0_p)$$

and the boundedness of the closed loop trajectories for any initial condition follow [45, Th. 4.8, p. 151].

To show that the tracking error e converges to zero, let us first prove that $\dot{V}$ converges to zero. We want to prove that $\ddot{V}$ is bounded, which in turn implies that $\dot{V}$ is uniformly continuous, and the application of Barbalat's Lemma ensures that $\dot{V}$ tends to zero. By direct calculations one verifies that

$$\ddot{V} = -2(\dot{e} + \Lambda_2\zeta)^\top K(\ddot{e} + \Lambda_2 e) - 2e^\top \Lambda_1 K \Lambda_1 \dot{e}.$$

By using Assumption 4.1, the properties 2.1, 2.3, and the fact that the variables $e, \xi, s, \tilde{\pi}$ are bounded because of (A.3), one deduces that $\ddot{V}$ is bounded. Note that (A.3) is the sum of two positive elements, thus their sum tends to zero implies that the error $e(t)$ converges to zero.

A.2 PROOF OF THEOREM 4.1

Thanks to the formulation of the control result of Lemma 4.1, this proof is similar to that of A.1. In particular, reconsider the candidate Lyapunov function (A.1) with

$$\tilde{\xi}_c := \xi_c - \dot{r}$$

instead of $\tilde{\xi}$. In view of Properties 2.2, 4.1 and the partitioning (4.9), the application of the controls (4.10)-(4.11) renders the time derivative of V as follows:

$$\dot{V} = -s^\top \begin{pmatrix} Y_n(q, \dot{q}, \xi, \dot{\xi})\hat{\pi} \\ K s_c \end{pmatrix} + 2e^\top K \Lambda_1 \dot{e} + 2(\dot{r} - \Lambda_1 e - \xi_c)^\top \Lambda_1 K e.$$

Given the Property 2.1, note that the auxiliary control input $\dot{\xi}_n$ is well defined in a neighbourhood of $\tilde{\pi} = 0_p$ since each leading principal minor of the mass matrix $M(q, \pi)$ is invertible when $\hat{\pi}$ belongs to a neighbourhood of π. As a consequence, the choice of the auxiliary control input $\dot{\xi}_n$ in (4.10b) implies that

$$\widehat{M}_n \dot{\xi}_n + Y_n \left(q, \dot{q}, \xi, \begin{pmatrix} 0_k \\ \dot{\xi}_c \end{pmatrix} \right) \hat{\pi} = Y_n(q, \dot{q}, \xi, \dot{\xi}) \hat{\pi} = K_n s_n.$$

In view of (4.9) and of the above equation, the expression of $\dot{V}$ in (A.4) becomes

$$\dot{V} = -s_n^T K_n s_n - s_c K s_c + 2e^\top K \Lambda_1 \dot{e} + 2(\dot{r} - \Lambda_1 e - \xi_c)^\top \Lambda_1 K e.$$

Analogously to the proof of Lemma 4.1, the variable $\xi_c(t)$ can be obtained by integration, i.e.

$$\xi_c(t) = \dot{r} - \Lambda_1 e - \Lambda_2 \int_0^t e(z) dz - \alpha = \dot{r} - \Lambda_1 e - \Lambda_2 \zeta,$$

with the collocated error e given by (4.6). Now, by substituting

$$s_c = \dot{q}_c - \xi_c = \dot{e} + \Lambda_1 e + \Lambda_2 \zeta$$

in the second term on the right hand side of $\dot{V}$ one obtains

$$\dot{V} = -s_n^T K_n s_n - (\dot{e} + \Lambda_2 \zeta)^\top K (\dot{e} + \Lambda_2 \zeta) - e^\top \Lambda_1 K \Lambda_1 e \leqslant 0. \quad \text{(A.4)}$$

Then, the stability of the equilibrium point

$$(e, \tilde{\xi}_c, s, \tilde{\pi}) = (0_m, 0_m, 0_n, 0_p)$$

follows, which clearly implies the boundedness of the system trajectories when the initial conditions belong to a neighbourhood of the equilibrium point.

Given the Assumption 4.1, the properties 2.1, 2.3, and the assumption that the noncollocated velocity $\dot{q}_n$ remains bounded, it is possible to verify that $\ddot{V}$ is bounded when the initial conditions belong to a neighbourhood of the equilibrium point. Then, $\dot{V}$ is uniformly continuous, and analogously to the proof of Lemma 4.1, one shows that $e(t)$ converges to zero.

A.3 PROOF OF LEMMA 4.2

First, define

$$M_i(q, \pi) := S_i M(q, \pi) S_i^\top \in \mathbb{R}^{i \times i},$$

with S_i given by (4.13), as the symmetric positive definite leading minor of order i of the mass matrix $M(q, \pi)$. Then, observe that:

$$\begin{aligned} M(q, \pi)\ddot{q} &:= Y_M(q, \ddot{q})\pi \\ &= [Y(q, 0_n, 0_n, \ddot{q}) - Y(q, 0_n, 0_n, 0_n)]\pi. \end{aligned} \quad \text{(A.5)}$$

Let $e_j \in \mathbb{R}^i$ denote the vector of i zeros except for the j*th* coordinate, which is equal to one. By using the Jacobi's formula[1], one has

$$\begin{aligned}
\partial_\pi \det(M_i) &= \det(M_i)\mathrm{tr}(M_i^{-1}\partial_\pi M_i) \\
&= \det(M_i)\mathrm{tr}(M_i^{-1}S_i\partial_\pi M S_i^\top) \\
&= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i\partial_\pi M S_i^\top e_j \\
&= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i\ \partial_\pi M \begin{pmatrix} e_j \\ 0_{n-i} \end{pmatrix}.
\end{aligned}$$

Then, in view of (A.5), one obtains

$$\partial_\pi \det(M_i) = \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i Y_M\left(q, \begin{pmatrix} e_j \\ 0_{n-i} \end{pmatrix}\right).$$

Consequently,

$$\begin{aligned}
\partial_\pi \det(M_i)\ \pi &= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i Y_M\left(q, \begin{pmatrix} e_j \\ 0_{n-i} \end{pmatrix}\right)\pi \\
&= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i M\begin{pmatrix} e_j \\ 0_{n-i} \end{pmatrix} \\
&= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}S_i M S_i^\top e_j \\
&= \det(M_i)\sum_{j=1}^{i} e_j^\top M_i^{-1}M_i e_j \\
&= i\det(M_i).
\end{aligned}$$

Since the inertia matrix M is positive definite, then

$$\det(M_i) > 0\ \forall\ i = \{1, \ldots, n\}.$$

This in turn implies that

$$\exists\ \gamma > 0 \text{ such that } \left|\partial_\pi \det\left(S_i M(q,\pi)S_i^\top\right)\right| > \gamma.$$

1 Although the Jacobi's formula is usually applied to a single-parameter dependent matrix, it is possible to verify that the above application to a multi-variable dependent matrix is correct.

A.4 PROOF OF PROPOSITION 4.1

Proof of *i)*. If

$$\det(\widehat{M}_n) > 0,$$

then the matrix $\widehat{M}_n^{-1}$ exists. Note that $\widehat{M}_n$ is symmetric by construction. Then, analogously to the proof of the Lemma 4.2, multiplying Eq. (4.18b) times $\hat{\pi}^\top$ yields:

$$\hat{\pi}^\top \delta = \sum_{i=1}^{k} \hat{\pi}^\top Y_{M_n}^\top(q, e_i)\widehat{M}_n^{-1} e_i = \sum_{i=1}^{k} e_i^\top \widehat{M}_n^\top \widehat{M}_n^{-1} e_i = k.$$

Since the system is underactuated, then $k \geqslant 1$. Consequently, $|\delta| > 0$ and $|\hat{\pi}| > 0$.

Proof of *ii)*. Consider the following storage function

$$V_d := \frac{1}{2}\det^2(\widehat{M}_n).$$

It is possible to verify that the time derivative of V_d is:

$$\dot{V}_d = \det^2(\widehat{M}_n)\operatorname{tr}\left(\widehat{M}_n^{-1}\dot{\widehat{M}}_n\right).$$

Let's consider now the time derivative of $\widehat{M}_n$.

$$\begin{aligned}
\dot{\widehat{M}}_n &= \left[\cdots, \dot{\hat{m}}_n^{(i)}(q, \hat{\pi}), \cdots\right] \\
&= \left[\cdots, \dot{\widehat{M}}_n e_i, \cdots\right] \\
&= \left[\cdots, \frac{\partial}{\partial q}(\widehat{M}_n e_i)\dot{q} + \frac{\partial}{\partial \pi}(\widehat{M}_n e_i)\dot{\hat{\pi}}, \cdots\right] \\
&= \left[\cdots, \frac{\partial}{\partial q}\left(Y_{M_n}(q, \begin{pmatrix} e_i \\ 0_m \end{pmatrix})\hat{\pi}\right)\dot{q} + Y_{M_n}(q, \begin{pmatrix} e_i \\ 0_m \end{pmatrix})\dot{\hat{\pi}}, \cdots\right] \\
&= \left[\cdots, \upsilon_i + \eta Y_{M_n}\Gamma\delta, \cdots\right]
\end{aligned}$$

where $\hat{m}_n^{(i)} = \widehat{M}_n e_i$ is the ith column of $\widehat{M}_n$, η is given by Eq (4.18a), υ by Eq (4.19c) and where we used the adaptation law by Eq (4.17).

By plugging the above equation into $\dot{V}_d$ we obtain:

$$\begin{aligned}
V_d &= \det^2(\widehat{M}_n)\operatorname{tr}\left(\widehat{M}_n^{-1}\left[\cdots, \upsilon_i + \eta Y_{M_n}\Gamma\delta, \cdots\right]\right) \\
&= \det^2(\widehat{M}_n)\left[\operatorname{tr}\left(\widehat{M}_n^{-1}\Upsilon\right) + \eta\sum_{i=1}^{k} e_i^\top \widehat{M}_n^{-1} Y_{M_n}\Gamma\delta\right] \\
&= \det^2(\widehat{M}_n)\left[\operatorname{tr}\left(\widehat{M}_n^{-1}\Upsilon\right) + \eta\delta^\top\Gamma\delta\right],
\end{aligned}$$

where Υ is given by Eq (4.19b) and δ is given by Eq (4.18). If

$$\det\left(\widehat{M}_n\right) \leqslant \varepsilon,$$

then one obtains that $\dot{V}_d \geqslant 0$ in view of the definition of η. As a consequence

$$\det(\widehat{M}_n) \geqslant \varepsilon \quad \forall t \quad \text{if} \quad \det\left(\widehat{M}_n\right)(0) > \varepsilon.$$

A.5 PROOF OF LEMMA 8.1

As described in Section 8.2 the proof is based first on the linearization of the constrained closed loop dynamics around the equilibrium $(q_j^d, 0)$ and then by means of Lyapunov analysis on the resultant linear system.

Linearization

Assume that Assumption 8.1 holds, and that we apply the control laws (8.1) with the gains as in Eq. (8.3). In particular, decompose the control variable φ in Eq. (8.1) as in the following:

$$\varphi = h_j - J_j^\top f - v$$

with

$$v := K_p^q(q_j - q_j^d) + K_d^q \dot{q}_j.$$

Note that Assumption 8.1 implies that the base velocity $\nu_{\mathcal{B}}$ can be written as a function of the joint velocity $\dot{q}_j$. Indeed, consider Eq. (2.18) decomposed in the base and joint submatrices, i.e.

$$\begin{bmatrix} J_b & J_j \end{bmatrix} \begin{bmatrix} \nu_{\mathcal{B}} \\ \dot{q}_j \end{bmatrix} = 0 \,.$$

We can obtain the expression of the base velocity as

$$\nu_{\mathcal{B}} = -J_b^{-1} J_j \dot{q}_j.$$

The closed loop joint space dynamics of system (7.10) constrained by (2.19) is given by the following equation:

$$\ddot{q}_j = -M_j^{-1}\left[\Lambda^\dagger(J_b M_b^{-1}\dot{H}^d + \Gamma\dot{q}_j) + N_\tau v\right] \tag{A.6}$$

where

$$\Gamma = \dot{J}_j - J_b M_b^{-1} C_{bj} + (J_b M_b^{-1} C_b - \dot{J}_b) J_b^{-1} J_j$$

and where we defined

$$\Lambda = J_j M_j^{-1}.$$

Define the joint error and its dynamics:

$$\tilde{q}_j = q_j - q_j^d \,, \quad \dot{\tilde{q}}_j = \dot{q}_j \,, \quad \ddot{\tilde{q}}_j = \ddot{q}_j \,.$$

The equilibrium point of the new state variables $(\tilde{q}_j, \dot{\tilde{q}}_j)$ corresponds thus to the origin $(0,0)$. We can now proceed to write Eq.(A.6) in state space form. Define the state x as

$$x := \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \tilde{q}_j \\ \dot{\tilde{q}}_j \end{bmatrix} .$$

Given $H^r \equiv 0$ the linearized dynamical system about the equilibrium point $(0,0)$ is given by

$$\dot{x} = \begin{bmatrix} \partial_{\tilde{q}_j} \dot{x}_1 & \partial_{\dot{\tilde{q}}_j} \dot{x}_1 \\ \partial_{\tilde{q}_j} \dot{x}_2 & \partial_{\dot{\tilde{q}}_j} \dot{x}_2 \end{bmatrix} x = \begin{bmatrix} 0_{n\times n} & 1_n \\ A_1 & A_2 \end{bmatrix} x \tag{A.7}$$

To compute the last row of the state matrix we have to consider the partial derivatives of Eq. (A.6). First, notice that

$$\partial_{\tilde{q}_j} (\Gamma \dot{\tilde{q}}_j)\Big|_{\substack{\tilde{q}_j=0 \\ \dot{\tilde{q}}_j=0}} = 0 \,, \quad \partial_{\dot{\tilde{q}}_j} (\Gamma \dot{\tilde{q}}_j)\Big|_{\substack{\tilde{q}_j=0 \\ \dot{\tilde{q}}_j=0}} = 0.$$

We are now left to compute

$$\begin{aligned} \partial_{\tilde{q}_j} \ddot{\tilde{q}}_j = &- \sum_{i=1}^{6} \partial_{\tilde{q}_j} (M_j^{-1} \Lambda^\dagger J_b M_b^{-1} e_i) e_i^\top \dot{H}^d - M_j^{-1} N_\tau \partial_{\tilde{q}_j} v \\ &- \sum_{i=1}^{6} \partial_{\tilde{q}_j} (M_j^{-1} N_\tau e_i) e_i^\top v - M_j^{-1} \Lambda^\dagger J_b M_b^{-1} \partial_{\tilde{q}_j} \dot{H}^d \end{aligned}$$

and

$$\begin{aligned} \partial_{\dot{\tilde{q}}_j} \ddot{\tilde{q}}_j = &- \sum_{i=1}^{6} \partial_{\dot{\tilde{q}}_j} (M_j^{-1} \Lambda^\dagger J_b M_b^{-1} e_i) e_i^\top \dot{H}^d - M_j^{-1} N_\tau \partial_{\dot{\tilde{q}}_j} v \\ &- \sum_{i=1}^{6} \partial_{\dot{\tilde{q}}_j} (M_j^{-1} N_\tau e_i) e_i^\top v - M_j^{-1} \Lambda^\dagger J_b M_b^{-1} \partial_{\dot{\tilde{q}}_j} \dot{H}^d. \end{aligned}$$

Note that $\dot{H}^d = 0$ and $v = 0$ when evaluated in $\tilde{q}_j = 0$ and $\dot{\tilde{q}}_j = 0$. We thus have to compute only the partial derivatives of $\dot{H}^d$ and v. The latter is trivially given by

$$\begin{aligned} \partial_{\tilde{q}_j} v &= \bar{K}_p^j N_\tau M_j \\ \partial_{\dot{\tilde{q}}_j} v &= \bar{K}_d^j N_\tau M_j. \end{aligned}$$

We now compute $\partial_{\tilde{q}_j}\dot{H}^d$ and $\partial_{\dot{\tilde{q}}_j}\dot{H}^d$. Noting that $H = M_b v_{\mathcal{B}}$ it is easy to verify that

$$\begin{aligned}
\partial_{\tilde{q}_j}\dot{H}^d &= -\partial_{\tilde{q}_j} K_i \int_0^t H\,ds = -\partial_{\tilde{q}_j} K_i \int_0^t M_b v_{\mathcal{B}}\,ds \\
&= \partial_{\tilde{q}_j} K_i \int_0^t M_b J_b^{-1} J_j \dot{\tilde{q}}_j\,ds \\
&= \partial_{\tilde{q}_j} K_i \int_{\tilde{q}_j(0)}^{\tilde{q}_j(t)} M_b J_b^{-1} J_j\,d\tilde{q}_j \\
&= K_i M_b J_b^{-1} J_j, \\
\partial_{\dot{\tilde{q}}_j}\dot{H}^d &= -\partial_{\dot{\tilde{q}}_j} K_p H - \partial_{\dot{\tilde{q}}_j} K_i \int_0^t H\,ds \\
&= K_p M_b J_b^{-1} J_j
\end{aligned}$$

which must then be evaluated in $(\tilde{q}_j, \dot{\tilde{q}}_j) = (0, 0)$.

Grouping the partial derivatives together, i.e.

$$\begin{aligned}
A_1 &:= -M_j^{-1}\Lambda^\dagger J_b M_b^{-1} K_i M_b J_b^{-1} J_j - M_j^{-1} N_\tau \bar{K}_p^j N_\tau M_j \\
A_2 &:= -M_j^{-1}\Lambda^\dagger J_b M_b^{-1} K_p M_b J_b^{-1} J_j - M_j^{-1} N_\tau \bar{K}_d^j N_\tau M_j
\end{aligned}$$

we obtain the expression for the matrices in (A.7).

Proof of Asymptotic Stability

Consider now the following Lyapunov candidate:

$$V(x) = \frac{1}{2} x_1^\top M_j^\top(0) Q_1 M_j(0) x_1 + \frac{1}{2} x_2^\top M_j^\top(0) Q_2 M_j(0) x_2$$

where

$$\begin{aligned}
Q_1 &:= \Lambda^\top J_b^{-\top} M_b^\top K_i M_b J_b^{-1} \Lambda + N_\tau \bar{K}_p^j N_\tau \Big|_{\substack{x_1=0 \\ x_2=0}} \\
Q_2 &:= \Lambda^\top J_b^{-\top} M_b^\top M_b J_b^{-1} \Lambda + N_\tau \Big|_{\substack{x_1=0 \\ x_2=0}}
\end{aligned}$$

V is a properly defined candidate, in fact $V = 0 \iff x = 0$ and is positive definite otherwise. Indeed, Q_1 can be rewritten in the following way:

$$Q_1 = \begin{bmatrix} \Lambda^\top & N_\tau \end{bmatrix} \begin{bmatrix} J_b^{-\top} M_b^\top K_i M_b J_b^{-1} & 0 \\ 0 & \bar{K}_p^j \end{bmatrix} \begin{bmatrix} \Lambda \\ N_\tau \end{bmatrix}.$$

and, because Λ and N_τ are orthogonal Q_1 is positive definite. The same reasoning can be applied to Q_2.

We can consider now the time derivative of V:

$$\begin{aligned}\dot{V} &= x_1^\top M_j^\top Q_1 M_j x_2 + x_2^\top M_j^\top Q_2 M_j \dot{x}_2 \\ &= x_1^\top M_j^\top Q_1 M_j x_2 + x_2^\top M_j^\top Q_2 M_j (A_1 x_1 + A_2 x_2) \\ &= \underbrace{x_1^\top M_j^\top Q_1 M_j x_2}_{\dot{V}_1} \\ &\quad \underbrace{-x_2^\top M_j^\top Q_2 (\Lambda^\dagger J_b M_b^{-1} K_i M_b J_b^{-1} J_j + N_\tau \bar{K}_p^j N_\tau M_j) x_1}_{\dot{V}_2} \\ &\quad \underbrace{-x_2^\top M_j^\top Q_2 (\Lambda^\dagger J_b M_b^{-1} K_p M_b J_b^{-1} J_j + N_\tau \bar{K}_d^j N_\tau M_j) x_2}_{\dot{V}_3}.\end{aligned}$$

Examining now the three terms separately:

$$\begin{aligned}\dot{V}_1 &= x_1^\top M_j^\top \Lambda^\top J_b^{-\top} M_b^\top K_i M_b J_b^{-1} \Lambda M_j x_2 + x_1^\top M_j^\top N_\tau \bar{K}_p^j N_\tau M_j x_2, \\ \dot{V}_2 &= -x_2^\top M_j^\top \Lambda^\top J_b^{-\top} M_b^\top K_i M_b J_b^{-1} J_j x_1 - x_2^\top M_j^\top N_\tau \bar{K}_p^j N_\tau M_j x_1 \\ &= -\dot{V}_1 \\ \dot{V}_3 &= -x_2^\top M_j^\top Q_2 (\Lambda^\dagger J_b M_b^{-1} K_p M_b J_b^{-1} J_j + N_\tau \bar{K}_d^j N_\tau M_j) x_2 \\ &= -x_2^\top M_j^\top \Lambda^\top J_b^{-\top} M_b^\top K_p M_b J_b^{-1} \Lambda M_j x_2 - x_2^\top M_j^\top N_\tau \bar{K}_d^j N_\tau M_j x_2 \\ &= -x_2^\top M_j^\top (\Lambda^\top J_b^{-\top} M_b^\top K_p M_b J_b^{-1} \Lambda + N_\tau \bar{K}_d^j N_\tau) M_j x_2.\end{aligned}$$

Thus one obtains

$$\dot{V} = \dot{V}_3 \leqslant 0$$

and the stability of the equilibrium follows.

In order to prove the asymptotic stability of the equilibrium we have to resort to the LaSalle's invariance principle. Let's define the invariant set

$$S := \{x : \dot{V}(x) = 0\}$$

which implies $S = \{(x_1, 0)\}$. It is easy to verify that the only trajectory which starts in S and remains in S is given by $x_1 = 0$ thus proving the LaSalle's principle. As a consequence the equilibrium point

$$x = 0 \quad \Rightarrow \quad (\tilde{q}_j, \dot{\tilde{q}}_j) = (0, 0)$$

is asymptotically stable.

BIBLIOGRAPHY

[1] Jürgen Ackermann. *Robust control: Systems with uncertain physical parameters*. Springer Science & Business Media, 2012.

[2] Brian D O Anderson. Failures of adaptive control theory and their resolution. *Communications in Information and Systems*, 5:1–20, 2005.

[3] Brian D. O. Anderson and John B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1990. ISBN 0-13-638560-5.

[4] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000. ISBN 1886529094.

[5] J. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, second edition, 2010. doi: 10.1137/1.9780898718577. URL http://epubs.siam.org/doi/abs/10.1137/1.9780898718577.

[6] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[7] RW Brockett. Asymptotic Stability And Feedback Stabilization. *Differential Geometric Control Theory*, 1983. doi: 10.1.1.324.9912.

[8] Giorgio Cannata, M. Maggiali, G. Metta, and G. Sandini. An embedded artificial skin for humanoid robots. In *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pages 434–438, Aug 2008. doi: 10.1109/MFI.2008.4648033.

[9] Hong Chen and Frank Allgower. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. In *Control Conference (ECC), 1997 European*, pages 1421–1426. IEEE, 1997.

[10] Stefano Chiaverini, Bruno Siciliano, and Olav Egeland. Review of the Damped Least-Squares Inverse Kinematics with Experiments on an Industrial Robot Manipulator.

Control Systems Technology, IEEE Transactions on, 2(2):123–134, 1994.

[11] H. Dallali, M. Mosadeghzad, G.A Medrano-Cerda, N. Docquier, P. Kormushev, N. Tsagarakis, Zhibin Li, and D. Caldwell. Development of a dynamic simulator for a compliant humanoid robot based on a symbolic multibody approach. In *Mechatronics (ICM), 2013 IEEE International Conference on*, pages 598–603, Feb 2013.

[12] Martin De Lasa and Aaron Hertzmann. Prioritized optimization for task-space control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5755–5762. IEEE, October 2009.

[13] Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. *ACM Trans. Graph.*, 29(4):131:1–131:10, July 2010. ISSN 0730-0301. doi: 10.1145/1778765.1781157. URL http://doi.acm.org/10.1145/1778765.1781157.

[14] A. De Luca, R. Mattone, and G. Oriolo. Dynamic mobility of redundant robots using end-effector commands. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 2, pages 1760–1767 vol.2, Apr 1996. doi: 10.1109/ROBOT.1996.506967.

[15] Alessandro De Luca and Giuseppe Oriolo. Motion Planning and Trajectory Control of an Underactuated Three-Link Robot via Feedback Linearization. *Proceedings of IEEE International Conference on Robotics and Automation*, 2000.

[16] G De Nicolao, L Magni, and R Scattolini. Stability and robustness of nonlinear receding horizon control. In *Nonlinear model predictive control*, pages 3–22. Springer, 2000.

[17] A. Del Prete, F. Romano, L. Natale, G. Metta, G. Sandini, and F. Nori. Prioritized optimal control. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2540–2545, May 2014. doi: 10.1109/ICRA.2014.6907214.

[18] Moritz Diehl, H Georg Bock, Johannes P Schlöder, Rolf Findeisen, Zoltan Nagy, and Frank Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[19] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, page 0278364914521306, 2014.

[20] Hanno Essén. Average angular velocity. *European journal of physics*, 14(5):201, 1993.

[21] Siyuan Feng, X. Xinjilefu, C.G. Atkeson, and Joohyung Kim. Optimization based controller design and implementation for the atlas robot in the darpa robotics challenge finals. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 1028–1035, Nov 2015. doi: 10.1109/HUMANOIDS.2015.7363480.

[22] L. Fiorio, F. Romano, A. Parmiggiani, G. Sandini, and F. Nori. On the effects of internal stiction in pnrVIA actuators. In *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, pages 362–367, Oct 2013. doi: 10.1109/HUMANOIDS.2013.7030000.

[23] L. Fiorio, F. Romano, A. Parmiggiani, G. Sandini, and F. Nori. Stiction compensation in agonist-antagonist variable stiffness actuators. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[24] Matteo Fumagalli, Serena Ivaldi, Marco Randazzo, Lorenzo Natale, Giorgio Metta, Giulio Sandini, and Francesco Nori. Force feedback exploiting tactile and proximal force/torque sensing. *Autonomous Robots*, 33 (4):381–398, 2012. ISSN 1573-7527. doi: 10.1007/s10514-012-9291-2. URL http://dx.doi.org/10.1007/s10514-012-9291-2.

[25] Gianluca Garofalo, Bernd Henze, Johannes Englsberger, and Christian Ott. On the inertially decoupled structure of the floating base robot dynamics. *IFAC-PapersOnLine*, 48(1):322–327, 2015.

[26] J. Ghommam, F. Mnif, and N. Derbel. Global stabilisation and tracking control of underactuated surface vessels. *IET Control Theory & Applications*, 4(1):71–88, January 2010. ISSN 1751-8644. doi: 10.1049/iet-cta.2008.0131.

[27] Robert H. Goddard. *A Method of Reaching Extreme Altitudes*, volume 71 of *Smithsonian Miscellaneous Collections*. Smithsonian Institution, 1919.

[28] J.W. Grizzle, C.H. Moog, and C. Chevallereau. Nonlinear control of mechanical systems with an unactuated cyclic variable. *Automatic Control, IEEE Transactions on*, 50(5): 559–576, May 2005. ISSN 0018-9286. doi: 10.1109/TAC.2005.847057.

[29] Y.-L. Gu and Y. Xu. Under-actuated robot systems: dynamic interaction and adaptive control. In *Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on*, volume 1, pages 958–963 vol.1, Oct 1994. doi: 10.1109/ICSMC.1994.399960.

[30] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur, and Moritz Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010. doi: 10.1163/016918610X493552. URL http://dx.doi.org/10.1163/016918610X493552.

[31] Hugh Herr and Marko Popovic. Angular momentum in human walking. *Journal of Experimental Biology*, 211(4): 467–481, 2008. ISSN 0022-0949. doi: 10.1242/jeb.008573. URL http://jeb.biologists.org/content/211/4/467.

[32] A. Herzog, L. Righetti, F. Grimminger, P. Pastor, and S. Schaal. Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 981–988, Sept 2014. doi: 10.1109/IROS.2014.6942678.

[33] M.A. Hopkins, R.J. Griffin, A. Leonessa, B.Y. Lattimer, and T. Furukawa. Design of a compliant bipedal walking controller for the darpa robotics challenge. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 831–837, Nov 2015. doi: 10.1109/HUMANOIDS.2015.7363450.

[34] Minh-Duc Hua. *Contributions to the automatic control of aerial vehicles*. PhD thesis, Ph. D. thesis, University of Nice-Sophia Antipolis, 2009.

[35] S. Hyon, J.G. Hale, and G. Cheng. Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces. *Robotics, IEEE Transactions on*, 23(5):884–898, Oct 2007. ISSN 1552-3098. doi: 10.1109/TRO.2007.904896.

[36] IEEE-RAS. Whole-body control ras technical committee. URL http://www.ieee-ras.org/whole-body-control.

[37] Alberto Isidori. *Nonlinear Control Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 1995. ISBN 3540199160.

[38] Alberto Isidori. The zero dynamics of a nonlinear system: From the origin to the latest progresses of a long successful story. *European Journal of Control*, 19(5):369 – 378, 2013. ISSN 0947-3580. doi: http://dx.doi.org/10.1016/j.ejcon.2013.05.014. URL http://www.sciencedirect.com/science/article/pii/S0947358013000836. The Path of Control.

[39] D.H. Jacobson and D.Q. Mayne. *Differential dynamic programming*. Modern analytic and computational methods in science and mathematics. American Elsevier Pub. Co., 1970.

[40] Julius Jellinek and DH Li. Separation of the energy of overall rotation in any n-body system. *Physical review letters*, 62(3):241, 1989.

[41] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 239–246 vol.1, 2001. doi: 10.1109/IROS.2001.973365.

[42] 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 *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1644–1650 vol.2, Oct 2003. doi: 10.1109/IROS.2003.1248880.

[43] Oussama Kanoun, Florent Lamiraux, and Pierre-Brice Wieber. Kinematic control of redundant manipulators: Generalizing the Task-Priority framework to inequality task. *IEEE Transactions on Robotics*, 27(4), 2011.

[44] S.S. Keerthi and Elmer G Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations. *Journal of optimization theory and applications*, 57(2):265–293, 1988.

[45] H.K. Khalil. *Nonlinear Systems*. Pearson Education. Prentice Hall, 2002. ISBN 9780130673893. URL https://books.google.it/books?id=t_d1QgAACAAJ.

[46] Wisama Khalil and E Dombre. *Modeling, identification and control of robots*. Butterworth-Heinemann, 2004. ISBN 0080536611.

[47] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43–53, February 1987. ISSN 0882-4967. doi: 10.1109/JRA.1987.1087068.

[48] C Kirches, L Wirsching, HG Bock, and JP Schlöder. Efficient direct multiple shooting for nonlinear model predictive control on long horizons. *Journal of Process Control*, 22 (3):540–550, 2012.

[49] Donald E Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2012.

[50] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard. Whole-body model-predictive control applied to the hrp-2 humanoid. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3346–3351, Sept 2015. doi: 10.1109/IROS.2015.7353843.

[51] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, pages 2149 – 2154, 2004.

[52] Twan Koolen, Sylvain Bertrand, Gray Thomas, Tomas de Boer, Tingfan Wu, Jesper Smith, Johannes Englsberger, and J Pratt. Design of a momentum-based control framework and application to the humanoid robot atlas. *preparation for International Journal of Humanoid Robotics*, 2015.

[53] S. Kuindersma, F. Permenter, and R. Tedrake. An efficiently solvable quadratic program for stabilizing dynamic locomotion. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2589–2594, May 2014. doi: 10.1109/ICRA.2014.6907230.

[54] Yoan D. Landau. *Adaptive control – the Model reference approach*, volume 8 of *Control and System Theory*. Dekker, 1979. ISBN 0-8247-6548-6.

[55] Sung-Hee Lee and A. Goswami. Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3157–3162, Oct 2010. doi: 10.1109/IROS.2010.5650416.

[56] Daniel B Leineweber, Irene Bauer, Hans Georg Bock, and Johannes P Schlöder. An efficient multiple shooting based reduced sqp strategy for large-scale dynamic process optimization. part 1: theoretical aspects. *Computers & Chemical Engineering*, 27(2):157–166, 2003.

[57] AA Maciejewski. Dealing with the ill-conditioned equations of motion for articulated figures. *Computer Graphics and Applications, IEEE*, 10(3):63–71, May 1990. ISSN 0272-1716.

[58] P. Maiolino, M. Maggiali, G. Cannata, G. Metta, and L. Natale. A flexible and robust large scale capacitive tactile system for robots. *Sensors Journal, IEEE*, 13(10):3910–3917, Oct 2013. ISSN 1530-437X. doi: 10.1109/JSEN.2013.2258149.

[59] Jerrold E. Marsden and Tudor S. Ratiu. *Introduction to Mechanics and Symmetry: A Basic Exposition of Classical Mechanical Systems*. Springer Publishing Company, Incorporated, 2010. ISBN 1441931430, 9781441931436.

[60] Jerrold E Marsden and Jürgen Scheurle. The reduced euler-lagrange equations. *Fields Institute Comm*, 1:139–164, 1993.

[61] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 1966.

[62] D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *Automatic Control, IEEE Transactions on*, 35(7):814–824, Jul 1990. ISSN 0018-9286. doi: 10.1109/9.57020.

[63] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789 – 814, 2000. ISSN 0005-1098. doi: http://dx.doi.org/10.1016/S0005-1098(99)00214-9. URL http://www.sciencedirect.com/science/article/pii/S0005109899002149.

[64] Giorgio Metta, Paul Fitzpatrick, and Lorenzo Natale. Yarp: yet another robot platform. *International Journal on Advanced Robotics Systems*, 3(1):43–48, 2006.

[65] Giorgio Metta, Lorenzo Natale, Francesco Nori, Giulio Sandini, David Vernon, Luciano Fadiga, Claes von Hofsten, Kerstin Rosander, Manuel Lopes, José Santos-Victor, Alexandre Bernardino, and Luis Montesano. The iCub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8–9):1125 – 1134, 2010. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/j.neunet.2010.08.010. URL http://www.sciencedirect.com/science/article/pii/S0893608010001619. Social Cognition: From Babies to Robots.

[66] E. Mingo, S. Traversaro, A. Rocchi, M. Ferrati, A. Settimi, F. Romano, L. Natale, A. Bicchi, F. Nori, and N.G. Tsagarakis. Yarp based plugins for gazebo simulator. In *2014 Modelling and Simulation for Autonomous Systems Workshop (MESAS)*, Roma, Italy, 5 -6 May 2014, 2014.

[67] Michael Mistry and Ludovic Righetti. Operational Space Control of Constrained and Underactuated Systems. In *Robotics: Science and Systems*, June 2011.

[68] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph.*, 31(4):43:1–43:8, July 2012. ISSN 0730-0301. doi: 10.1145/2185520.2185539. URL http://doi.acm.org/10.1145/2185520.2185539.

[69] Richard M. Murray, S. Shankar Sastry, and Li Zexiang. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1994. ISBN 0849379814.

[70] Yoshihiko Nakamura, H. Hanafusa, and T. Yoshikawa. Task-Priority Based Redundancy Control of Robot Manipulators. *The International Journal of Robotics Research*, 6(2):3–15, June 1987.

[71] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[72] F. Nori, S. Traversaro, J. Eljaik, F. Romano, A. Del Prete, and D. Pucci. iCub Whole-body Control through Force Regulation on Rigid Noncoplanar Contacts. *Frontiers in*

Robotics and AI, 2(6), 2015. ISSN 2296-9144. doi: 10.3389/frobt.2015.00006.

[73] Henrik Olsson, Karl J Åström, C Canudas De Wit, Magnus Gäfvert, and Pablo Lischinsky. Friction models and friction compensation. *European journal of control*, 4(3): 176–195, 1998.

[74] D. E. Orin and A. Goswami. Centroidal momentum matrix of a humanoid robot: Structure and properties. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 653–659, Sept 2008. doi: 10.1109/IROS.2008.4650772.

[75] DavidE. Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35(2-3):161–176, 2013. ISSN 0929-5593. doi: 10.1007/s10514-013-9341-4. URL http://dx.doi.org/10.1007/s10514-013-9341-4.

[76] C. Ott, M.A. Roa, and G. Hirzinger. Posture and balance control for biped robots based on contact force optimization. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 26–33, Oct 2011. doi: 10.1109/Humanoids.2011.6100882.

[77] Christian Ott, Alexander Dietrich, and Alin Albu-Schäffer. Prioritized multi-task compliance control of redundant manipulators. *Automatica*, 53:416 – 423, 2015. ISSN 0005-1098. doi: http://dx.doi.org/10.1016/j.automatica.2015.01.015. URL http://www.sciencedirect.com/science/article/pii/S0005109815000163.

[78] Mun-Soo Park and Dongkyoung Chwa. Swing-up and stabilization control of inverted-pendulum systems via coupled sliding-mode control method. *Industrial Electronics, IEEE Transactions on*, 56(9):3541–3555, 2009.

[79] Ugo Pattacini, Francesco Nori, Lorenzo Natale, Giorgio Metta, and Giulio Sandini. An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots. In *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, pages 1668–1674. IEEE, 2010.

[80] Jan Peters, Michael Mistry, Firdaus E. Udwadia, Jun Nakanishi, and Stefan Schaal. A unifying framework for robot control with redundant DOFs. *Autonomous Robots*, 24(1):1–12, October 2007.

[81] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture point: A step toward humanoid push recovery. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 200–207, Dec 2006. doi: 10.1109/ICHR.2006.321385.

[82] Andrea Del Prete, Francesco Nori, Giorgio Metta, and Lorenzo Natale. Prioritized motion–force control of constrained fully-actuated robots: "task space inverse dynamics". *Robotics and Autonomous Systems*, 63, Part 1:150 – 157, 2015. ISSN 0921-8890. doi: http://dx.doi.org/10.1016/j.robot.2014.08.016. URL http://www.sciencedirect.com/science/article/pii/S0921889014001742.

[83] D. Pucci, F. Romano, and F. Nori. Collocated adaptive control of underactuated mechanical systems. *Robotics, IEEE Transactions on*, 31(6):1527–1536, Dec 2015. ISSN 1552-3098. doi: 10.1109/TRO.2015.2481282.

[84] Zhihua Qu, Richard A. Hull, and Jing Wang. Globally Stabilizing Adaptive Control Design for Nonlinearly-Parameterized Systems. *IEEE Transactions on Automatic Control*, 51(6):1073–1079, 2006.

[85] Michael Yu Rachkov, Lino Marques, and AnÍbal T de Almeida. Multisensor demining robot. *Autonomous robots*, 18(3):275–291, 2005.

[86] Mahmut Reyhanoglu, Arjian van der Schaft, N. Harris McClamroch, and Ilya Kolmanovsky. Dynamics and control of a class of underactuated mechanical systems. *IEEE Transactions on Automatic Control*, 44(9):1663–1671, 1999.

[87] L. Righetti, J. Buchli, M. Mistry, and S. Schaal. Control of legged robots with optimal distribution of contact forces. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 318–324, Oct 2011. doi: 10.1109/Humanoids.2011.6100832.

[88] F. Romano, L. Fiorio, G. Sandini, and F. Nori. Control of a two-DoF manipulator equipped with a pnr-variable stiffness actuator. In *Intelligent Control (ISIC), 2014 IEEE International Symposium on*, pages 1354–1359, Oct 2014. doi: 10.1109/ISIC.2014.6967620.

[89] F. Romano, A. Del Prete, N. Mansard, and F. Nori. Prioritized optimal control: A hierarchical differential dynamic programming approach. In *Robotics and Automa-*

tion (ICRA), 2015 IEEE International Conference on, pages 3590–3595, May 2015. doi: 10.1109/ICRA.2015.7139697.

[90] Layale Saab, Nicolas Mansard, Francois Keith, J-Y. Fourquet, and Philippe Soueres. Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints. *IEEE International Conference on Robotics and Automation*, pages 1091–1096, May 2011.

[91] Claude Samson, Bernard Espiau, and Michel Le Borgne. *Robot Control: The Task Function Approach*. Oxford University Press, 1991. ISBN 0198538057.

[92] Raul Santiesteban, Thierry Floquet, Yury Orlov, Samer Riachy, and Jean-Pierre Richard. Second-order sliding mode control of underactuated mechanical systems ii: Orbital stabilization of an inverted pendulum with application to swing up/balancing control. *International Journal of Robust and Nonlinear Control*, 18(4-5):544–556, 2008. ISSN 1099-1239. doi: 10.1002/rnc.1203. URL http://dx.doi.org/10.1002/rnc.1203.

[93] S. Shankar Sastry and Alberto Isidori. Adaptive Control of Linearizable Systems. *IEEE Transactions on Automatic Control*, 34(11):1123–1131, 1989.

[94] Jon M Selig. *Geometric fundamentals of robotics*. Springer Science & Business Media, 2004.

[95] Luis Sentis. *Synthesis and control of whole-body behaviors in humanoid systems*. PhD thesis, Stanford University, 2007.

[96] J. Seyama and R. S. Nagayama. The uncanny valley: Effect of realism on the impression of artificial human faces. *Presence*, 16(4):337–351, Aug 2007. ISSN 1054-7460. doi: 10.1162/pres.16.4.337.

[97] A. Sherikov, D. Dimitrov, and P.-B. Wieber. Balancing a humanoid robot with a prioritized contact force distribution. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 223–228, Nov 2015. doi: 10.1109/HUMANOIDS.2015.7363555.

[98] B. Siciliano and J.-J.E. Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *IEEE Fifth International Conference on Advanced Robotics, 1991.*, pages 1211–1216 vol.2, June 1991.

[99] Jean-Jacques E. Slotine and Weiping Li. Adaptive manipulator control: A case study. *IEEE Transactions on Automatic Control*, 33(11):995–1003, 1988. ISSN 00189286. doi: 10.1109/9.14411.

[100] Mark W Spong. Energy based control of a class of underactuated mechanical systems. *1996 IFAC World Congress*, 1996.

[101] Mark W. Spong. *Control Problems in Robotics and Automation*, chapter Underactuated mechanical systems, pages 135–150. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. ISBN 978-3-540-40913-7. doi: 10.1007/BFb0015081. URL http://dx.doi.org/10.1007/BFb0015081.

[102] Mark W Spong, Romeo Ortega, and Rafael Kelly. Comments on "Adaptive Manipulator Control: A Case study". *IEEE Transactions on Automatic Control*, 35:761–762, 1990.

[103] B.J. Stephens and C.G. Atkeson. Dynamic balance force control for compliant humanoid robots. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1248–1255, Oct 2010. doi: 10.1109/IROS.2010.5648837.

[104] John Stuelpnagel. On the parametrization of the three-dimensional rotation group. *SIAM Review*, 6(4):422–430, 1964. doi: 10.1137/1006093. URL http://dx.doi.org/10.1137/1006093.

[105] Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4906–4913, Oct 2012. doi: 10.1109/IROS.2012.6386025.

[106] Valerie Thielemans, Pieter Meyns, and Sjoerd M. Bruijn. Is angular momentum in the horizontal plane during gait a controlled variable? *Human Movement Science*, 34:205 – 216, 2014. ISSN 0167-9457. doi: http://dx.doi.org/10.1016/j.humov.2014.03.003. URL http://www.sciencedirect.com/science/article/pii/S0167945714000359.

[107] E. Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005. Proceedings of the 2005*, pages 300–306 vol. 1, June 2005.

[108] Veeravalli Seshadri Varadarajan. *Lie Groups, Lie Algebras, and Their Representations*, volume 102. Springer Science & Business Media, 1984.

[109] Dipl Math Oskar Von Stryk. *Numerical solution of optimal control problems by direct collocation*. Springer, 1993.

[110] Miomir Vukobratović and Branislav Borovac. Zero-moment point—thirty five years of its life. *International Journal of Humanoid Robotics*, 1(01):157–173, 2004.

[111] Hanlei Wang. On adaptive inverse dynamics for free-floating space manipulators. *Robotics and Autonomous Systems*, 59(10):782 – 788, 2011. ISSN 0921-8890. doi: http://dx.doi.org/10.1016/j.robot.2011.05.013.

[112] F.W. Warner. *Foundations of Differentiable Manifolds and Lie Groups*. Graduate Texts in Mathematics. Springer, 1971. ISBN 9780387908946. URL https://books.google.it/books?id=iaeUqc2yQVQC.

[113] Patrick M. Wensing, Ghassan Bin Hammam, Behzad Dariush, and David E. Orin. Optimizing foot centers of pressure through force distribution in a humanoid robot. *International Journal of Humanoid Robotics*, 10(03):1350027, 2013. doi: 10.1142/S0219843613500278. URL http://www.worldscientific.com/doi/abs/10.1142/S0219843613500278.

[114] P.M. Wensing and D.E. Orin. Generation of dynamic humanoid behaviors through task-space control with conic optimization. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3103–3109, May 2013. doi: 10.1109/ICRA.2013.6631008.

[115] Koji Yoshida and Wisama Khalil. Verification of the Positive Definiteness of the Inertial Matrix of Manipulators Using Base Inertial Parameters. *The International Journal of Robotics Research*, 19(5):498–510, May 2000. ISSN 0278-3649. doi: 10.1177/02783640022066996.