Estimation-Based Control for Humanoid Robots

by

Nicholas Rotella

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements of the Degree
DOCTOR OF PHILOSOPHY
(Computer Science)

May 2018

Table of Contents

List of Figures

List of Tables

Abstract

As sensor, actuator and processor technology continues to improve, humanoid robots have become more common in both academic and industrial environments as general-purpose platforms capable of performing a wide variety of automated tasks. These robots have the potential to operate in complex environments built for humans; their form factor allows them to navigate buildings, utilize existing tools and perform cooperative tasks with humans, making them ideal for disaster recovery and household labor among many other applications. However, the challenge of operating autonomously in unknown environments involves obtaining accurate estimates of the robot's state by fusing information from onboard sensors and using these estimates for control in ways which allow robustness to uncertainty and disturbances. In this work, we propose methods for estimating important states of humanoid robots and evaluate the role of sensory information and state estimation in executing behaviors on a torque-controlled humanoid. First, we discuss estimation of the floating base pose of the robot in the world frame using only onboard sensing and kinematics models. This estimator takes into account contact switching and is shown to exhibit desirable observability characteristics through a nonlinear observability analysis. While the base pose and its velocity are crucial in computing joint torques via inverse dynamics, the configuration-dependent center of mass (COM) and its dynamics (the momentum of the system) is more-often the quantity of interest for planning and control. However, robot dynamic models are never perfect; we thus introduce estimators using the momentum dynamics of the robot with measured contact wrenches to estimate configuration-dependent center of mass and momentum offsets as well as external wrenches applied about the COM. We propose to evaluate the utility of this estimator for online replanning in a momentum model predictive control (MPC) framework, thereby indirectly using measured contact wrenches for feedback. Finally, we introduce methods for computing joint velocities and

accelerations from link-mounted IMUs and knowledge of kinematics, avoiding numerical differentiation of noisy joint angle sensors. This information is fused in several estimators and used to increase the maximum stable joint feedback gains; we propose to utilize inertial sensor-based estimates of joint derivatives in our whole-body control framework to achieve better Cartesian damping control and thus improve tracking and robustness in the system. These estimation methods are discussed and evaluated in the context of optimization-based inverse dynamics control for locomotion.

Acknowledgments

I've been extremely privileged to work in a research lab which is truly dedicated to and passionate about robotics. Stefan Schaal and Ludo Righetti have never forced me to work on a particular project, told me to write a specific paper or discouraged me from trying a new idea. While the possibility of conducting any research I could dream up was intimidating at first, the flexibility and transparency of the research process itself made for an organic development of ideas, the likes of which I may never experience again in any other lab. While advice and guidance was always available (in particular from Ludo, who sacrificed many a relaxing evening at home to make video chatting work across a nine hour time difference), I'm very thankful for the autonomy I was afforded in developing the work which constitutes this thesis. From day one, I felt like a trusted and valued member of the lab rather than a student or employee; I greatly appreciate this egalitarian approach to advising and want to thank Stefan and Ludo for their efforts and patience in helping me become (I hope) an integral part of both the CLMC and AMD labs. I was able to travel all over the world not only to learn more about humanoids but to advertise my work, to expand my professional network and, in general, to grow as a person. My travels these past five years have led to great friendships, expanded my perspective on the world and greatly enriched my life in general. I cannot emphasize how much these opportunities mean to me, or thank Stefan and Ludo enough for allowing (and, what's more, encouraging) me to take them.

My many labmates throughout my time in grad school have also been incredibly supportive in helping me reach this point. In particular, I'm indebted to the core group of people that shared with me the opportunity (often, the struggle) of working with legged robots. Thank you to Ludo and Alex Herzog, who worked tirelessly to develop much of the codebase we now use on an everyday basis - Alex has been a great mentor in both robotics

and software development. John Rebula is an academic in the truest sense of the word, and never fails to inspire new ideas by approaching problems in what can only be described as the John-est of ways. I owe a huge thank-you to Sean Mason who is a fantastic engineer with a true passion for robotics, even in the face of the most discouraging hardware and software failures; he is an amazing colleague and now a lifelong friend. Felix Grimminger, without whom every one of our robots would be broken by now, is the best mechatronics engineer I've met. Thank you also to Brahayam Ponton, Maximilien Naveau, Julian Viereck, Bilal Hammoud, Michael Bloesch of ETH Zurich and the rest of our collaborators for helping me reach this point.

The degree of collaboration between different research groups and labs, both in Los Angeles and Tuebingen, is unique and wonderful; I felt incredibly lucky to be surrounded by hard-working people who love what they do and are always willing to help each other with experiments or simply chat about new ideas. My non-research conversations, of which there were probably far too many, were similarly enlightening - it was a privilege to work with such a diverse group of people who greatly expanded my world-view. I'm very grateful to have shared my journey with Franzi Meier, Vince Enachescu, Bharath Sankaran, Yevgen Chebotar, Giovanni Sutanto, Karol Hausman and Harry Su during my time at USC.

Finally, a big thank you to the unsung heroes whose names never appear in my papers or collaborations - my family and friends. First of all, thank you to my wonderful wife Dr. Lena Hoober-Burkhardt for her love, support, and patience in me and my work. You helped me achieve something much more important than a degree - a home, a renewed sense of self, and a life I love. Thank you to my big Italian family back in New York - Mom, Dad, Vinny, Karina, grandparents, aunts, uncles, cousins, cats, dogs and all the rest - for teaching me to work hard and never settle for less. Thank you to my wonderful friends in Los Angeles for all your support and distractions from grad school - Sean, Karol, Ola, Becky, Devang, Rahne, Harsh, Shruti, Carrie, Mark, Jimmy and many more - I hope we can continue adventuring together and inspiring one another for years to come.

And thank you, dear reader, for taking the time to read this thesis. I hope it serves you well.

Chapter I

Introduction

Over the course of the past decade, legged robots have become popular research platforms due to the availability of powerful, compact and low-cost sensors, actuators and processors (Semini, 2010), (Hutter et al., 2012b), (Englsberger et al., 2014). Humanoid robots in particular offer the ability to operate in complex and unstructured environments built for humans, making them ideal for a variety of tasks such as disaster recovery and household assistance. This versatility, however, comes at a cost - legged platforms in general must deal with underactuation, discontinuous dynamics and an inherent lack of stability. These are relatively new problems in robotics; wheeled mobile robots have traditionally been used for research on planning in two-dimensional worlds assuming deterministic dynamics, while fixed-base manipulators were normally used to investigate problems of joint space planning with kinematic redundancy and robust endpoint control in well-known environments.

In order to locomote autonomously in unknown environments, humanoids have largely been controlled using model-based approaches rather than the stiff joint controllers often used in manipulator robotics; this affords the system compliance which contributes to robustness in dealing with unplanned disturbances and contacts (Stephens and Atkeson, 2010). Purely model-based approaches, however, necessitate accurate models. Our robot dynamic (and even kinematic) models are often incorrect - and to uncertain degrees. Parameter identification methods can improve the quality of our models, but only within the form we assume they take (Mistry et al., 2009). This lack of accurate models has led

to the use of a combination of feedforward and feedback control to ensure stability while remaining robust to disturbances.

Such a framework, however, requires knowledge of the state of the robot. For legged robots in particular, the quantities we wish to control usually cannot be measured directly by any one sensor - these include kinematic quantities like endeffector poses and well as dynamic quantities such as the momenta of the rigid-body system (Herzog et al., 2014a). However, we can characterize the noise properties of individual sensors and build models which relate these sensor measurements to the robot's state (Woodman, 2007). Estimation of the state of the robot thus entails fusion of different sensing modalities with knowledge of how the state and measurements evolve according to our imperfect dynamics and sensor models.

The estimation of high-level quantities for humanoids additionally provides insight into what kinds of sensors we should incorporate into legged robots. As sensors have increasingly become cheaper and smaller due to advancements in design and manufacturing, we have the ability to augment robots with large amounts of redundant sensing. This can range from surface-mounted inertial sensors to a full tactile skin covering the links of the robot (Tsagarakis et al., 2007). Not only does such a design provide backup functionality in case a sensor fails during operation (for example using Kalman Filter validation gates (Alag et al., 2001)), but it also enables the fusion of many different modes of information having different noise characteristics and operating ranges. By analyzing sensor models and observability, we can make informed design decisions regarding the number of sensors and their optimal placement necessary to estimate the quantities we desire to control.

Beyond the estimation problem itself, it is difficult to understand how our estimators and controllers interact with one another in a real system. According to classical controls, for a linear system the design of an observer and the controller which uses its output for state feedback can be decoupled; this is known as the *separation principle* (Friedland, 1995). Humanoid robots, however, have highly nonlinear dynamics which invalidate this approach to estimation and control design, meaning that the performance of these components is coupled. On the other hand, we actually neglect to use much of the sensory information

available to us on humanoid platforms because it is unclear how to incorporate this data in the typical control framework employed on legged robots. Perhaps the most egregious example is the lack of use of contact wrench sensing on humanoids; despite the fact that most successful walking controllers plan stable motions by optimizing for desired contact wrenches, very few use these measurements for anything more than threshold-based contact detection (Kuindersma et al., 2016). Although it is difficult to use sensor signals directly for feedback control, we can implicitly incorporate this information in our controllers by using it to estimate states which are fed back or are used for online model predictive control. In addition, we can explore methods for utilizing sensor traces recorded from previous successful task executions to adapt planned motions to disturbances (Pastor et al., 2011).

I.1 The Current State of State Estimation

The results of the DARPA Robotics Challenge (DRC) 2014 trials and 2015 finals gave us unique insight into estimation strategies which were proven to work well in closed-loop control on full systems. Team IHMC employed a simple complementary filter for fusion of IMU sensory data and leg kinematics; however, this estimator included only the base position and velocity since the military grade Atlas gyroscope was assumed to be extremely low-drift. This method has the advantage of being computationally cheap but neglects IMU biases (which can be significant for cheaper IMUs) and potential foot slip, and fails to use knowledge of sensor noise to compute optimal state corrections as in a Kalman Filter (Johnson et al., 2015). Team CMU similarly fused inertial information with leg kinematics using precomputed steady-state Kalman Filters to estimate base position and velocity, also using the onboard IMU orientation directly (Feng et al., 2015). This approach thus fails to address incorporation of the rotational state of the base into the estimator (which is nontrivial and renders sensor biases observable); further, it does not permit foot slippage. The addition of a simplified model-based external force estimator (Xinjilefu et al., 2015) similar to (Stephens, 2011) helped compensate for disturbances and modeling errors; the COM and momentum estimator proposed here in Chapter IV can be

seen as a generalization of this LIPM-based method. MIT developed an EKF similar to that proposed in Chapter III, however it does not include foot orientation measurements (again because the gyroscope was assumed to be very accurate) and implements the foot position measurement as a velocity measurement instead (Kuindersma et al., 2016). Additionally, it neglects to use multiple contacts when more than one is available. However, this approach has been extended to use the LIDAR unit mounted on Atlas' head which renders position and yaw observable (Fallon et al., 2014). Overall, we conclude the following:

- Although estimation research over the past few years has focused increasingly on using the full robot dynamics often in an optimization-based framework (Xinjilefu et al., 2014b), (Xinjilefu et al., 2014a), (Lowrey et al., 2014), (Kolev and Todorov, 2015), the estimators used in the DRC were based on simplified dynamics and kinematic models. This highlights a disconnect between current research and approaches which are working reliably on real robots. We attempt to find a middle ground by using dynamic models primarily for state prediction, by simultaneously estimating modeling errors and by making better use of sensory information in estimation and control.

- While base state estimation is essential for whole-body control as evidenced by each team's efforts, estimation of the COM and momentum of the system are arguably more important as these dynamic quantities determine the stability of the robot and are often modeled incorrectly. Additionally, there are a number of works on momentum planning and control (Herzog et al., 2014b), (Herzog et al., 2015) which have seen limited use on real robots due to poor dynamic models. In the DRC, only CMU developed a COM estimator; it was seen to be highly beneficial in control.

- Sensory information is underutilized in control, missing both from direct sensor feedback as well as indirect feedback through estimation; in future work, we hope to introduce more coupling between estimation and control. Further, with the availability of low-cost MEMS sensors we can augment robots with large amounts of sensing; we believe that redundant measurements can help improve estimation and should be utilized.

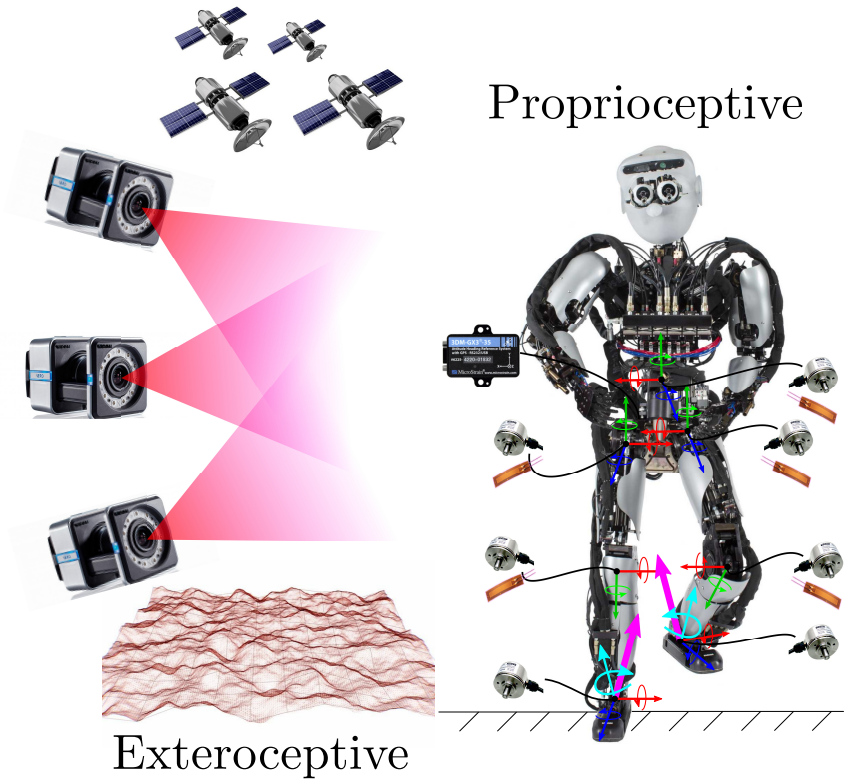


Figure I.1: Estimation involves the fusion of robot models and sensing; in our work, we rely entirely on *proprioceptive* or *internal* sensing (right) such as joint angle sensors, torque sensors and IMUs rather than *exteroceptive* or *external* sensing (left) such as Vicon, GPS and terrain maps in order to develop general-purpose methods which work in any environment.

I.2 Thesis Outline

In this thesis, we introduce methods for the estimation of various aspects of a humanoid robot's state and evaluate their performance and interaction with state-of-the-art methods for bipedal planning and whole-body control. The remainder of this document is organized as follows:

- Chapter II introduces the torque-controlled humanoid robots used in this work and details the low-level control setup and pipeline. Inverse dynamics control for both

fixed and floating-based systems is briefly introduced and used to develop an optimization-based controller which is used in subsequent experiments. Simplified models of a humanoid's dynamics are presented and used to summarize a class of model predictive control techniques which are popular for planning humanoid robot locomotion.

- Chapter III presents an estimator for the floating base pose of a humanoid robot which fuses information from inertial sensors and kinematics, taking into account contact switching. An observability analysis is used to investigate the theoretical properties of this estimator and its performance is evaluated both in a noisy simulation environment and on the real robot in the context of inverse dynamics balance control (Rotella et al., 2014).

- Chapter IV discusses the role of the center of mass and linear and angular momentum in planning and controlling stable motions on a humanoid robot and highlights the difficulties in doing so on real platforms. Multiple estimators based on the momentum dynamics are presented to address these issues; their performance is evaluated in simulation using perturbed dynamic models and their incorporation into popular control methods is proposed (Rotella et al., 2016).

- Chapter V addresses the issue of obtaining filtered and low-delay joint angle derivatives for a floating base robot using link-mounted inertial sensors and knowledge of the robot's kinematic structure. The resulting estimators motivate the use of redundant sensing in humanoid robots and offer insight on the number and placement of sensors necessary to obtain estimates which have been shown experimentally to increase control bandwidth on the real system (Rotella et al., 2016).

- Chapter VI investigates methods for estimating the contact state of a legged robot using proprioceptive sensing and a learning-based method. Clustering is applied to endeffector contact wrench and inertial sensor data collected during walking tasks, resulting in a method for computing the probability of endeffector contact directly from sensor data with minimal computation. This is seen to improve estimation

performance and has direct applications in inverse dynamics control as well (Rotella et al., 2017).

- Chapter VII concludes this thesis with a summary of the presented methods and an outlook on the future of sensing and control for legged robots.

- In addition, several appendices are included in order to present background information on key topics relevant to this work. We additionally maintain a large repository of relevant information at http://www-clmc.usc.edu/ nrotella/MyNotes.pdf which may prove useful.

Chapter II

Background

In this chapter, we begin by describing the Sarcos humanoids used for testing in the remainder of this work and detail the entire control pipeline from low-level hydraulic valve control to task implementation. We then introduce notation for describing robot dynamics in the context of methods for inverse dynamics-based robot control. Basic theory for controlling fixed-base manipulators is presented in order to describe the transition to floating base robots and highlight challenges in controlling legged robots. Finally, we conclude the chapter by introducing an optimization-based inverse dynamics controller based on a number of state-of-the-art approaches which will be used extensively for robot testing.

II.1 Hardware

All experiments in this thesis were performed on the lower portion of a Sarcos humanoid robot known as Hermes (Figure II.1, left) and in the corresponding SL simulation and control environment (Schaal, 2007). The lower body platform has 17 total degrees of freedom (DoFs) - seven in each leg and three in the torso. Each joint has potentiometers which measure joint angles and strain gauge-based load cells which measure torque; joints are actuated by hydraulic pistons acting through linkages which convert linear to rotational motion. Custom motor controller cards, located on the back of the torso, control each actuator by sending a command corresponding to a hydraulic valve position. At the lowest level, a proportional-integral-derivative (PID) control loop on applied current servos the

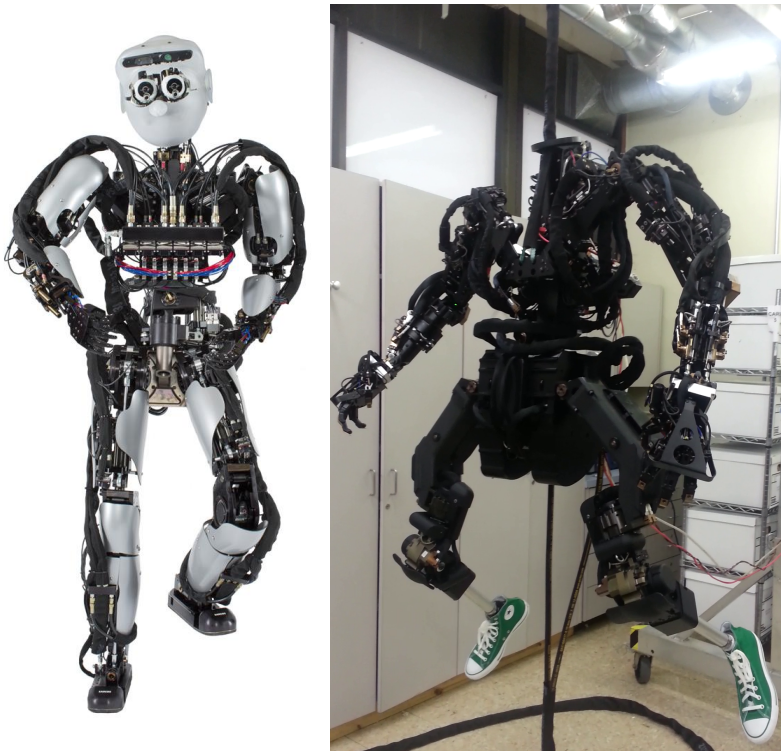


Figure II.1: The hydraulic, torque-controlled Sarcos humanoids Hermes and Athena.

valve to the desired position, achieving a certain flow rate. The Sarcos firmware allows for position, torque or direct valve control; in the past, we have implemented torque control by computing a valve command according to

$$u = \mathrm{PID}(F, F_{des}) + k_{vc}\dot{x}_{piston} + b$$

where F is the measured piston force, F_{des} is the desired piston force (computed from the desired torque and the linkage geometry), k_{vc} is a velocity compensation gain multiplying the piston linear velocity $\dot{x}_{piston}$ (computed from the joint velocity again using the linkage geometry) and b is a bias term used to compensate for a constant mechanical valve position offset. This control is inspired by the model-based torque control of (Boaventura et al.,

2012), however we cannot measure piston chamber pressures so we only perform an approximate compensation of the natural velocity feedback present in hydraulic actuators. Valve commands are computed as above offboard at 1kHz and sent to the motor controller cards to be translated into current and forwarded to the valve's internal controller. Alternatively, we have obtained the firmware source code and have successfully implemented the same valve control law directly on the motor controller cards; this allows us to feed back torque information at the motor controller frequency of 5kHz. These onboard torque controllers have been tuned to achieve better torque tracking performance in terms of tracking error and step response time (see Figure II.2). Additionally, we no longer perform velocity compensation because tracking is already sufficient and we prefer to keep some damping in the system as it contributes to stability.

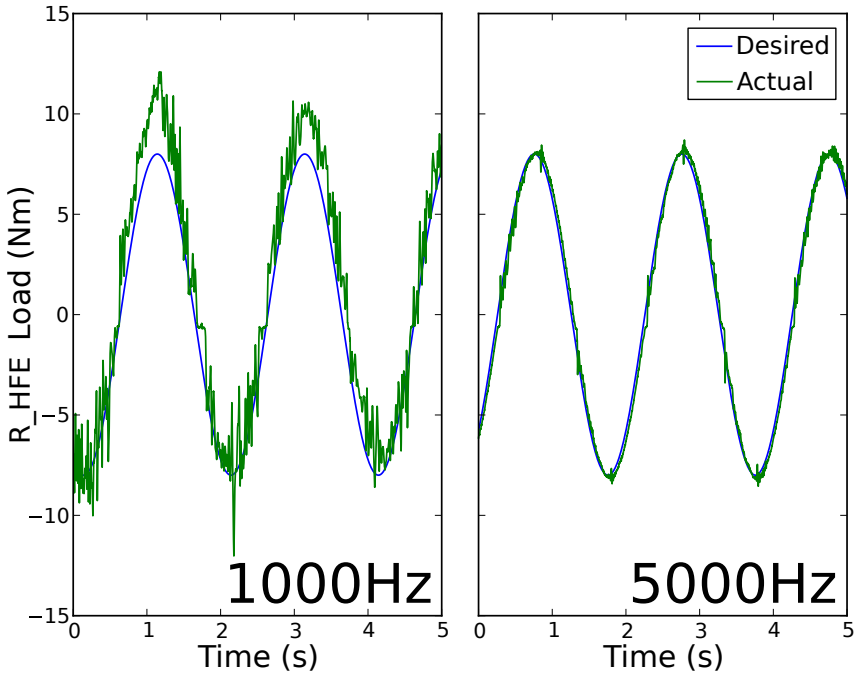


Figure II.2: Torque tracking comparison showing superior performance when running torque control onboard at 5 kHz (right) versus offboard at 1 kHz (left) for one joint.

Task-level controllers which compute desired torque commands are implemented off-board in SL on a Linux machine patched with the real-time Xenomai kernel (Gerum, 2004). Communication between this machine and the motor controller cards is achieved via UDP using the RTNet real-time ethernet Xenomai modules. We use Microstrain IMUs (models 3DM-GX3-25 and 3DM-GX3-45) with the open-source usb4rt real-time driver* to guarantee streaming of pre-processed angular velocity and linear acceleration at 1kHz.

II.1.1 The Athena Robot

We have a second Sarcos humanoid robot known as Athena (Figure II.1, right) located at our partner lab† which is controlled in an identical manner. The primary differences in this platform are

- The upper body is fully integrated with the lower body, adding many DoFs which can be used for creating additional contacts (e.g. holding a handrail while climbing stairs) and generating angular momentum for balancing. This will cause us to reconsider the simplified models we use for planning.
- The lower portion of each leg has been replaced with an Ottobock prosthesis. This makes the legs extremely light and easy to swing. However, it presents challenges for static locomotion approaches because the kinematic model is uncertain due to the unknown prosthesis deflection.

While the experiments and results of this thesis focus on the Hermes robot, the estimation approaches presented here can be applied to the Athena robot with minimal modification. From an estimation perspective, the main difference between the two robots is Athena's lack of ankle sensing; the ankle state is thus unobservable, making kinematics-based estimation impossible without assumptions. We propose to attach IMUs to the foot and shank in order to integrate the relative angular velocity into an estimate of the ankle angle when in contact; when not in contact, the prosthesis is undeformed and thus the

*http://git.kiszka.org/?p=usb4rt.git;a=summary

†https://am.is.tuebingen.mpg.de/

angle is known, allowing to reset its state in a Kalman filter. We leave this ankle angle estimator to future work.

II.2 Robot Dynamics and Control

While the previous section detailed the low-level control used on our humanoids, here we discuss the computation of joint torques to achieve desired motions using the robot's dynamic model; we begin with a brief summary of inverse dynamics control. While humanoids have only recently become popular research platforms, torque-controlled robots have existed for decades in the form of fixed-base manipulators. In contrast to the high-precision, position-controlled manipulators long used in industrial settings, torque-controlled manipulators have traditionally been used for research on controlling interaction between an endeffector with the environment (Nakanishi et al., 2008). The dynamics of such systems take the form

$$M(q)\ddot{q} + h(q, \dot{q}) = \tau + J^T \lambda \tag{II.1}$$

where $\ddot{q} \in R^n$ is the vector of joint accelerations, $\lambda \in R^{6m}$ is the vector of endeffector contact wrenches and $\tau \in R^n$ is the vector of joint torques; n denotes the number of DoFs and m denotes the number of endeffector contact points. The matrix $M(q) \in R^{n \times n}$ is the configuration-dependent mass inertia matrix of the robot, and $h(q, \dot{q}) \in R^n$ is the configuration-dependent vector of nonlinear terms (Coriolis, centripetal and gravity forces) acting on the robot. $J \in R^{6m \times n}$ denotes the endeffector Jacobian which (by the principle of virtual work) relates contact wrenches applied at the endeffector to joint torques through its transpose.

The final term in Eq (II.1) vanishes for a manipulator moving through free space. In this case, the feedforward (model-computed) joint torques τ_{ff} required to achieve a particular vector of joint accelerations $\ddot{q}_{des}$ can be computed directly as

$$\tau_{ff} = M(q)\ddot{q} + h(q, \dot{q})$$

this is called *inverse dynamics* because it maps from desired motion to controls by inverting the differential equations which describe the dynamics. Assuming accurate measurements of q and $\dot{q}$ at each timestep and an accurate dynamic model (no joint friction or other unmodeled dynamic effects), motion in joint space can be tracked extremely well. Of course, joint sensors are noisy and models are never perfect, so joint proportional-derivative (PD) control is added to the feedforward torques as

$$\tau = \tau_{ff} + \tau_{fb} = M(q)\ddot{q} + h(q, \dot{q}) + K_p(q_{des} - q) + K_d(\dot{q}_{des} - \dot{q})$$

where q_{des} and $\dot{q}_{des}$ are desired joint position and velocity. It can be shown that inverse dynamics with PD control is sufficient to stabilize a fixed-based manipulator from a Lyapunov stability perspective (Siciliano et al., 2009). The feedforward portion of the above inverse dynamics control is more generally known in the field of nonlinear control as *feedback linearization*. The purpose of this control method is to "linearize" the system by canceling its nonlinear dynamics using knowledge of the current state and dynamic model, allowing one to impose any desired dynamics (for example, PD control as above).

For controlling the position and orientation of the manipulator endeffector, a trajectory is usually designed in task (Cartesian) space and mapped to a desired joint space trajectory through inverse kinematics. In this case, there are six task DoFs; many manipulators are designed as *overactuated* (for example, with seven DoFs as in a human arm) which introduces redundancy into the system and allows for achieving the same endeffector motion while optimizing for secondary tasks, for example posture control or obstacle avoidance (Nakamura et al., 1987). In contrast to fixed-base robots, floating base robots such as quadrupeds and humanoids have an additional six DoFs corresponding to the pose of the base (root) link; these robots are thus *underactuated*. The dynamics become

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T\tau + J^T\lambda \tag{II.2}$$

where the matrix $S \in R^{n+6\times n}$ is a selector matrix which encodes the underactuation of the system. Joint torques corresponding to a desired joint plus floating base acceleration

vector $\ddot{q} \in R^{n+6}$ thus cannot be solved directly from Eq ($II.2$); we require an additional six constraints to determine τ. These come from contact constraints in the form

$$J_c\ddot{q} + \dot{J}_c\dot{q} = 0 \tag{II.3}$$

Here, $J_c \in R^{m_c \times n+6}$ denotes the portion of the Jacobian corresponding to the m_c endeffectors assumed to be in rigid contact with the ground. A single planar foot contact adds six constraints; when both feet are in planar contact with the ground, the robot becomes overactuated which allows for redundancy in inverse dynamics. The dynamics of a floating base robot thus vary over time as contacts are made and broken, resulting in discrete changes in the number of constrained degrees of freedom over time. We deal with task objectives and contact constraints as well as a number of other physical constraints on the system through the use of an optimization-based solver, developed according to the approaches used successfully in the DRC.

II.3 Quadratic Program-Based Inverse Dynamics

Optimization-based approaches have become popular in recent years due to the availability of efficient solvers which enable high frequency torque control (Stephens and Atkeson, 2010), (Hutter et al., 2012a), (Righetti et al., 2013). Some formulations allow a great deal of flexibility in task specification at the cost of added complexity (Herzog et al., 2014a), but here we use methods similar those which have been tested extensively on multiple real systems (Feng et al., 2015), (Johnson et al., 2015), (Fallon et al., 2015), (DeDonato et al., 2015) to focus on evaluating the estimators developed in this thesis with a widely-used whole-body control methodology. Again, the floating-base dynamics of the robot are formulated as

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T\tau + J_c^T\lambda$$

where $M \in R^{n+6 \times n+6}$ is the mass-inertia matrix, $h \in R^{n+6}$ is the vector of nonlinear terms (Coriolis, centrifugal, gravitational and joint friction forces) and $J_c \in R^{6m \times n}$ is the

endeffector contact Jacobian. As shown in (Herzog et al., 2014a), the above dynamics can be decomposed into the actuated and unactuated (floating base) dynamics, respectively:

$$M_u(q)\ddot{q} + h_u(q, \dot{q}) = \tau + J_{c,u}^T \lambda$$
$$M_l(q)\ddot{q} + h_l(q, \dot{q}) = J_{c,l}^T \lambda$$

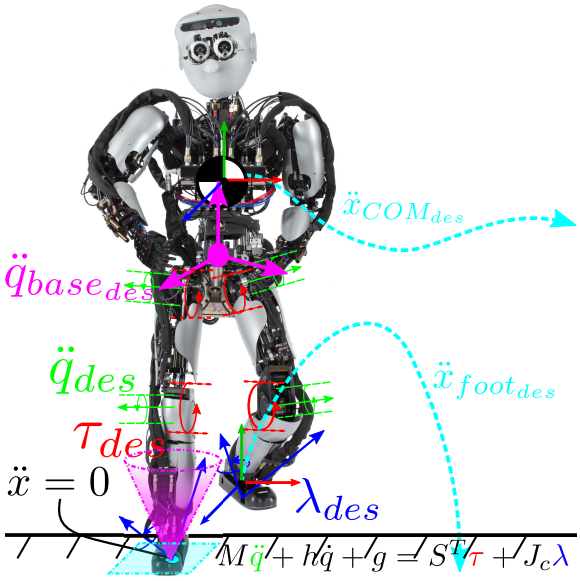


Figure II.3: Illustration of the various quadratic program-based inverse dynamics cost and constraint terms outlined in this section.

Due to the linear relationship between $\ddot{q}$, λ and τ, we need only optimize over two of these variables; we thus eliminate joint torques and constrain the optimization using the unactuated portion of the dynamics. The cost to be minimized is

$$\min_{\ddot{q},\lambda} ||\ddot{x} - \ddot{x}_{des}||_{W_x}^2 + ||P_{null}\ddot{q} - \ddot{q}_{des}||_{W_q}^2 + ||\lambda - \lambda_{des}||_{W_\lambda}^2 + ||\tau - \tau_{des}||_{W_\tau}^2$$

subject to the equality constraints

$$M_l(q)\ddot{q} + h_l(q, \dot{q}) = J_{c,l}^T \lambda$$
$$J_c\ddot{q} + \dot{J}_c\dot{q} = 0$$

as well as inequality constraints on foot COPs, friction forces, resultant normal torques, joint torques and joint accelerations. Joint torques are written in terms of $\ddot{q}$ and λ as

$$\tau = M_u(q)\ddot{q} + h_u(q, \dot{q}) - J_{c,u}^T \lambda$$

In the above cost, $W_x, W_q, W_\lambda, W_\tau$ denote weighting matrices, $P_{null} \in R^{n+6 \times n+6}$ projects into the nullspace of the Cartesian task constraints, and $x \in R^{6m}$ denotes the stacked Cartesian poses of the endeffectors. The relationship between joint and Cartesian accelerations is given by

$$\ddot{x} = J_t\ddot{q} + \dot{J}_t\dot{q}$$

where $J_t \in R^{6m_t \times n+6}$ is the task Jacobian for the m_t unconstrained endeffectors and $J_c \in R^{6m_c \times n+6}$ is the contact Jacobian for the m_c constrained endeffectors. Desired endeffector accelerations $\ddot{x}_{des}$ are computed from a planned Cartesian reference motion using a PD control law as

$$\ddot{x}_{des} = \ddot{x}_{ref} + P_x(x_{ref} - x) + D_x(\dot{x}_{ref} - \dot{x}) \tag{II.4}$$

Similarly, desired joint accelerations $\ddot{q}_{des}$ are specified as

$$\ddot{q}_{des} = P_q(q_{ref} - q) - D_q\dot{q}$$

where q_{ref} is a default joint configuration, achieving posture control in the nullspace of the task through the use of P_{null}. The inverse dynamics problem is illustrated in Figure II.3; for more details on this solver, see Appendix D.

II.4 Model Predictive Control

Optimization-based inverse dynamics allows for high-frequency control of conflicting tasks in a manner which guarantees satisfaction of *instantaneous* stability constraints. However, to achieve high-level behaviors such as walking and multi-contact interaction we must plan trajectories over a *horizon*. Further, planning should be done in a manner which considers the key aspects of the robot dynamics and respects physical limitations (for example, the robot cannot walk a meter in a single stride because of kinematic limits) else the inverse dynamics solver cannot achieve the behavior. This is known as *model predictive control* (MPC) and constitutes the highest layer of our control framework.

While approaches for MPC using the full dynamics of floating base robots exist and produce impressive results in simulation (Tassa et al., 2012), these are computationally expensive and suffer from a lack of accurate models on real platforms. Instead, humanoids are often controlled using plans generated based on the dynamics of a *simplified* model which captures the most important aspects of the task (Kajita et al., 2001). Such methods are generally simple enough to allow efficient online replanning (at frequencies anywhere from 10Hz to 1kHz) which makes them robust to disturbances and model discrepancies. Since the inverse dynamics control handles instantaneous stability, the MPC can be run at a slower rate according to the dynamics used. We discuss the most popular simplified model for humanoid control below.

II.4.1 The Linear Inverted Pendulum Model

For balancing and walking, the center of pressure (COP) - informally, the point on the contact surface about which planar moments sum to zero - provides a stability metric used in planning and control (Goswami, 1999). If the overall COP (assuming coplanar contacts) leaves the support polygon (the convex hull of all contact points) then an unbalanced

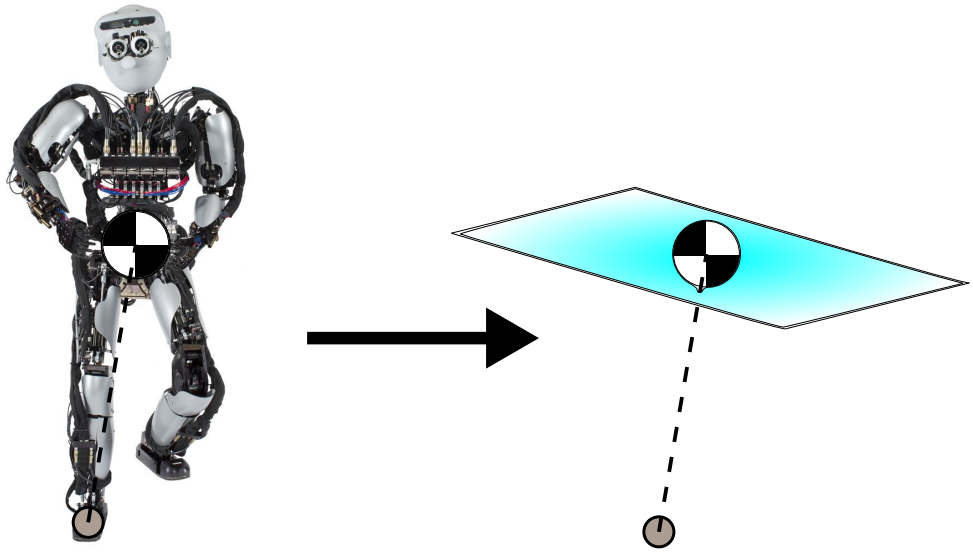


Figure II.4: Illustration of simplification of dynamics from the full humanoid to the LIPM.

moment about the COM exists which will cause the robot to topple. The linear and angular momentum summed about the COP are

$$\dot{l} = F_{res} + mg \tag{II.5}$$

$$\dot{k} = (p - r_{COM}) \times F_{res} + \begin{bmatrix} 0 \\ 0 \\ \tau_n \end{bmatrix} \tag{II.6}$$

where l is the linear momentum, k is the angular momentum about the COM denoted r_{COM}, and F_{res} and τ_n are the resultant force and normal moment (torque) at the COP denoted p. We can solve for the COP as

$$p_x = r_{COM,x} - \frac{1}{\dot{l}_z - mg}\left(r_{COM,z}\dot{l}_x + \dot{k}_y\right) \tag{II.7}$$

$$p_y = r_{COM,y} - \frac{1}{\dot{l}_z - mg}\left(r_{COM,z}\dot{l}_y - \dot{k}_y\right) \tag{II.8}$$

The COP is thus the sum of the COM and a dynamic term computed from the rate of change of the system momenta; at rest, the COM and the COP are aligned. This means that static walking can be planned simply by planning slow motions of the COM. However, for faster walking trajectories the momentum of the system must be accounted for to ensure stability. Neglecting angular momentum, limiting the COM to a fixed height z_c (or more generally, motion on a specified plane) and using the fact that $l = m\dot{r}_{COM}$ we obtain

$$p_x = r_{COM,x} + \frac{z_c}{g}\ddot{r}_{COM,x} \tag{II.9}$$

$$p_y = r_{COM,y} + \frac{z_c}{g}\ddot{r}_{COM,y} \tag{II.10}$$

These are the dynamics of the *linear inverted pendulum model* (LIPM), a popular simplified model which has been used extensively for humanoid control (as shown in Figure II.4). This model relates the COP linearly to the COM and its derivatives, allowing us to use linear model predictive control techniques to compute a COM trajectory which achieves a desired COP motion - usually set to move linearly between the feet during walking.

Alternatively, recent work has shown the utility of planning and controlling the full momentum of Eq (II.5) in order to stabilize humanoid robots and generate dynamic motions (Stephens and Atkeson, 2010),(Lee and Goswami, 2012),(Wensing and Orin, 2013),(Herzog et al., 2014b),(Herzog et al., 2015). The advantage of such methods is their use of the dynamically-consistent momentum dynamics which constitute a *reduced* rather than simplified model. We discuss the use of this model for estimation and propose the combination of momentum estimation and control in Chapter IV.

II.5 MPC-Based Walking Controller

The theory and resulting methods detailed in this chapter form the building blocks for a complete, whole-body walking controller similar to those used extensively in the DRC. We implement such a controller using a state machine approach, allowing new gait phases

to be easily introduced and transitioned to based on either a fixed sequence or, ideally, a higher-level gait controller.

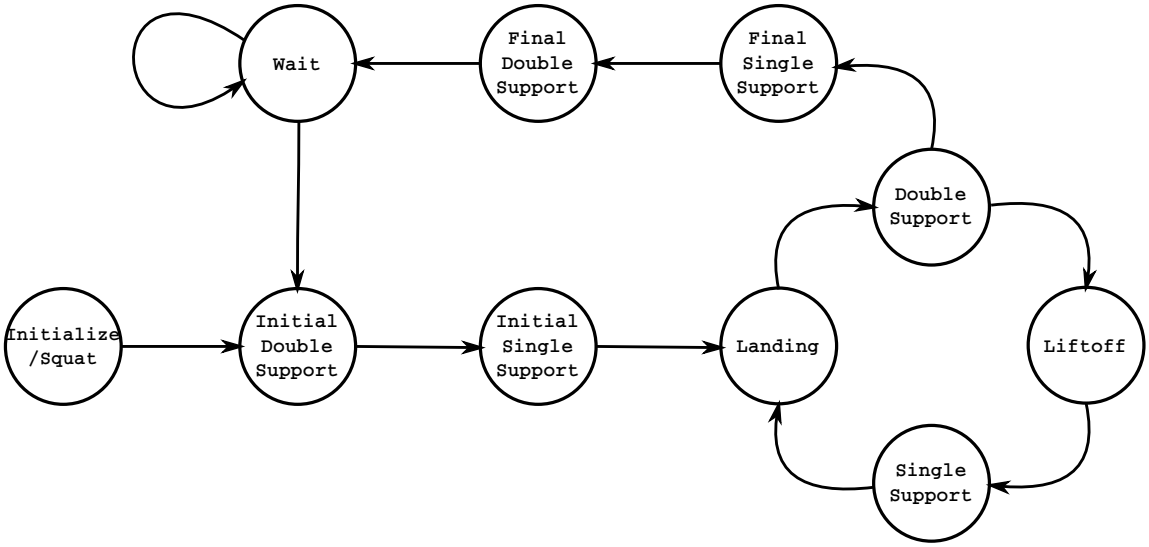


Figure II.5: Simple walking state machine.

The advantage of developing a state-based walking controller is that it naturally offers the flexibility required to translate user input from a device into gait parameter changes and state transitions. We store gait information in a YAML-based class which can be both read and modified from within both the walking states and the higher-level state scheduler, allowing user input data to be passed down to the classes which actually implement gait phases as well as back up to the state machine controller.

Using this structure, we have implemented a generic walking controller which takes input from a joystick (or general gaming input device) using a Robot Operating System (ROS) interface Quigley et al. (2009) to allow starting and stopping the gait, changing step height, modulating walking step width and length as well as changing the walking forward direction. In addition, we use the remaining inputs to apply simulated forces of adjustable magnitude and direction at the robot base; this permits easily testing the robustness of our walking controller. Using the ROS tool *rosbag* allows us to record joystick input during a task and play it back to create repeatable desired walking gaits or disturbance forces from

user commands. This significantly simplifies the testing of a new controller or controller module.

In an effort to make the state machine and user input structure of our controller as general as possible, we have additionally implemented the ability to switch user input modes. This allows having, for example, different input layouts for walking and upper body manipulation. The controller designer can then check the input mode state from within the state machine and translate user input commands accordingly. This has been used to implement single and dual arm manipulation modes for the simulated, full-body robot.

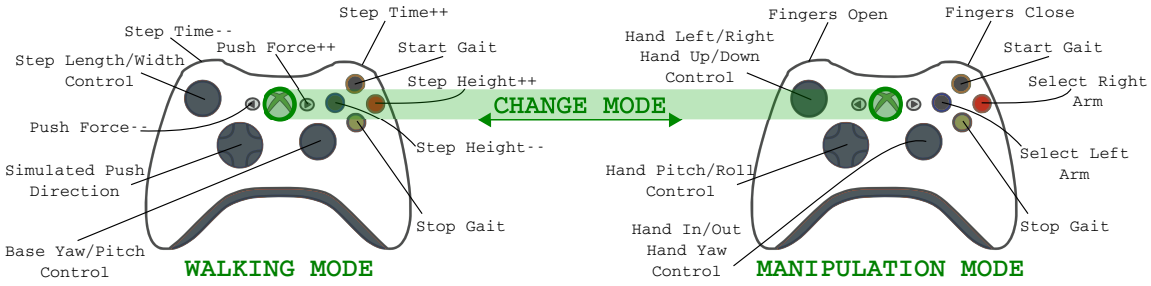


Figure II.6: Example user input mapping for an Xbox controller with multiple modes.

II.6 Summary

Now that the control pipeline has been introduced, the remainder of this thesis focuses on the challenges of estimation for use in humanoid control. Not only is controlling legged robots difficult because of discontinuous dynamics, but also because the additional six degrees of freedom in the generalized joint angle vector $q \in R^{n+6}$ cannot be measured directly except through the use of one or more *exteroceptive* sensors (those which are external to the robot, for example GPS or visual marker-based tracking systems). Since the proposed utility of such platforms is their potential for operation in unstructured environments, exteroceptive sensors are unfit for use; this forces the use of *proprioceptive* or internal sensing - for example inertial, joint angle and contact wrench sensors - while

dealing with intermittent contact switching. Base pose estimation in the context of these considerations is the focus of Chapter III.

The simplified and reduced model-based approaches of Section II.4 are limited on real systems due to the lack of accurate estimates of the COM and momentum. Joint velocities, computed on our robots from noisy potentiometer signals, result in extremely noisy momentum which limits how much we can damp our controllers. Inaccurate link masses and link inertias contribute to configuration-dependent offsets in the COM and momenta, while unmodeled wrenches applied about the COM can likewise destabilize control. Even traditional approaches using simplified models (such as the LIPM) for planning rely on accurate center of mass estimates to achieve good tracking. We therefore focus on momentum estimators which deal with these challenges in Chapter IV.

In general, achieving good damping on real systems is difficult due to noise on joint derivatives which gets propagated to task-level quantities in control. Since we compute joint velocities and accelerations by numerically differentiating noisy position signals, we are limited as to how high we can increase our joint and Cartesian damping gains before the controller goes unstable. By using dynamics-based process models and measurements derived from link-mounted inertial sensors we can filter the joint state and thus use higher gains; this is the focus of Chapter V.

Knowledge of contact is crucial for determining endeffector constraints in both base state estimation and whole-body control, however the contact state is almost always determined by thresholding the measured normal force at the endeffector. This is an extremely simplified view of contact, which is truly a six DoF quantity; it is assumed that a high-enough force threshold ensures the endeffector will not slip or rotate. Instead, in Chapter VI we propose a clustering-based method which uses endeffector force/torque sensing and additional IMUs to detect slip and rotation automatically and modulate the probability of contact in six DoFs, and evaluate this contact estimator for base state estimation.

Before proceeding to discuss each of these estimation-based approaches to the control problems we encounter on real systems when using the solver of Section II.3, we summarize the humanoid control framework presented in this chapter.

- We work with Sarcos hydraulic humanoids which are torque-controlled at 5kHz on the motor controller cards, receiving desired torque commands at 1kHz from the SL simulation and control framework over real-time ethernet from a Xenomai-patched Linux machine.

- Humanoids are typically controlled via inverse dynamics, a model-based control method which is used to cancel nonlinearities and impose desired behavior in joint and/or Cartesian space. By using a combination of feedforward and feedback control, the robot remains compliant and thus robust to disturbances.

- We have developed a quadratic program-based inverse dynamics solver based on control methods used by the DRC teams on the torque-controlled Atlas humanoid platform. This method of control has more-or-less become common practice and allows us to test estimation-based planning and control using a benchmarked solver.

- The Linear Inverted Pendulum Model (LIPM) was introduced as a simplified form of the full robot dynamics for use in Model Predictive Control (MPC) over a time horizon. This control methodology is used to plan walking trajectories, which are modulated from user input in the context of a general-purpose walking state machine. The resulting walking controller is used extensively in this thesis for testing control and estimation methods in locomotion experiments.

Chapter III

Base State Estimation

One of the earliest attempts (Roston and Krotkov, 1991) at base pose estimation for a legged robot was performed on the CMU Ambler, a hexapod robot having only joint encoders. The positions of the feet were computed both from motor commands and encoder measurements. Each foot contributed a measurement and the transformation which minimized the error between the two estimates was computed. Nearly fifteen years later, (Gassmann et al., 2005) relied on the same dead-reckoning method, extending earlier work by fusing odometry with orientation and position measurements from new inertial sensors such as IMU and GPS units. However, this approach was inherently unable to handle non statically-stable gaits. Around the same time, (Lin et al., 2006) extended previous work on pose estimation using a strain-based model with MEMS inertial sensors to measure motion of the body. The gait was split into multiple phases, each of which corresponded to a simple Kalman filter. Models for each phase were switched using sensory cues, allowing non statically-stable gaits. However, it was shown that the filter provides accurate pose estimates only when in tripod support.

For quadrupeds, (Chitta et al., 2007) employed a particle filter which used an odometry-based prediction model for the COM during quadruple support and an update model based on IMU data, joint encoder readings and knowledge of the terrain relief for state estimation with the quadruped LittleDog. While this method permitted global localization, it assumed knowledge of the terrain and a statically-stable gait. A state estimator was introduced (Cobano et al., 2008) for the quadruped SILO4 which fused changes in position

(computed using dead-reckoning) with magnetometer and DGPS measurements in an Extended Kalman Filter (EKF). While they tracked global position and heading without gait assumptions, the full 6 DoF pose was not estimated. (Reinstein and Hoffmann, 2011) introduced an EKF for a quadruped which combined kinematic predictions from IMU data with a "data-driven" velocity measurement. By assuming an uninterrupted gait, the velocity was computed using a learned model. However, the gait assumption and model limited the approach's utility on other platforms. Recent work (Chilian et al., 2011) on the DLR Crawler hexapod platform introduced an information filter suitable for combining multiple types of measurements. The process model integrated IMU data to track the pose of the hexapod while visual and leg odometry were used for updates. Absolute measurements of roll and pitch angles were obtained from the accelerometer and used as well. While they made no gait assumptions, the leg odometry measurements were valid only during periods of three or more contacts.

Base pose state estimation for bipedal legged robots has only recently gained attention as more humanoid platforms have become available. (Park et al., 2009) introduced a Kalman Filter in the context of a Zero Moment Point (ZMP) balance controller using the Linear Inverted Pendulum Model (LIPM) to approximate the dynamics of the humanoid robot. The position, velocity, and acceleration of the COM of the LIPM were estimated using the pendulum dynamics and the measured ZMP location was used for the update step. However, the approach was only applicable in the context of ZMP balancing and walking. In a similar context, an Unscented Kalman Filter which provided estimates of the joint angles and velocities for predictive ZMP control was proposed (Wang et al., 2013). The filter treated the biped in single support as a fixed-base manipulator with corresponding dynamics. Of course, this assumption was violated if the robot lost contact or slipped. Additionally, the absolute orientation could not be observed. This filter was also computationally-demanding as it used the full manipulator dynamics for prediction. The DRC led to the development of a number of base state estimators for humanoids, however all fused inertial sensing with kinematics as done in the work presented here. Only one approach (Fallon et al., 2014) was truly novel in that it incorporated LIDAR-based

measurements for absolute position and yaw sensing (see Chapter I for a more complete analysis of estimation in the DRC).

More recently, estimation methods using optimization have been proposed (Xinjilefu et al., 2014a) using the full robot dynamics and measurements from many sensors. Both states and controls are estimated in a quadratic program and sensors can be easily integrated since the dynamics are linear in terms of the state. Additionally, constraints can be enforced whereas this is not straightforward to do in a KF. However, this approach relies heavily on the full dynamics (namely link inertias) which we usually do not have good estimates of. Further, offsets in the COM and momentum are not considered. Another recent work (Faraji et al., 2015) uses optimization methods to estimate base position, velocity and yaw angle (assuming roll and pitch can be used directly from the internal IMU estimator) in a multi-stage filter. This method can easily handle contact switching, however it assumes no foot movement or slippage.

In this chapter, we propose a base pose estimator, analyze its theoretical characteristics and evaluate its performance in a simulation environment with realistic added noise. Finally, we evaluate the performance of this estimator in combination with the inverse dynamics solver proposed in Section II.3 for different whole-body control tasks on the real robot.

III.1 Prediction and Measurement Models

In order to introduce required terminology and notation, we begin by introducing the models used in the presented estimator. Note that these models are nonlinear, leading us to employ an Extended Kalman Filter (EKF); we chose this formulation over alternatives such as the Particle Filter or Unscented Kalman Filter for its simplicity and low computational cost. However, the presented framework and observability analysis hold regardless of the method employed. For a review of Kalman Filtering, see Appendix A.

In (Bloesch et al., 2012) a continuous-time, nonlinear prediction model describing the time evolution of the state of the quadruped was developed based on rigid body kinematics

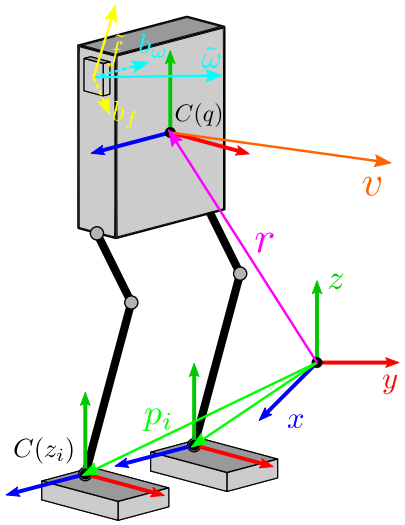


Figure III.1: Base state estimation notation for a bipedal robot.

and a simple model of an IMU consisting of a three-axis accelerometer and a three-axis gyroscope. This prediction model is shown below. Note that this formulation can accommodate an arbitrary number of feet, denoted N.

$$\dot{r} = v \tag{III.1}$$

$$\dot{v} = a = C^T(\tilde{f} - b_f - w_f) + g \tag{III.2}$$

$$\dot{q} = \frac{1}{2}\begin{bmatrix}\tilde{\omega} - b_\omega - w_\omega \\ 0\end{bmatrix} \otimes q \tag{III.3}$$

$$\dot{p}_i = C^T w_{p,i} \quad \forall i \in \{1, \ldots, N\} \tag{III.4}$$

$$\dot{b}_f = w_{bf} \tag{III.5}$$

$$\dot{b}_\omega = w_{b\omega} \tag{III.6}$$

The state of the filter is $x = [r, v, q, p_i, b_f, b_w]$ where r is the position of the IMU (assumed to be located at the base), v is the base velocity, q is the quaternion representing a rotation from the world to body frame, p_i is the world position of the i^{th} foot and b_f and b_w are the accelerometer and gyroscope biases, respectively. Unless noted, $C(q)$ or simply C

represents the rotation matrix corresponding to q. The raw accelerometer and gyroscope data are $\tilde{f}$ and $\tilde{\omega}$ and are modeled as being subject to additive thermal noise processes w_f and w_ω as well as random-walk biases parameterized by the noise processes w_{bf} and $w_{b\omega}$ (Woodman, 2007). Finally, $w_{p,i}$ denotes the noise process representing the uncertainty of the i^{th} foothold position.

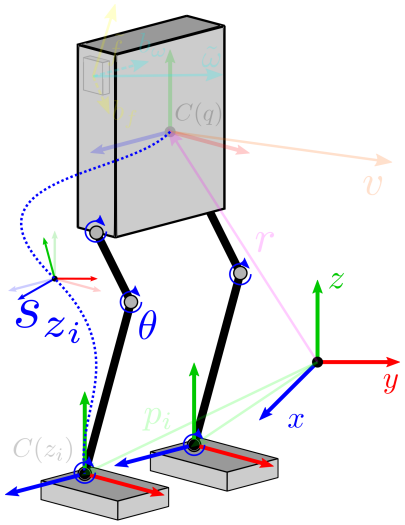


Figure III.2: Base state estimation augmented with the relative foot orientation measurement s_{z_i} corresponding to foot i for a bipedal robot.

To extend the established filter to a humanoid platform, we now augment the state vector with the orientations of the feet. Including only one foot for brevity, the new state is defined as $x = [r, v, q, p, b_f, b_w, z]$ where z is the quaternion representing the rotation from the world to foot frame. Analogous to the assumption made in (Bloesch et al., 2012), we assume that the orientation of the foot remains constant while in contact; to permit a small amount of rotational slippage, the prediction equation is defined to be

$$\dot{z} = \frac{1}{2} \begin{bmatrix} w_z \\ 0 \end{bmatrix} \otimes z \tag{III.7}$$

where w_z is the process noise having covariance matrix Q_z. The foot orientation noise is applied as an angular velocity in order to remain consistent with the original model.

The original filter update step was performed using one measurement equation for each foot which represented the position of that foot relative to the base as measured in the base frame. This measurement is a function only of the measured joint angles and the kinematic model of the leg; it is written as

$$s_p = C(p - r) + n_p \tag{III.8}$$

The noise vector n_p represents the combination of noise in the encoders and uncertainty in the kinematic model. Its covariance is the main tuning parameter of the filter.

Following the original filter formulation, we introduce an additional measurement which is again a function of only the measured joint angles and the kinematics model. The orientation of the foot in the base frame represented by the quaternion

$$s_z = \exp(n_q) \otimes q \otimes z^{-1} \tag{III.9}$$

describes the rotational constraint imposed by a flat foot contact. The noise term n_q is applied using the exponential map; see Appendix (A.12) for details. Again, the noise term depends on the forward kinematics uncertainty and constitutes a tuning parameter.

When a foot loses contact, its measurement equations are temporarily dropped from the filter. Additionally, its pose is removed from the state; this can be achieved more simply by setting the variances of w_p and w_z corresponding to the foot to large values. This causes the pose uncertainty to grow rapidly. When contact is restored, the pose and measurements are included again; this triggers a reset of the foot pose to its new value. This allows for contact switching without the need for separate models. During an aerial phase, the filter reduces to integration of the prediction model.

Since the above system is continuous and nonlinear, it must be discretized and linearized for implementation purposes. Following the EKF framework outlined above, this requires two systems of equations: a *discrete, nonlinear* system for prediction of the state and

measurement and a *discrete, linear* system for propagation of the state covariance through the prediction model and for computation of the gain in the update step. Additionally, we will derive in an intermediate step a continuous, linear system. We begin with a discussion of the discrete, nonlinear system used for prediction.

III.1.1 Discrete, Nonlinear Model

The first step in the EKF is the propagation of the expected value of the state using the discretized nonlinear model. Assuming a zero-order hold on the IMU data over a small timestep Δt, we discretize the original system using a first-order integration scheme as

$$\hat{r}_{k+1}^{-} = \hat{r}_k^{+} + \Delta t \hat{v}_k^{+} + \frac{\Delta t^2}{2}(\hat{C}_k^{+T}\hat{f}_k + g) \tag{III.10}$$

$$\hat{v}_{k+1}^{-} = \hat{v}_k^{+} + \Delta t(\hat{C}_k^{+T}\hat{f}_k + g) \tag{III.11}$$

$$\hat{q}_{k+1}^{-} = \exp(\Delta t \hat{\omega}_k) \otimes \hat{q}_k^{+} \tag{III.12}$$

$$\hat{p}_{k+1}^{-} = \hat{p}_{i,k}^{+} \tag{III.13}$$

$$\hat{b}_{f,k+1}^{-} = \hat{b}_{f,k}^{+} \tag{III.14}$$

$$\hat{b}_{\omega,k+1}^{-} = \hat{b}_{\omega,k}^{+} \tag{III.15}$$

$$\hat{z}_{k+1}^{-} = \hat{z}_{i,k}^{+} \tag{III.16}$$

where $\hat{f}_k = \tilde{f} - \hat{b}_{f,k}^{+}$ and $\hat{\omega}_k = \tilde{\omega} - \hat{b}_{\omega,k}^{+}$ denote the expected values of the measured acceleration and angular velocity, respectively. Note that the quaternion representing the base pose is updated using the exponential map formed from the infinitesimal rotation $\Delta t\hat{\omega}$ which is measured in the base frame directly. Also note that a second-order discretization is used for the position in order to incorporate the IMU acceleration at the current timestep.

The measurement model is discretized simply as

$$\hat{s}_{p,k} = \hat{C}_k^{-}(\hat{p}_k^{-} - \hat{r}_k^{-}) \tag{III.17}$$

$$\hat{s}_{z,k} = \hat{q}_k^{-} \otimes \left(\hat{z}_k^{-}\right)^{-1} \tag{III.18}$$

to produce the expected measurement of the nonlinear system.

III.1.2 Continuous, Linear Model

Recall that discretized, linearized dynamics are required in order to propagate the state estimate covariance and perform the update step. It is our preference to linearize and then discretize; this is the approach found in many texts. Linearization is performed by expanding each state around its current estimate using a first-order approximation. For the full derivation of the linearized system presented in this section, see Appendix B. This approach results in the linearized model

$$\dot{\delta r} = \delta v \tag{III.19}$$

$$\dot{\delta v} = -C^T f^\times \delta\phi - C^T \delta b_f - C^T w_f \tag{III.20}$$

$$\dot{\delta\phi} = -\omega^\times \delta\phi - \delta b_\omega - w_\omega \tag{III.21}$$

$$\dot{\delta p} = C^T w_p \tag{III.22}$$

$$\dot{\delta b}_f = w_{bf} \tag{III.23}$$

$$\dot{\delta b}_\omega = w_{b\omega} \tag{III.24}$$

$$\dot{\delta\theta} = w_z \tag{III.25}$$

where the measured IMU quantities are bias-compensated and where $v^\times$ denotes the skew-symmetric matrix corresponding to the vector v. The error state and process noise vectors are defined to be $\delta x = [\delta r, \delta v, \delta\phi, \delta p, \delta b_f, \delta b_\omega, \delta\theta]^T$ and $w = [w_f, w_\omega, w_p, w_{bf}, w_{b\omega}, w_z]^T$, respectively. The linearized system can then be written in state-space form as $\dot{\delta x} = F_c \delta x + L_c w$ where

$$F_c = \begin{pmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -C^T f^\times & 0 & -C^T & 0 & 0 \\ 0 & 0 & -\omega^\times & 0 & 0 & -I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

and $L_c = diag\{-C^T, -I, C^T, I, I, I\}$ are the prediction and noise Jacobians, respectively. It is assumed for simplicity that the covariance matrix of each process noise vector is diagonal with equal entries. The continuous process noise covariance matrix is then $Q_c = diag\{Q_f, Q_\omega, Q_p, Q_{b_f}, Q_{b_\omega}, Q_z\}$. The measurement model defined by (III.8) and (III.9) is linearized as

$$s_p = -C\delta r + (C(p-r))^x \delta\phi + C\delta p + n_p$$
$$s_z = \delta\phi - C[q \otimes z^{-1}]\delta\theta + n_z$$

where $C[m]$ is used to denote the rotation matrix corresponding to the quaternion m. This model can be written in the form $\delta y = H_c \delta x + v$ where $v = [n_p, n_z]^T$ is the measurement noise vector and

$$H_c = \begin{pmatrix} -C & 0 & (C(p-r))^x & C & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 & -C[q \otimes z^{-1}] \end{pmatrix}$$

is the measurement Jacobian. It is assumed that measurements are uncorrelated and hence the measurement noise covariance matrix is defined as $R_c = diag\{R_p, R_z\}$ where R_p and R_z are again each diagonal with equal entries.

III.1.3 Discrete, Linear Model

Assuming a zero-order hold on inputs over the interval $\Delta t = t_{k+1} - t_k$, the discretized prediction Jacobian is given by

$$F_k = e^{F_c \Delta t}$$

and the discretized state covariance matrix is given by

$$Q_{k-1} = \int_{t_{k-1}}^{t_k} e^{F_c(t_k - \tau)} L_c Q_c L_c^T e^{F_c^T (t_k - \tau)} d\tau$$

In practice, these expressions are often truncated at first order for simplicity to yield $F_k \approx I + F_c \Delta t$ and $Q_k \approx F_k L_c Q_c L_c^T F_k^T \Delta t$. In contrast to (Bloesch et al., 2012), the system

is discretized using a zero-order hold on the noise terms as above in order to simplify the implementation.

Following the above procedure, we find the discretized prediction and measurement Jacobians to be

$$F_k = \begin{pmatrix} I & I\Delta t & 0 & 0 & 0 & 0 & 0 \\ 0 & I & -C_k^T f_k^\times \Delta t & 0 & -C_k^T \Delta t & 0 & 0 \\ 0 & 0 & I - \omega_k^\times \Delta t & 0 & 0 & -I\Delta t & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{pmatrix},$$

$$H_k = \begin{pmatrix} -C_k & 0 & (C_k(p_k - r_k))^\times & C_k & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 & -C[q_k \otimes z_k^{-1}] \end{pmatrix}$$

where all quantities are computed using the *a priori* state vector $\hat{x}_k^-$. Finally, the continuous measurement covariance matrix is discretized as $R_{k-1} \approx \frac{R_c}{\Delta t}$.

III.2 Observability Analysis

As in (Bloesch et al., 2012), a nonlinear observability analysis of the filter was performed. This section discusses the resulting observability characteristics and analyzes the theoretical advantages of the addition of the foot rotation constraints.

The unobservable subspace informally describes all directions along which state disturbances cannot be observed in the outputs. This corresponds to the nullspace of the observability matrix. For a nonlinear system, determining this matrix involves computing the gradient of successive Lie derivatives of the measurement model with respect to the process model as in (Hermann and Krener, 1977); see Appendix C for more details. Since this space depends on the robot's motion, the presented analysis reveals all possible singularities and corresponding rank losses (RL). Note that since absolute position and yaw

are inherently unobservable in this system, rank loss here represents an increase in the dimension of the unobservable subspace beyond this nominal case. While the full derivation omitted for brevity, the pertinent results are summarized in tables III.1, III.2, and III.3.

Table III.1 below describes the rank deficiency for the case in which a single point foot is in contact. The top row ($w = 0$) describes the case in which there is no rotational motion. Depending on the acceleration, the rank loss is either 3 or 5. The second row ($w \perp Cg$) states that rotational motion around an axis which is perpendicular to the gravity axis leads to a rank loss of 1. The third row ($w \parallel Cg$) describes the case in which there is rotation only around the gravity axis; in general, this corresponds to a rank loss of 1. If the axis of rotation additionally intersects the point of contact, rank loss increases to 2. Further, if the IMU is directly above the point of contact then rank loss increases to 3. Finally, the last row summarizes the nominal case.

Table III.1: Rank deficiency for a single point foot contact.

Rotation	Acceleration/Velocity	Foothold	RL
$w = 0$	$a = -1/2g$	$*$	5
	$a \neq -1/2g$	$*$	3
$w \perp Cg$	$*$	$*$	1
$w \parallel Cg$	$\wedge$ $a = \left(C^T w\right) \times v$	$(r-p) \parallel g$	3
	$\wedge$ $v = \left(C^T w\right) \times (r-p)$	$(r-p) \nparallel g$	2
	$\vee$ $a \neq \left(C^T w\right) \times v$ $v \neq \left(C^T w\right) \times (r-p)$	$*$	1
$\wedge$ $w \not\perp Cg$ $w \nparallel Cg$	$*$	$*$	0

Table III.2 below details the singular cases for two point foot contacts. These are similar to the single point foot contact cases but with reduced rank losses.

Table III.2: Rank deficiency for two point foot contacts.

Rotation	Footholds	RL
$w = 0$	$2a + g \parallel \Delta p$	3
	$2a + g \nparallel \Delta p$	2
$w \perp Cg$	$*$	1
$w \parallel Cg$	$g \parallel \Delta p$	1
	$g \nparallel \Delta p$	0
$\wedge \begin{matrix} w \not\perp Cg \\ w \nparallel Cg \end{matrix}$	$*$	0

In comparison to the point foot cases, the flat foot case is significantly simpler as shown in Table III.3. These results are valid for any number of flat foot contacts since a single flat foot contact fully constrains the pose of the base.

Table III.3: Rank deficiency for an arbitrary number of flat foot contacts.

Rotation	RL
$w = 0$	2
$w \perp Cg$	1
$w \not\perp Cg$	0

The rank loss of the new filter depends only on the base angular velocity. If the angular velocity is zero then the rank loss is 2; if the axis of rotation is perpendicular to the gravity axis then the rank loss is 1. For all other cases, only absolute position and yaw are unobservable; inclusion of foot orientations renders the IMU sensor biases observable.

It's clear that the additional information resulting from the the rotational constraint of the foot significantly reduces rank loss, as expected. In summary, the maximum rank loss is reduced from 5 to 2 when there is no rotational motion. Additionally, rotation purely around the gravity axis no longer induces rank loss. Finally, since the results of Table III.3 hold for any number of flat foot contacts, both single and double support walking

phases have desirable observability characteristics. The experimental results of section V.4 demonstrate the practical effects of the singular cases.

III.3 Experiments and Results

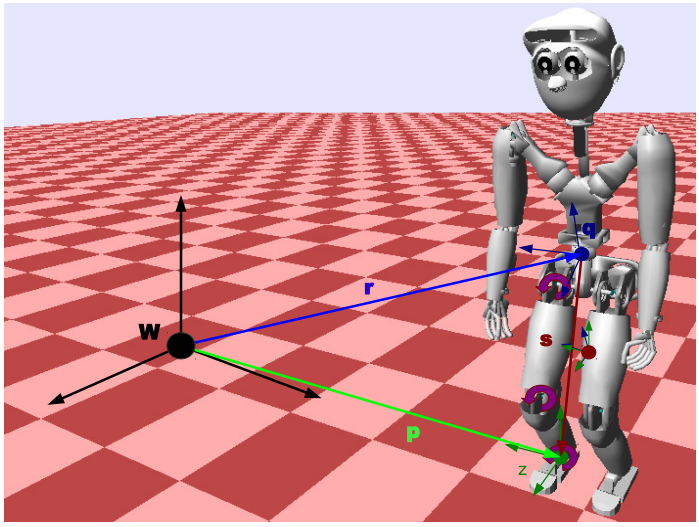


Figure III.3: Base state estimation notation on Hermes robot in SL simulation environment.

The goal of this work is to implement the proposed EKF on a Sarcos humanoid equipped with a Microstrain 3DM-GX3-25 IMU. However, it was desired that its performance first be verified in simulation. For this purpose, both filters were implemented in the SL simulation environment (Schaal, 2007).

In order to provide an accurate assessment of these filters, realistic levels of noise were added in the simulator by drawing samples from an i.i.d. Gaussian white noise process. Thermal noise was added to the simulated IMU sensor data along with an integrated random walk bias. Similarly, measurement noise was added to the measurements at each timestep. The standard deviations of the noise processes are given in Table III.4; all but the measurement densities are given in continuous time and are converted to discrete variances by squaring them and dividing by the timestep.

Table III.4: Simulated noise parameters.

w_f	$0.00078 m/s^2/\sqrt{Hz}$	w_p	$0.001 m/\sqrt{Hz}$
w_ω	$0.000523 rad/s/\sqrt{Hz}$	w_z	$0.01 rad/\sqrt{Hz}$
w_{bf}	$0.0001 m/s^3/\sqrt{Hz}$	n_p	0.01 m
$w_{b\omega}$	$0.000618 rad/s^2/\sqrt{Hz}$	n_z	0.01 rad

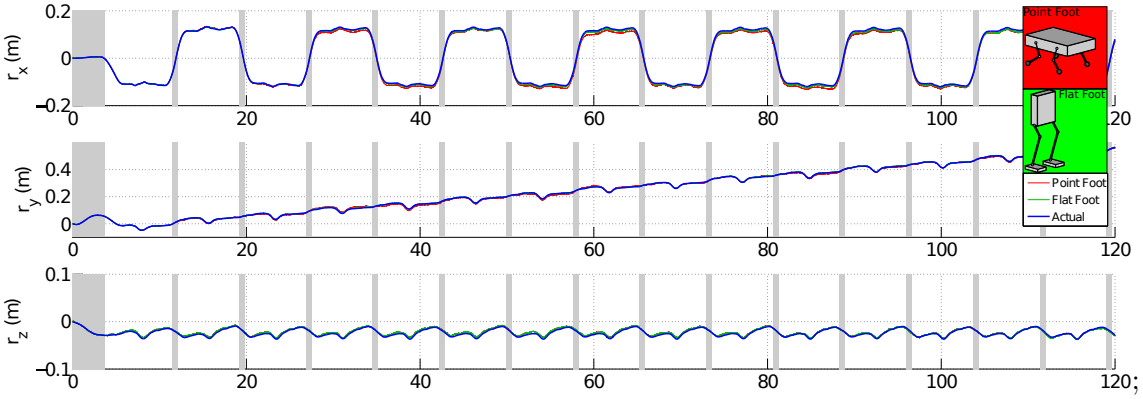
The simulated IMU noise parameters were derived directly from the 3DM-GX3-25 datasheet and the measurement noise parameters were based on the observed uncertainty in the encoders and kinematic model of the actual robot. Experiments were conducted at an update rate of 1kHz as this is the fastest possible IMU streaming rate.

The measurement noise parameters were empirically tuned from their simulated values; all other parameters were set to their simulated values. Initialization of the base orientation was performed using stationary accelerometer measurements, initial foot poses were computed from kinematics and all other states were initialized to zero.

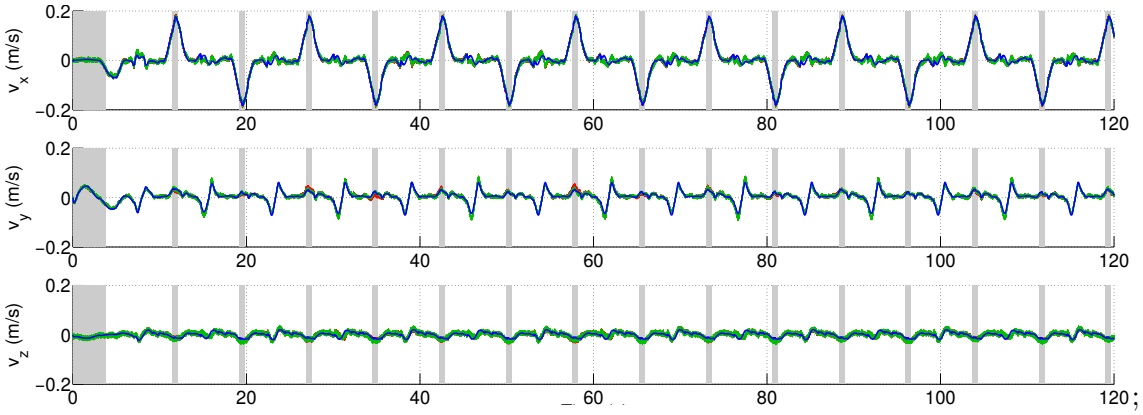
The plots in Figure III.4 show the results obtained on the simulated walking dataset of length 120 seconds. Table III.5 below lists the RMS and maximum errors for all plotted quantities with the last three rows corresponding to the Euler angles roll, pitch and yaw.

Based on the above errors, the two filters perform equally well for many of the plotted quantities. However, investigation of the velocity estimation reveals that the point foot filter periodically diverges more drastically throughout the task, causing the flat foot position estimates to be noticeably more accurate as demonstrated by the plots and error values. Indeed, the maximum absolute errors in v_x, v_y and v_z for the point foot filter are reduced for the flat foot filter as compared to the point foot filter as shown in Table III.5. While small, these repeated periods of divergence can lead to considerably more integrated error over the course of a lengthy task.

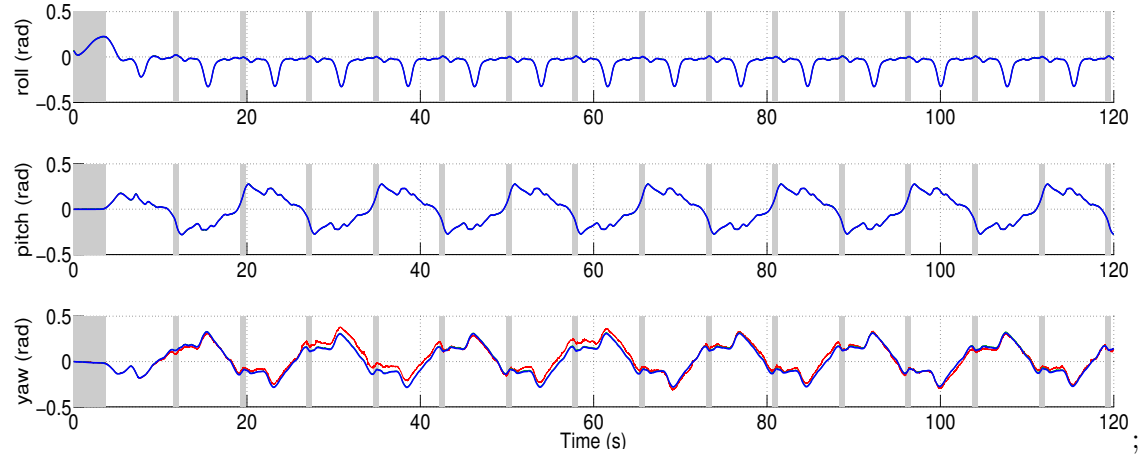
The divergence of the point foot filter in v_y corresponds primarily to the double support portions of the task. There is relatively little base rotation during these intervals since the robot is shifting its center of mass in preparation for the next step; proximity to the $\omega = 0$



(a) Position



(b) Velocity



(c) Euler Angles

Figure III.4: Position, velocity and orientation (Euler angle) estimates for the 120 second walking task. The portions with a white background indicate the single support phase while those with a grey background indicate the double support phase. Overall, the flat foot filter introduced in this work performs better than the filter introduced in (Bloesch et al., 2012) as confirmed by the estimation errors listed in Table III.5.

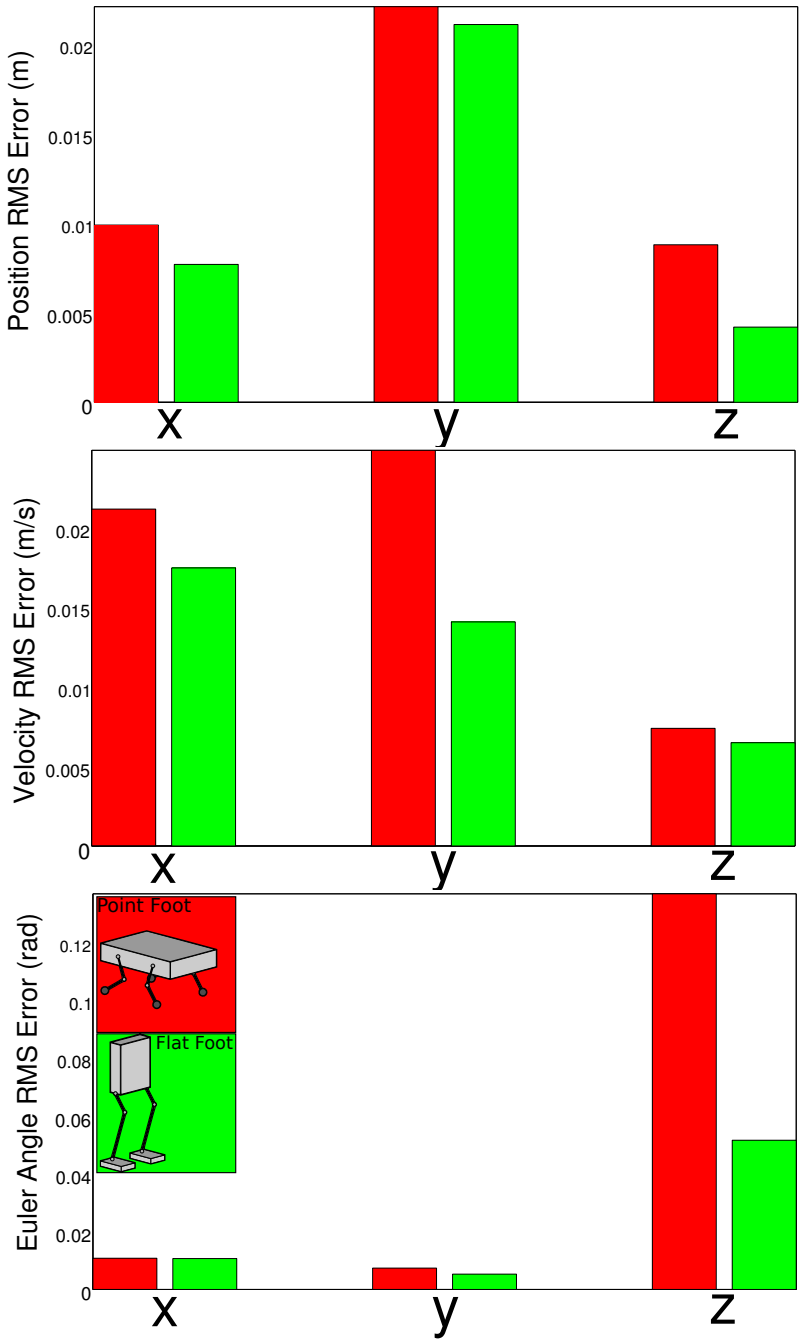


Figure III.5: RMS estimation error bar plots for the Point Foot and Flat Foot filters for the simulated walking task corresponding to those listed in Table III.5.

Table III.5: RMS and maximum (absolute) error values for the 120 second walking task for point and flat foot filters.

	RMS Error		Max Error	
	Point	Flat	Point	Flat
$r_x(m)$	0.0088	0.0042	0.0172	0.0086
$r_y(m)$	0.0046	0.0017	0.0123	0.0047
$r_z(m)$	0.0020	0.0019	0.0051	0.0050
$v_x(m/s)$	0.0079	0.0082	0.0417	0.0393
$v_y(m/s)$	0.0058	0.0053	0.0282	0.0276
$v_z(m/s)$	0.0067	0.0066	0.0357	0.0321
$\alpha(rad)$	0.0011	0.0011	0.0037	0.0038
$\beta(rad)$	0.0010	0.0013	0.0034	0.0046
$\gamma(rad)$	0.0379	0.0055	0.1032	0.0110

singular case may account for the performance difference between the filters since the flat foot filter has a reduced maximum rank loss in this situation. Both filters diverge in v_x during the single support phase, suggesting that one or more singular cases are reached here (for example, the angular velocity becomes nearly orthogonal to the gravity axis when the leg is swung forward). However, it's clear from the plots that the point foot filter provides poorer state estimates during this interval; this is to be expected since the rank loss cases for the flat foot filter are fewer and less drastic during single support. Additionally, the large difference in error values between filters for the yaw angle is explained by the fact that constraining the rotational state renders the gyroscope bias fully observable for the flat foot filter. Finally, note that the setup simulated in this paper employs high-quality sensors and a fast update rate; the difference between the two formulations is expected to be even more pronounced on a robot which has a lower control frequency, a less-accurate kinematic model or low-grade sensors. The effect of hardware on performance remains to be tested in future work by adjusting simulated noise appropriately.

III.4 Inverse Dynamics Experiments

While we have analyzed performance of the base state estimator introduced in this chapter in a simulation environment with realistic levels of noise in Section V.4, we validate the approach on a real humanoid platform in the context of inverse dynamics control. Using the base state estimator of Section III.1 in combination with the controller introduced in Section II.3, we have conducted a number of experiments on our Sarcos humanoid, Hermes (see II.1 for more information on the hardware and low-level control). Here, we present two tasks which highlight the use of Cartesian feedback control in global frame and contact switching.

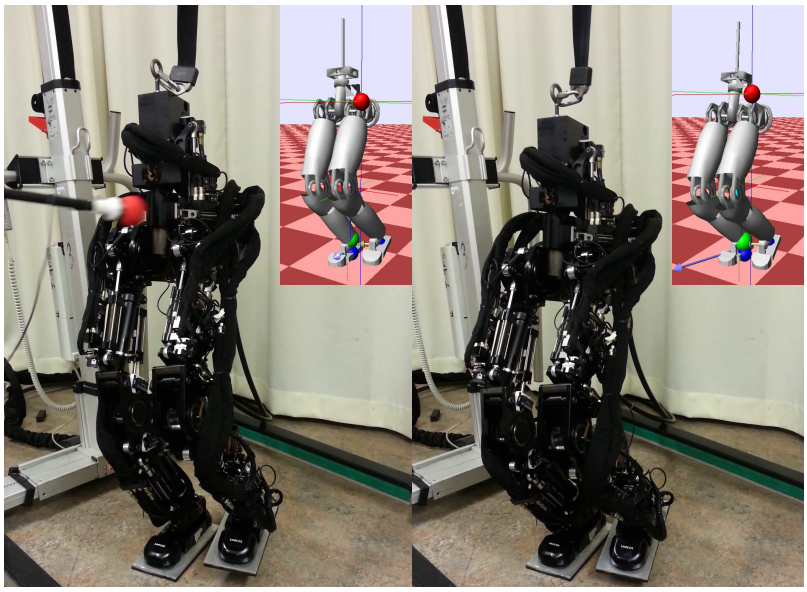


Figure III.6: The Sarcos humanoid during a single legged squatting experiment with perturbations. Inset are screen captures of the SL kinematics visualization updated using the base pose estimator presented in this chapter.

The first task is single support balancing, implemented as COM and swing foot regulation (a desired setpoint for each unconstrained endeffector is specified with zero desired

velocity and acceleration). The transition to single support is achieved using motions created using minimum jerk trajectories (fifth-order polynomials with desired initial and final position and zero velocity and acceleration endpoint constraints). The robot first squats 3cm and then shifts the COM above the stance foot while unloading the swing leg; desired vertical forces are specified in the solver to achieve a linear weight shift. A simple threshold-based contact estimator (with debounce safeguards) is used to trigger the release of the swing foot contact; the estimated state is used to determine the constraints in both the estimator and controller. The results of this experiment are summarized in Figure III.7. COM tracking is good in the x and y directions, however there is an offset in the z direction due to a discrepancy between the mass model and the true robot mass. The shaded portions of the plot indicate pushes applied primarily in the y direction to the robot's base or swing foot using an ATI wrench sensor mounted on a rod; the magnitude (norm) of the push force is shown in the final plot. The foot y position is also plotted to show this endeffector's response to pushes around the 35s mark. The COM control is quite stiff, quickly converging back to within several millimeters of the desired y position even after sustained pushes of upwards of 50N; an integral controller could compensate for such a prolonged disturbance, or more appropriately we could estimate the external wrench using the methods of Chapter IV and add it into the dynamic model when performing inverse dynamics. The swing foot position control is less stiff which is desirable during the swing phase of walking in order to deal with unknown terrain, however the control could be tuned more aggressively if desired.

The second experiment, shown in Figure III.8, is identical to the first for the beginning portion (squatting and shifting COM motions) but the COM is then commanded to move up and down repeatedly in single support while maintaining the swing foot at a specific height. Since the COM and swing foot tracking are traded off with weights in the cost function (see Section II.3), a higher weight on the COM tracking means that the foot moves a small amount (about 1cm in each direction) in order to make the desired COM motion feasible. Since the configuration of the leg affects the position of the COM, these tasks are coupled and thus the swing leg tracking is compromised in order to improve the COM

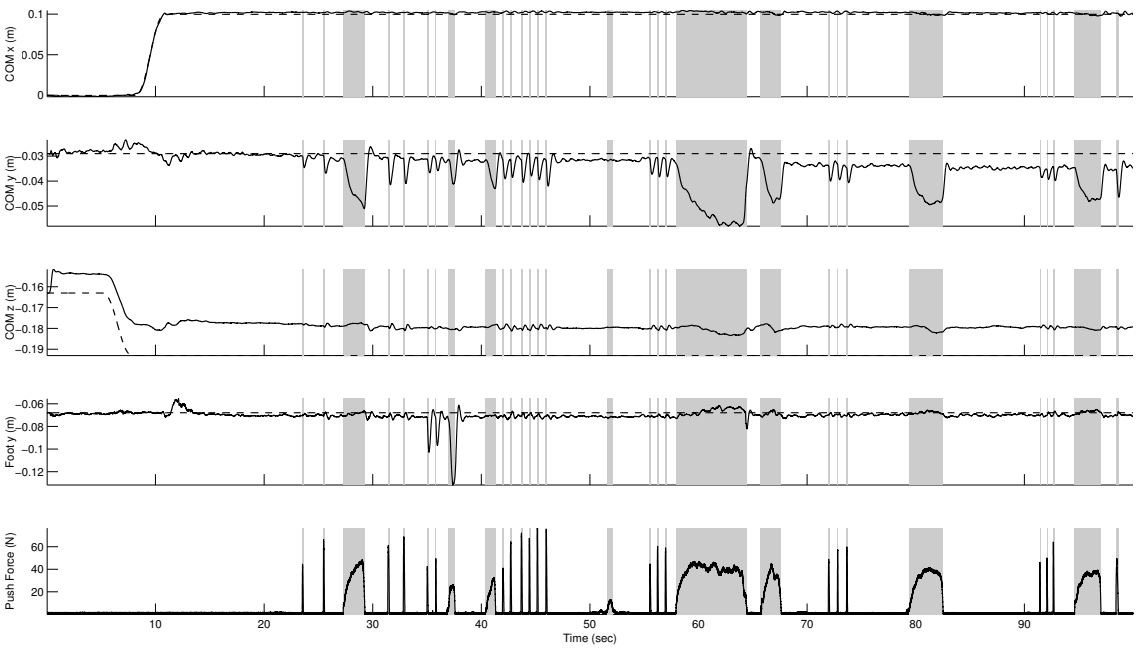


Figure III.7: QP-Based Inverse Dynamics control using the estimated base pose for a single-legged balancing task in which the COM is regulated to a desired position over the stance foot. We disturb the robot with pushes of varying strength and duration as shown in the bottom plot and highlighted elsewhere in grey.

tracking. Again, the COM height as well as the swing foot height are offset due to mass modeling errors, however this could be remedied by adjusting the mass model to correct the gravity compensation portion of the inverse dynamics control. We again disturb the robot at the base and foot during the task, however the motion continues to track well and remains stable.

In both experiments, the release of the left foot contact is seen to be smooth and does not destabilize the estimator or controller. We are currently working to use this controller for walking using an online MPC solver such as (b. Wieber, 2006) and eventually (Herdt et al., 2010) which additionally plans reactive footsteps; both of these MPC solvers have been implemented in simulation and enable robust walking control.

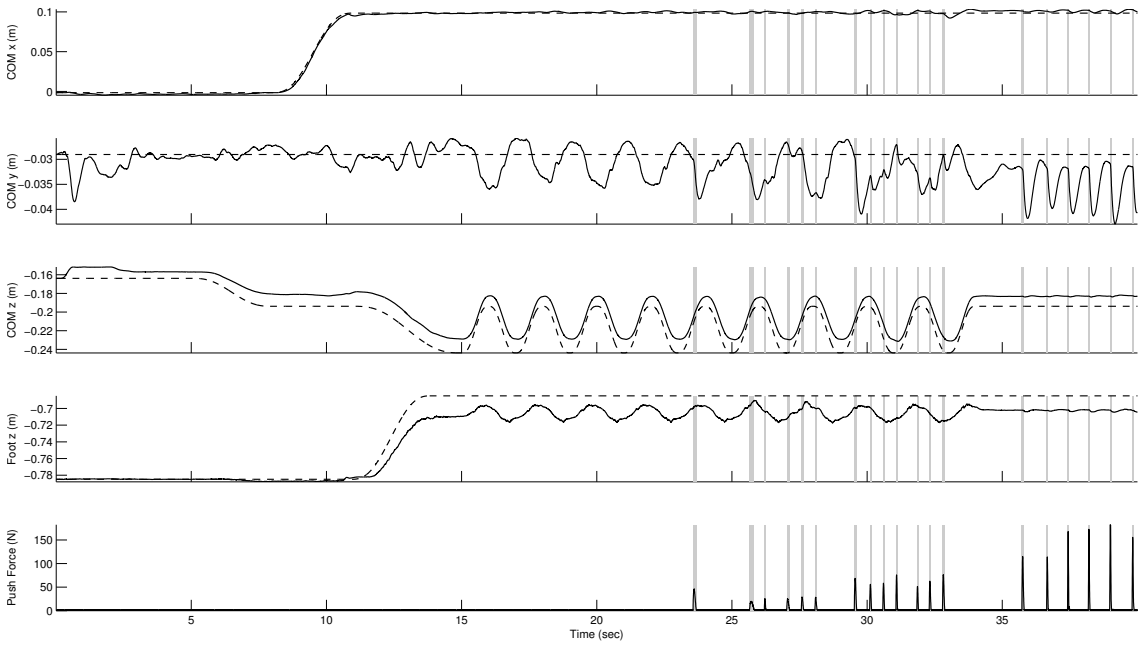


Figure III.8: QP-Based Inverse Dynamics control using the estimated base pose for a single-legged balancing task in which the desired COM moves sinusoidally in the vertical direction while the swing foot is set to maintain a position. Again, we disturb the robot during the portions of the plot having a grey background.

III.5 Summary

In this chapter, we have:

- Developed an EKF based on (Bloesch et al., 2012) which incorporates endeffector rotational constraints for humanoid base pose estimation.
- Described the implementation details related to linearization and discretization of the filter prediction and measurement models (more details on which can be found in the corresponding Appendices).

- Performed a nonlinear observability analysis on the estimator, concluding that the inclusion of flat foot constraints simplifies and reduces the number of motion-dependent cases in which the observability matrix loses rank.
- Conducted experiments in simulation on ZMP-based walking data which demonstrate the superiority of the flat foot-constrained approach presented in this work in terms of estimation error.
- Begun to characterize the performance of our base pose estimator in the whole-body inverse dynamics control framework of Section II.3 on the real robot for single legged balancing tasks with disturbances.

Chapter IV

COM and Momentum Estimation

In the past, researchers (Kwon and Oh, 2007) have recognized the challenge of performing ZMP preview control using inaccurate COM information. Their solution was a Kalman Filter (KF) which fused a LIPM-based process model and the expected motion of the COM, computed from the previewed ZMP and resolved into joint angles. This relied heavily on the expected ZMP, making it suitable only for open-loop walking on simple terrain. Endeffector contact wrench sensors were used but the simplified model limited their utility. Rotational motion (angular momentum) was ignored.

Simplified models were again used for estimation of COM position/velocity, center of pressure (COP), COM offsets and external forces (Stephens, 2011). This work introduced filters based on the LIPM, analyzed their observability and demonstrated their performance. However, the use of a LIPM-based process model compromises the filter optimality which can lead to delays and even destabilize control. In addition, the observability of a simplified model is not the same as that of the original system. Third, use of the COP limits these estimators to planar terrain. Finally, this work does not consider estimation of angular momentum.

Another work (Xinjilefu and Atkeson, 2012) investigated whether simplified models are sufficient for estimation. They fused LIPM dynamics with COM and COP measurements in one estimator and a planar, five-link model with joint angle measurements and inertial measurements from an IMU in another. The estimators were evaluated on simulated data with added biases. However, biases were not modeled explicitly; instead, noise parameters

were tuned to account for them. No observability analysis was performed, making the filters difficult to analyze; they are difficult to compare since they serve different purposes.

In contrast to the above approaches, (Hashlamon and Erbatur, 2013) presented a COM and COP estimator which does not require contact wrench sensors. Contact wrenches and COPs were determined from IMU acceleration by solving an optimization problem using contact constraints. However, they assumed the IMU measures COM acceleration and that the angular momentum is constant. They then presented a LIPM-based KF which uses the computed COP as an input to the LIPM dynamics. The resulting filter, despite providing COP estimates without using contact wrench sensors, demonstrates poor performance likely due to its strong assumptions.

Another recent work (Xinjilefu et al., 2015) focuses exclusively on COM estimation in a manner similar to that in (Stephens, 2011) but with applications for inverse dynamics control on the Atlas humanoid. A COM offset is estimated by a LIPM-based filter and used as a virtual force to compensate for disturbances in inverse dynamics control as well as to compute the capture point of the robot which helps predict falls. This work stands out as the only estimation approach from the DRC which explicitly considers estimation of the COM for use in control.

Finally, (Carpentier et al., 2016) takes a different approach to COM estimation by fusing kinematics and measured contact wrenches in the frequency domain using a complementary filter according to the spectral characteristics of each source of information. An observability analysis is presented and filter is successfully applied on human running data (using a Vicon system and an approximate mass model); application on a humanoid instead would be straightforward. This approach however does not consider momentum or external wrench estimation.

This chapter introduces several different estimators which address the shortcomings of previous approaches by using dynamics consistent with the full robot model to estimate the COM, linear and angular momentum as well as COM and momentum offsets and external wrenches.

IV.1 Models of Robot Dynamics

In recent work, we have implemented a momentum controller derived as a state feedback controller using Linear-Quadratic Regulator (LQR) design and realized using a hierarchical QP-based inverse dynamics solver (Herzog et al., 2014b). Similarly, we can modify the inverse dynamics solver presented in III to control momentum rather than the COM and base pose. We thus propose a momentum estimator to use with this class of controllers based on the same dynamics as given below. Aside from new notation, these equations are identical to Eq (II.5) but summed about the point at which the contact wrench sensor is located rather than about the COP.

$$\dot{c} = \frac{1}{m} l \tag{IV.1}$$

$$\dot{l} = \sum_{i=1}^{M} (F_i + w_{F_i}) + mg \tag{IV.2}$$

$$\dot{k} = \sum_{i=1}^{M} (p_i - c) \times (F_i + w_{F_i}) + \sum_{i=1}^{M} (\tau_i + w_{\tau_i}) \tag{IV.3}$$

Here c, l and k are the COM, linear and angular momentum respectively; p_i, F_i and τ_i are the i^{th} contact point position, force and torque respectively. The vectors w_{F_i} and w_{τ_i} represent additive white Gaussian sensor noise processes with standard deviations q_F and q_τ. These dynamics, illustrated in Figure IV.1, are preferable to the full dynamics for prediction because they avoid using inaccurate link inertia models and instead use only the total mass m, which can be measured.

The estimators in (Stephens, 2011) are derived from the LIPM dynamics

$$\ddot{x} = \frac{g}{z_c}(x - x_{COP}) \tag{IV.4}$$

where x denotes the COM position, z_c is the (constant) height of the COM and x_{COP} denotes the COP position. The dynamics of the y direction are exactly the same since the LIPM is decoupled. This equation can be derived from (IV.2) and (IV.3) by replacing

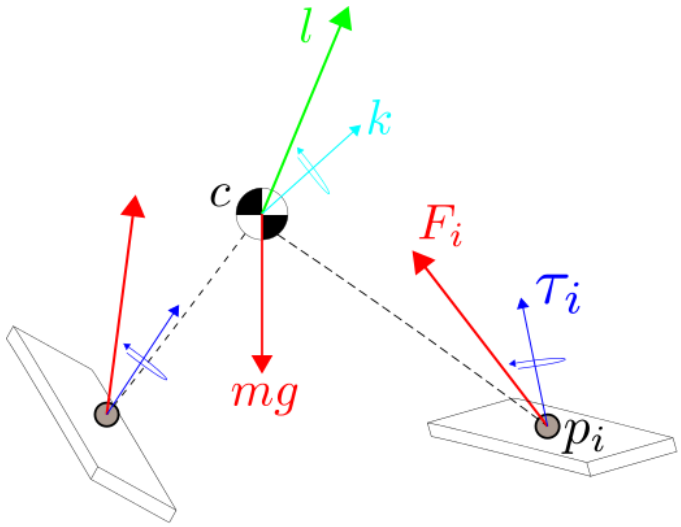


Figure IV.1: Illustration of the momentum dynamics.

p_i with the COP, neglecting rotation and constraining motion to a plane as was done in Chapter II to obtain Eq (VI.2).

For use with less-dynamic controllers, an estimator based on (IV.4) may prove sufficient. Most previous studies hence used the LIPM to avoid using the full dynamics. However, for linear systems it can be shown (Friedland, 1995) that an inaccurate process model can lead to unstable estimation error dynamics; stability proofs assume exact plant cancellation. Further, an inaccurate model can destabilize both the controller and observer. This is because the separation principle - which says that the controller and observer can be designed independently - no longer holds. Of course, since the true dynamics are nonlinear, these theorems can only be applied qualitatively to inform design. This suggests avoiding simplified models for estimation when possible.

IV.2 Estimators

In this section, we introduce four estimators based on the momentum dynamics, each serving a different purpose. We choose to implement these using Extended Kalman Filters,

however the observability results of Section IV.3 hold for any implementation. Each estimator's nonlinear process dynamics are used for prediction and linearized for the update step as follows.

$$\begin{aligned} \dot{x} = f(x, w) \quad &\rightarrow \quad \dot{x} = Ax + Lw \\ y = h(x, w) \quad &\rightarrow \quad y = Cx + v \end{aligned}$$

where x, w and v are the state, process noise, and measurement noise vectors and $A = \frac{\partial f}{\partial x}$, $L = \frac{\partial f}{\partial w}$ and $C = \frac{\partial h}{\partial x}$ are the relevant Jacobians. While the controls F and τ are used in prediction and in forming Jacobians, they do not explicitly appear in the EKF formulation. We specify estimators by their nonlinear process and measurement models and the Jacobians A, L and C resulting from linearization.

IV.2.1 Momentum Estimator

On our humanoid, joint velocities are computed by differentiating noisy potentiometer signals; momentum is computed from kinematics using these noisy velocities and base information from a base-state estimator developed in previous work (Rotella et al., 2014)*. We currently filter momentum using second-order Butterworth low-pass filters but this introduces delays. By integrating low-noise measured contact wrenches and using the noisy kinematics computations as measurements, we can filter momentum without inducing as large a delay.

We choose the state $x = [c, l, k]$ so the process model is (IV.1)-(IV.3) and the measurement model is $y = [c, k]$; we measure the COM and angular momentum computed from base information, kinematics and inertias. These measurements have noise standard

*Note that since the COM and momentum are functions of the floating base pose and its velocity, we could choose to combine the base state estimator with the COM and momentum estimator proposed here. However, we wish to decouple estimation of the base pose (which is a kinematic quantity) from estimation of the momenta (which are functions of the dynamic model).

deviations r_c and r_k, respectively. Although inaccurate inertias are used to compute k, we demonstrate robustness to model errors in Section IV.4. The dynamics are

$$A = \begin{bmatrix} 0 & \frac{1}{m}I & 0 \\ 0 & 0 & 0 \\ \sum \bar{F}_i^{\times} & 0 & 0 \end{bmatrix}, \quad L_i = \begin{bmatrix} 0 & 0 \\ I & 0 \\ (\bar{p}_i - \bar{c})^{\times} & I \end{bmatrix}$$
$$C = \begin{bmatrix} I & 0 & 0 \\ 0 & 0 & I \end{bmatrix}$$

where $a^{\times}$ denotes the cross product matrix formed from vector a. The noise Jacobian L is formed by concatenating L_i horizontally for all $i = 1 \dots M$ contacts. $\bar{F}_i$, $\bar{p}_i$ and $\bar{c}$ are the measured force and the foot and COM positions, respectively, treated as constants over a single timestep.

IV.2.2 Offset Estimator

The kinematic model of a robot is often imprecise. Here, we assume the total mass m is known but that incorrect link masses and COM locations contribute to a configuration-dependent COM offset. These errors also create offsets in momentum; we thus extend the state to $x = [c, l, k, \Delta c, \Delta l]$ where Δc and Δl are the COM and linear momentum offset vectors, respectively. In theory, there is also an offset Δk but this is unobservable (see Section IV.3) so we choose not to estimate it. Since the offset dynamics are unknown, we assume random walks so the prediction dynamics are (IV.1)-(IV.3) plus $\Delta \dot{c} = w_{\Delta c}$ and $\Delta \dot{l} = w_{\Delta l}$ where $w_{\Delta c}$ and $w_{\Delta l}$ are additive Gaussian white noise processes with standard deviations $q_{\Delta c}$ and $q_{\Delta l}$. The measurement model is

$$y = \begin{bmatrix} c + \begin{bmatrix} \Delta c_x \\ \Delta c_y \\ 0 \end{bmatrix} \\ l + \Delta l \\ k \end{bmatrix}$$

where the linear momentum measurement noise has standard deviation r_l. This equation indicates that the kinematics-based COM and linear momentum measurements contain offsets. It is shown in Section IV.3 that Δc_z is unobservable and is thus ignored. The linearized dynamics are given by

$$A = \begin{bmatrix} 0 & \frac{1}{m}I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \sum \bar{F}_i^{\times} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, L_i = \begin{bmatrix} 0 & 0 \\ I & 0 \\ (\bar{p}_i - \bar{c})^{\times} & I \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} I & 0 & 0 & I_2 & 0 \\ 0 & I & 0 & 0 & I \\ 0 & 0 & I & 0 & 0 \end{bmatrix}$$

where $I_2 = \text{diag}(1, 1, 0)$ since we ignore Δc_z. Because the dynamics of Δc_z affect the estimate of l_z, it is best to ignore this offset and accept that it cannot be observed; this is why we write the measurement this way. Note that we do not explicitly consider link inertia errors, however these would contribute only to the unobservable offset Δk.[†]

IV.2.3 COP-Based Offset Estimator

In the above estimator, Δc_x and Δc_y are observable due to the dynamics of the angular momentum measurement (see Section IV.3 for details). However, this measurement is subject to an offset Δk. While ignored above, this leads to degraded performance for significant modeling errors. We add the COP measurement to give us force-based information about the COM which is accurate despite modeling errors. This comes at the disadvantage of assuming coplanar contacts.

We use the same state and add the COP measurement (with noise standard deviation r_{COP}). We keep the measurement of k and increase r_k relative to r_{COP} in order to filter

[†]Unobservable states can drift arbitrarily; when coupled to other states through their dynamics, this drift affects their estimation. Because of this, adding an unobservable offset Δk would corrupt the information about Δc which is provided by the angular momentum measurement. We are better off incorrectly assuming that $\Delta k = 0$ and accepting that our estimates of Δc may be inaccurate depending on the severity of the modeling errors. The filter of Section IV.2.3 addresses this.

angular momentum while relying primarily on the COP to render Δc_x and Δc_y observable. The measurement model is augmented with the measurement

$$y_{COP,x} = c_x - \frac{1}{\sum F_{i,z}} \left(c_z \sum F_{i,x} + \dot{k}_y \right)$$
$$y_{COP,y} = c_y - \frac{1}{\sum F_{i,z}} \left(c_z \sum F_{i,y} - \dot{k}_x \right)$$

where $\dot{k} = \sum (p_i - c) \times F_i + \sum \tau_i$. The measurement Jacobian is

$$C = \begin{bmatrix} I & 0 & 0 & I_2 & 0 \\ 0 & I & 0 & 0 & I \\ 0 & 0 & I & 0 & 0 \\ H & 0 & 0 & 0 & 0 \end{bmatrix}, \quad H = \begin{bmatrix} 2 & 0 & \frac{-2\sum \bar{F}_{i_x}}{\sum \bar{F}_{i_z}} \\ 0 & 2 & \frac{-2\sum \bar{F}_{i_x}}{\sum \bar{F}_{i_z}} \end{bmatrix}$$

where H is the Jacobian relating the COM to the COP.

IV.2.4 External Wrench Estimator

It is often the case that contact force/torque (F/T) sensors drift or disturbances are applied to the robot. We begin with the Momentum Estimator state and introduce an external wrench acting at the COM which encapsulates these errors. The state becomes $x = [c, l, k, \Delta F, \Delta \tau]$ where $\Delta F, \Delta \tau$ are the external force and COM torque (torque computed about the COM). Again, we assume random walks for each so the dynamics are

$$\begin{aligned} \dot{c} &= \frac{1}{m} l \\ \dot{l} &= \sum_{i=1}^{M} (F_i + w_{F_i}) + mg + \Delta F \\ \dot{k} &= \sum_{i=1}^{M} (p_i - c) \times (F_i + w_{F_i}) + \sum_{i=1}^{M} (\tau_i + w_{\tau_i}) + \Delta \tau \\ \Delta \dot{F} &= w_{\Delta F} \\ \Delta \dot{\tau} &= w_{\Delta \tau} \end{aligned}$$

where $w_{\Delta F}$ and $w_{\Delta \tau}$ have standard deviations $q_{\Delta F}$ and $q_{\Delta \tau}$. The measurement model is $y = [c, k]$ as in the original Momentum Estimator. The linearized dynamics are

$$A = \begin{bmatrix} 0 & \frac{1}{m}I & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 \\ \sum F_i^{\times} & 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, L_i = \begin{bmatrix} 0 & 0 \\ I & 0 \\ (\bar{p}_i - \bar{c})^{\times} & I \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \end{bmatrix}$$

IV.3 Observability Analysis

This section investigates the observability of each of the estimators proposed in the previous section. Since the nonlinearity of the process models is introduced through the product of the state and input (contact wrench) noise, and since observability is analyzed by assuming the noise is zero, the proposed estimators can be analyzed using observability analysis for linear systems. However, the nonlinear observability analysis process produces the same results. Details on analyzing observability for both linear and nonlinear systems can be found in Appendix C.

IV.3.1 Momentum Estimator

The observability matrix is

$$O = \begin{bmatrix} I & 0 & 0 \\ 0 & 0 & I \\ 0 & \frac{1}{m}I & 0 \\ \sum F_i^{\times} & 0 & 0 \\ 0 & \frac{1}{m}\sum F_i^{\times} & 0 \end{bmatrix}$$

This matrix has full rank so the state is observable. We could add a kinematics-based linear momentum measurement but it would be redundant since it is computed from the

same sensors as the COM. Since linear momentum is a function of noisy velocities, it is subject to considerable noise; it is better to leave it out and let differentiation occur in the filter.

IV.3.2 Offset Estimator

The observability matrix is given below; we include the full Δc in the state to prove that it cannot be observed.

$$O = \begin{bmatrix} I & 0 & 0 & I^* & 0 \\ 0 & I & 0 & 0 & I \\ 0 & 0 & I & 0 & 0 \\ 0 & \frac{1}{m}I & 0 & 0 & 0 \\ \sum F_i^\times & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m}\sum F_i^\times & 0 & 0 & 0 \end{bmatrix}$$

This matrix is rank deficient; the unobservable subspace is a linear combination of the COM and COM offset. This occurs because the skew-symmetric matrix $\sum F_i^\times$ has at most rank two. In theory, this means we could observe any two directions of the COM offset, effectively replacing I^* in O. However, when the robot is stationary on flat ground,

$$\sum F_i^\times = \begin{bmatrix} 0 & -mg & 0 \\ mg & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

forcing the z direction to be unobservable. Either the x or y direction can be made unobservable in its place only when $\sum F_{i,x}$ or $\sum F_{i,y}$ is nonzero. This occurs only when an force acts on the robot in these directions at the COM (when the robot accelerates or is on a slope so gravity affects x and/or y). For this reason, we set $I^* = I_2 = \text{diag}(1, 1, 0)$ so that Δc_x and Δc_y are observable. Note that $\sum F_i = 0$ for a robot in flight phase; in this case, Δc is completely unobservable.

IV.3.3 COP-Based Offset Estimator

The observability matrix is

$$O = \begin{bmatrix} I & 0 & 0 & I_2 & 0 \\ 0 & I & 0 & 0 & I \\ 0 & 0 & I & 0 & 0 \\ H & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m}I & 0 & 0 & 0 \\ \sum F_i^{\times} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m}\sum F_i^{\times} & 0 & 0 & 0 \\ 0 & \frac{1}{m}H & 0 & 0 & 0 \end{bmatrix}$$

Observability is the same as for the Offset Estimator; however, the COP provides information about the COM derived from F/T sensors rather than from kinematics. This improves offset estimation in the case when the unmodeled angular momentum offset is large. This makes the important point that adding redundant measurements can affect performance even though observability is unchanged.

IV.3.4 External Wrench Estimator

The observability matrix is

$$O = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & \frac{1}{m}I & 0 & 0 & 0 \\ \sum F_i^{\times} & 0 & 0 & 0 & I \\ 0 & 0 & 0 & \frac{1}{m}I & 0 \\ 0 & \frac{1}{m}\sum F_i^{\times} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{m}\sum F_i^{\times} & 0 \end{bmatrix}$$

This matrix has full rank - since the external force and torque appear in the momentum dynamics, and since momentum is observable through the COM and angular momentum measurements, the external wrench is observable.

IV.3.5 Offset and External Wrench Estimator

If we extend the Offset Estimator state so that $x = [c, l, k, \Delta c, \Delta l, \Delta F, \Delta\tau]$ then we obtain the matrix

$$O = \begin{bmatrix} I & 0 & 0 & I^* & 0 & 0 & 0 \\ 0 & I & 0 & 0 & I & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m}I & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ \sum F_i^\times & 0 & 0 & 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{m}I & 0 \\ 0 & \frac{1}{m}\sum F_i^\times & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{m}\sum F_i^\times & 0 \end{bmatrix}$$

This matrix is rank deficient; its nullspace is a linear combination of c, Δc and $\Delta\tau$. This implies that we can never observe both the offsets and an external wrench simultaneously. However, if the external force is applied at the COM then $\Delta\tau = 0$ and the state (except Δc_z) becomes observable.

IV.4 Experiments and Results

All results presented in this work were obtained in the SL simulation environment (Schaal, 2007). White Gaussian noise was generated based on data from Hermes, discretized at the filtering frequency and added to simulated sensor outputs. Noise having standard deviation q_θ was added to joint angles (and propagated to joint velocities through numerical differentiation) as well as to endeffector F/T sensor signals. Table VI.1 below lists the values of the standard deviations of the simulated noise processes.

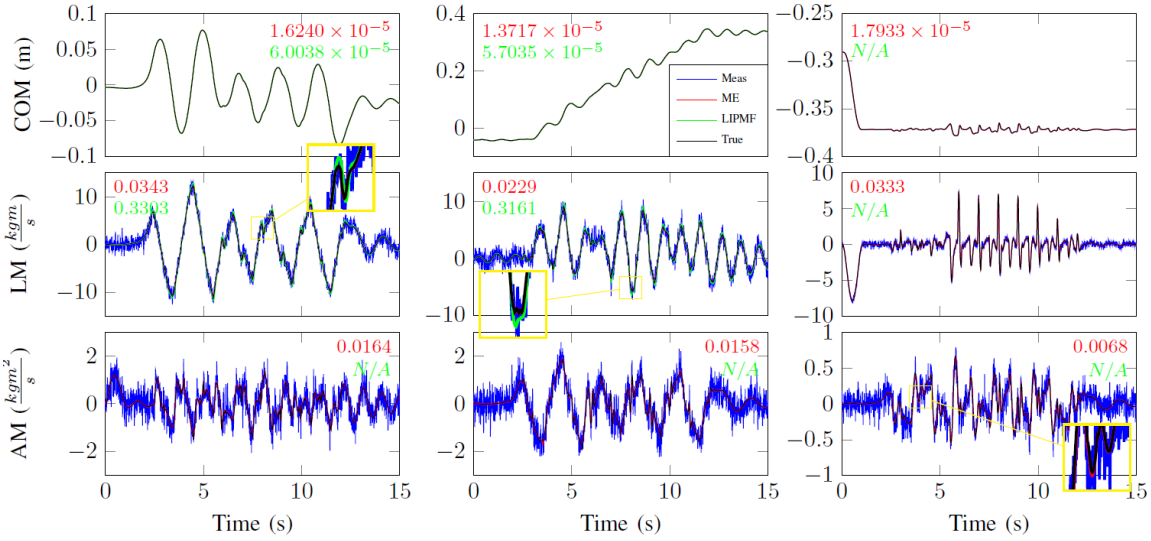


Figure IV.2: Estimation of COM (top row), linear momentum (middle row) and angular momentum (bottom row); columns denote x,y and z. RMS errors for the relevant filters are shown on each plot.

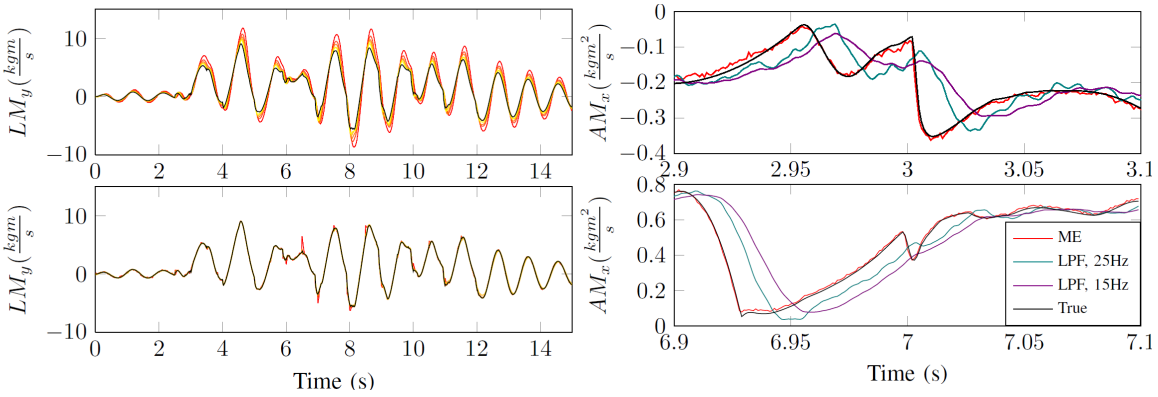


Figure IV.3: Estimation of linear momentum for increasing frequencies (Top: LIPM Filter. Bottom: Momentum Estimator)

Figure IV.4: Zoomed-in views of angular momentum estimates versus low-pass filtered kinematics measurements.

Table IV.1: Simulated sensor noise standard deviations. Corresponding values for 1kHz sampling rate are shown.

	Continuous	Discrete ($1kHz$)
q_θ	$0.00000316rad/\sqrt{Hz}$	$0.0001rad$
q_F	$0.06325N/\sqrt{Hz}$	$2N$
q_τ	$0.00316Nm/\sqrt{Hz}$	$0.1Nm$

In this and subsequent sections, we refer to the four filters based on the momentum dynamics as the ME (Momentum Estimator), OE (Offset Estimator), COE (COP-Based Offset Estimator) and EWE (External Wrench Estimator). Process noise parameters were set using the values in Table VI.1. All other noise parameters were tuned for each filter and are summarized in Table IV.2. Note that measurement noise standard deviations are specified in discrete time. All filters based on the LIPM are referred to as LIPMF in this section and denote the corresponding filter and noise parameters introduced in (Stephens, 2011).

Estimation was performed during a $15s$ ZMP preview control-based walking task (Kajita et al., 2003) having single and double support phases lasting $0.5s$ each and a forward motion of $5cm$ per step for 10 steps. Since this is a relatively-dynamic gait, the contacts created are often not completely flat and subject to impulsive contact wrenches; this was desired in order to test the estimators with realistic contact switching. Ideal base state estimation was assumed for simplicity. Estimation and data recording were performed at $1kHz$ unless noted.

IV.4.1 Momentum Estimator

Both the LIPMF and ME accurately estimate the COM well as shown in Figure IV.2 (note that the LIPMF does not estimate COM_z, LM_z and AM). This is expected since the COM

Table IV.2: Noise standard deviations for momentum-based estimators. N/A indicates that a parameter is not used.

Name (Units)	ME	OE	COE	EWE
$q_{\Delta c}(m/\sqrt{Hz})$	1.0	1.0	1.0	N/A
$q_{\Delta l}(kgm/s/\sqrt{Hz})$	1.0	1.0	1.0	N/A
$q_{\Delta F}(N/\sqrt{Hz})$	N/A	N/A	N/A	1.0
$q_{\Delta \tau}(Nm/\sqrt{Hz})$	N/A	N/A	N/A	0.1
$r_c(m)$	0.0001	0.001	0.001	0.00001
$r_l(kgm/s)$	N/A	1.0	1.0	N/A
$r_k(kgm^2/s)$	0.1	1.0	10.0	0.01
$r_{COP}(m)$	N/A	N/A	1.0	N/A

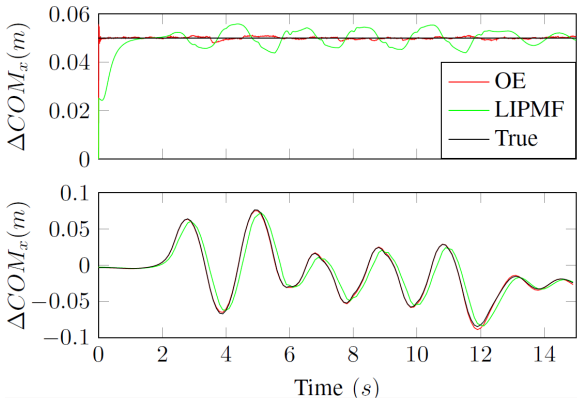


Figure IV.5: Estimation of a 5cm COM offset while stationary (top) and estimation of a configuration-dependent COM offset during the $15s$ walking task (bottom).

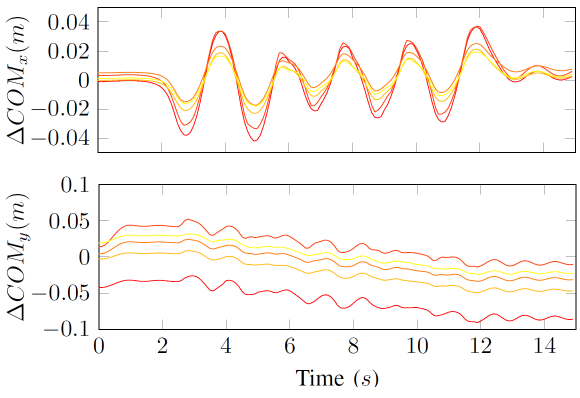


Figure IV.6: Time-varying COM offsets for different n (representing different degrees of modeling error) in x-direction (top) and y-direction (bottom.

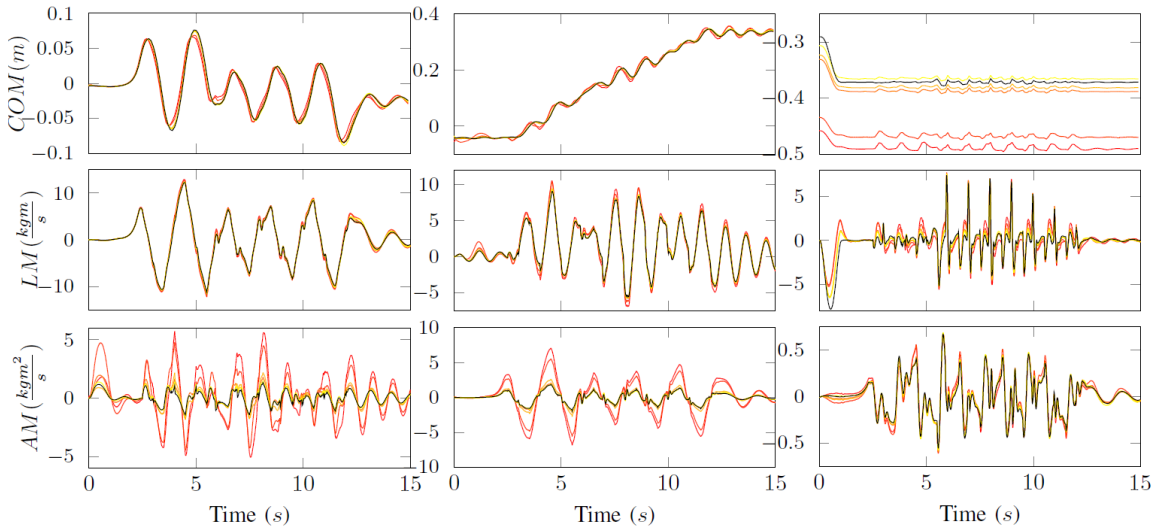


Figure IV.7: Estimation of COM (top row), linear momentum (middle row) and angular momentum (bottom row) for different n (representing different degrees of modeling error). Red denotes the largest modeling error while yellow denotes the least.

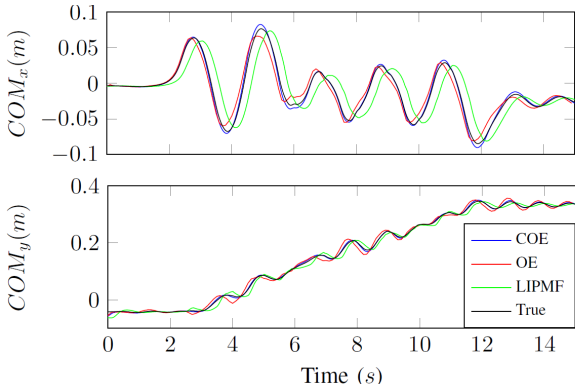


Figure IV.8: COM estimation for $n = 5$ in x-direction (top) and y-direction (bottom).

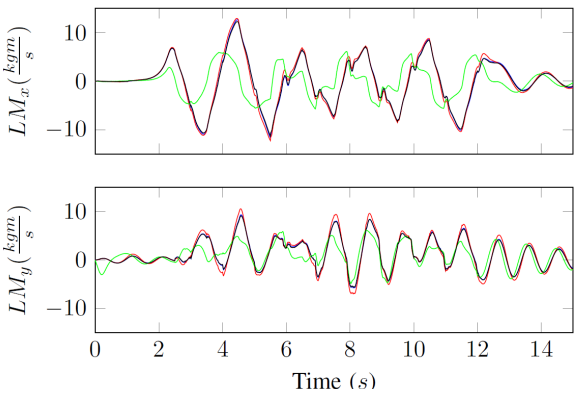


Figure IV.9: Linear momentum estimation for $n = 5$ in x-direction (top) and y-direction (bottom).

is measured directly and r_c was chosen small in both. However, linear momentum estimation is significantly better using the ME since the dynamics of this state are inaccurate in the LIPMF; the given RMS errors demonstrate this.

The difference in performance between the two filters is more pronounced when they are updated at slower rates. Figure IV.3 shows estimation of l_y by each estimator for update rates of $50Hz$, $125Hz$, $250Hz$, $500Hz$ and $1000Hz$ (lower frequencies are plotted in red, higher ones in yellow and ground truth in black). The degradation of performance with decreases in update rate is much more severe for the LIPMF.

The ME was motivated by the desire to avoid low-pass filtering computed momentum. Figure IV.4 shows the kinematics-based angular momentum filtered using different cutoffs and compared to the estimated angular momentum. While the estimate lags by several milliseconds, the delays induced by the Butterworth filters are as large as $20ms$. For dynamic motions, filtering can even change the structure of the signal.

IV.4.2 Offset Estimator

First, a constant offset of $\Delta c_x = 5cm$ was added directly to the COM measurement, as was done in (Stephens, 2011) (assuming Δl and Δk remain zero). Both filters converge to the true COM, though the OE provides more accurate estimates as shown in the top plot of Figure IV.5. This is likely due to the fact that a physically-consistent simulator and realistic sensor noise are used here, unlike in (Stephens, 2011). Next, configuration-dependent offsets in the COM and momentum were created using a set of perturbed link parameters. The LIPMF manages to estimate the COM but with delays up to $100ms$ as shown in the bottom plot of Figure IV.5. This is the result of using simplified dynamics and was not improved with tuning.

The fact that the OE can track the COM and linear momentum offsets despite the unmodeled angular momentum offset suggests robustness to unmodeled errors. In order to analyze this we estimate offsets for five different sets of link mass parameters. Each was generated by adding to every mass-weighted link COM position $m_i x_{COM_i}$ an offset drawn from a zero-mean Gaussian having standard deviation $n(m_i x_{COM_i})$ with $n = 1, 2, 3, 4, 5$.

Figure IV.6 shows the generated configuration-dependent COM offsets throughout the walking task, with red denoting the most perturbed parameters ($n = 5$). The COM offset is as large as $4cm$ in x and nearly twice that in y for $n = 5$. Each perturbed model also results in offsets in c_z and momentum but these are not shown to save space. Figure IV.7 shows the performance of the OE for the different models; black denotes ground truth. Note that angular momentum estimation is poor yet COM and linear momentum estimation remain relatively accurate even for larger values of n, demonstrating OE robustness.

IV.4.3 COP-Based Offset Estimator

It is clear that performance of the OE is degraded for significant modeling errors due to the unmodeled angular momentum offset. By including the force-based COP measurement and tuning r_{COP} against r_k, we achieve accurate COM and linear momentum estimation for $n = 5$ as shown in Figures IV.8 and IV.9. Also shown are the corresponding LIPMF estimates; the LIPMF estimates the COM fairly well but with significant delay while linear momentum estimation is poor due to the unmodeled offset which is significant for $n = 5$.

IV.4.4 External Wrench Estimator

We first estimate a constant external force equal to half the robot's weight as in (Stephens, 2011). This is applied at the left hip of the robot in the y-direction, creating associated torques in x and z. The top row of Figure IV.10 shows that the EWE quickly converges to the true external force and torque. The LIPMF converges to the wrong value, likely because unlike in (Stephens, 2011), the force is physically applied and there is realistic sensor noise. This could not be improved with tuning.

The second row of Figure IV.10 shows estimation of a $10N$ force applied in the same manner during the walking task. Despite the fact that the robot is subject to impulsive forces throughout, the EWE provides accurate estimates of both external force and the small, configuration-dependent COM torques. The LIPMF has difficulty estimating the value of the force due to impulses resulting from contact switching.

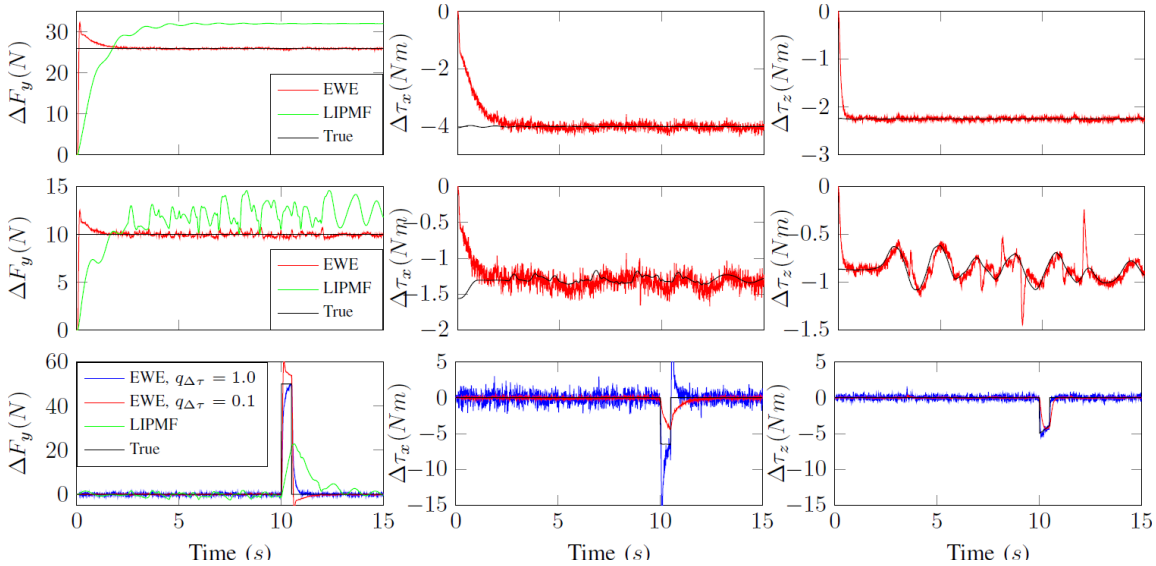


Figure IV.10: External wrench estimation for a constant external force in global y-direction while stationary (top row), while walking (middle row) and for an impulsive force (bottom row) while walking (with process noise values of 0.1 and 1.0)

Finally, the bottom row of IV.10 shows estimation of a $50N$ disturbance applied at time $10s$ for $0.5s$ during the walking task. The EWE overshoots the force value but performs much better than the LIPMF overall, exhibiting a fast rise time. As shown in blue, EWE estimation during the transient disturbance is improved by increasing the external torque process noise. However, estimates become much noisier.

IV.5 Closed-Loop Control Using Momentum Estimators

Ultimately, the purpose of the estimators introduced in this chapter is to improve whole-body control in the presence of modeling errors or applied external wrenches. We now detail how to use these estimated quantities for control.

IV.5.1 Offset Estimators

The COM and linear momentum offsets resulting from either the OE or the COE can be used in a number of ways. The most straightforward approach is to use the resulting COM and linear momentum estimates to adjust the desired COM position and velocity, respectively, in the inverse dynamics controller introduced in Section II.3. Alternatively, since the effect of a pure COM offset is a torque about the model COM (since the force due to gravity acts at a different distance from the model COM), this offset can be treated as pure moment in an additional wrench

$$\lambda_{ext} = \begin{bmatrix} 0 \\ mg\Delta c \end{bmatrix}$$

to be compensated in inverse dynamics as

$$M_l(q)\ddot{q} + h_l(q, \dot{q}) = J_{c,l}^T \lambda + J_{ext}^T \lambda_{ext} \tag{IV.5}$$

where $J_{ext} = \begin{bmatrix} I & J_{c,l} \end{bmatrix}$. This is essentially the approach taken by Xinjilefu et al. (2015), however the COM offset was estimated from a LIPM-based estimator (similar to the LIPMF of this chapter); using the LIPM dynamics, a COM offset can be written as a force in $m\omega^2\Delta c$ where $\omega = \sqrt{g/z_c}$ and z_c again is the constant COM height. This creates a moment about the COP equivalent to λ_{ext} above if z_c is the true COM height.

IV.5.2 External Wrench Estimator

In the previous section, we informally proved that a COM offset has the same effect on the robot dynamics as an external moment at the COM; this is in agreement with the observability analyses of Section IV.3, which showed that it is impossible to distinguish between a COM offset and an external wrench. We thus treat the combined effect of potential modeling error and forces applied at arbitrary locations on the robot as a single external wrench λ_{ext} at the COM, which we seek to estimate and compensate in inverse dynamics as in (IV.5).

Simulation Experiments

In order to evaluate the proposed method of compensating for an external wrench, we first apply simulated forces to the robot while updating the EWE as in Section IV.4.4, however neglecting simulated sensor noise for simplicity. A force is applied at the robot base in each Cartesian direction (in base frame), with increasing amplitude over a short time; this force results in an external wrench about the COM which is estimated and compensated using (IV.5). Figure IV.11 shows the applied and estimated wrench, which is estimated accurately and with a fast response time. This allows the robot to compensate for the increasing load in all directions. The same load was applied to the robot without the use of external wrench compensation in inverse dynamics; the resulting COM tracking both with and without load compensation is shown in Figure IV.12.

There is a lag in the estimation which causes the COM to move slightly even when using the EWE, however the deviation is much smaller than when external wrench compensation is not used. Note that the amount of deviation of the COM from its desired state is a function of the chosen Cartesian feedback gains; by using an external wrench estimator such as the one proposed in this work, we can theoretically lower the COM control gains while achieving the same robustness to perturbations. Modulating the feedforward torque through estimation is desirable over stiff control because high gains can lead to instability.

The effect of a load is much more pronounced at higher load amplitudes - consider an experiment in which an increasing external force in the direction opposite gravity simulates adding 100N of additional weight (which is roughly one-fifth the total weight of the robot) in 20N increments (for example, handing the robot packages to carry). This is done while the robot is walking in place; without compensating for the wrench estimated by the EWE, the robot quickly goes unstable as shown in Figure IV.13.

Real Robot Experiments

When controlling the real robot, one must deal with a variety of issues - both modeling errors which create COM and momentum offsets as well as external wrenches caused by

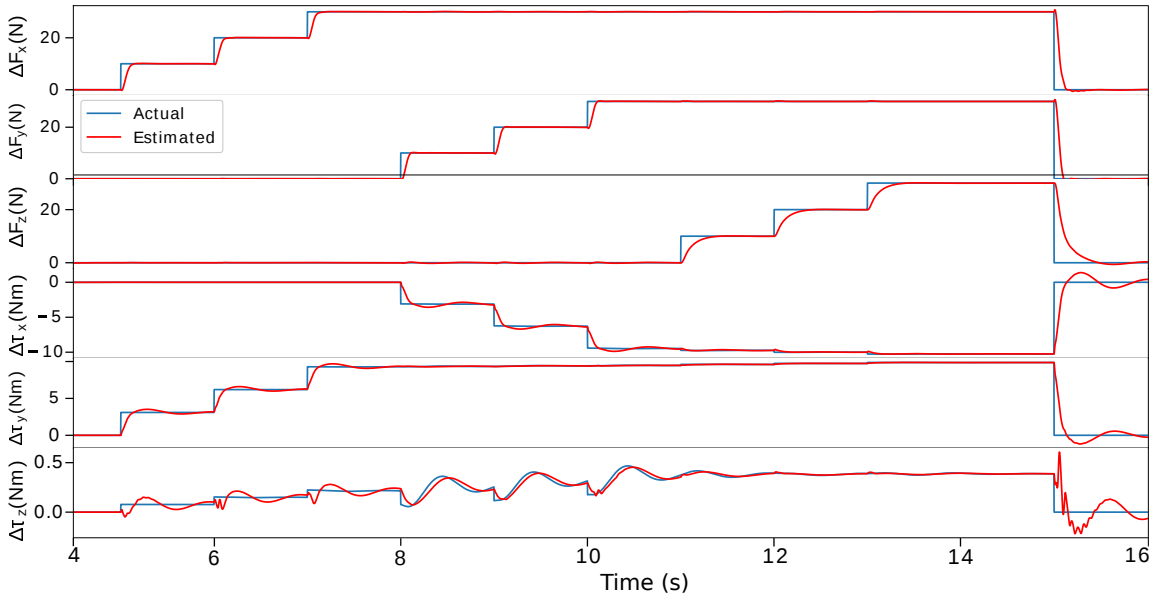


Figure IV.11: External wrench estimation for an increasing simulated load at the robot base in different directions. The estimated wrench is actively being compensated in the inverse dynamics controller.

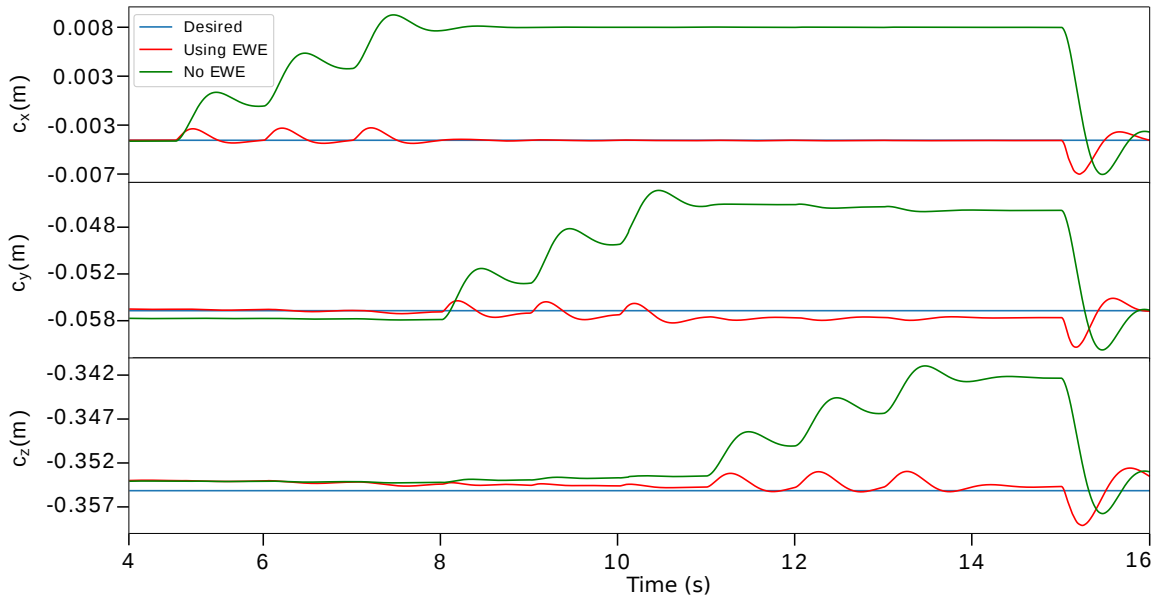


Figure IV.12: External wrench estimation for an increasing simulated load at the robot base in different directions. Load compensation prevents the robot from deviating from its desired COM position.

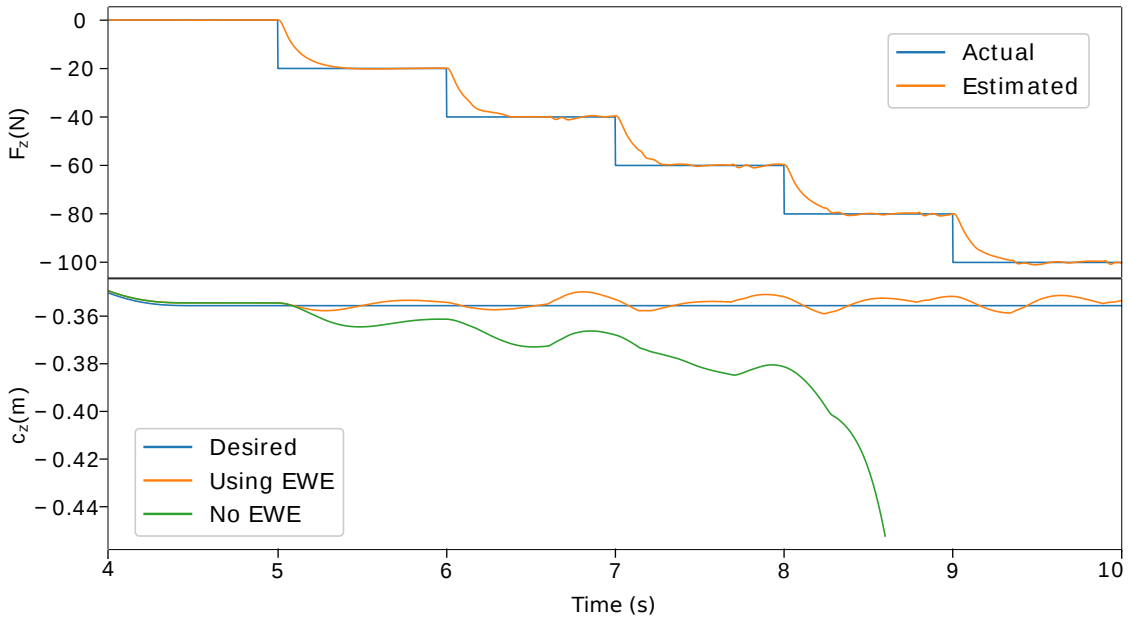


Figure IV.13: Top: External wrench estimation for an increasing simulated load at the robot base in the $-z$ direction during walking (approximately one-fifth the robot's weight). Bottom: COM height tracking, showing the robot going unstable without the EWE.

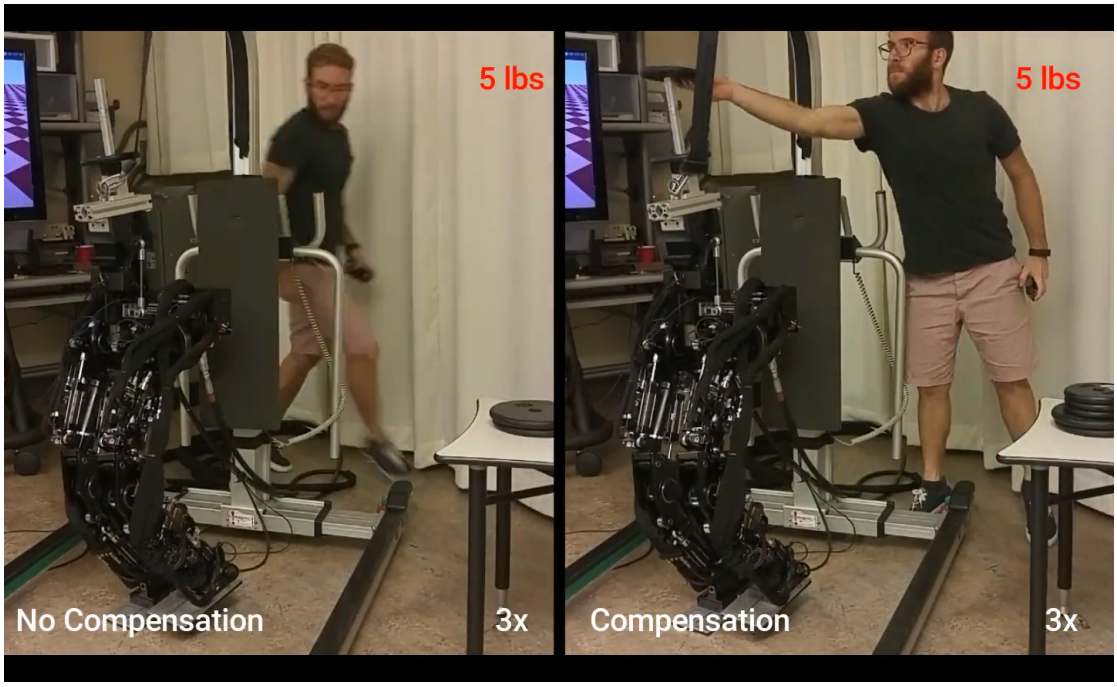


Figure IV.14: Loading of the robot during a single-leg squatting task in five pound (20N) increments, both with and without external wrench estimation/compensation.

the hydraulic hoses and disturbances. We thus seek to evaluate the EWE for use in compensating these modeling errors on the real robot.

For the following experiments, the same estimator noise parameters are used as in simulation (these were originally based on noise statistics from the actual sensors). We perform a single-leg balancing task in which the robot squats, shifts its COM over one foot and lifts its leg after switching contacts based on a measured normal force threshold. The EWE is updated throughout the experiment (across the contact change, demonstrating proper contact handling); the resulting estimated external torque is smoothly ramped to its full value over several seconds starting at the beginning of the task. In order to focus on disturbances created in particular directions, a selection matrix is specified which premultiplies the external wrench sent to the inverse dynamics solver.

First, we performed a similar experiment to that shown in Figure IV.13 in which the robot was incrementally loaded with 5 pound weights (weighing roughly 20N each) up to approximately 100N during a single-leg squatting task as shown in the video snapshot of Figure IV.14. As shown in the top portion of Figure IV.15, the resulting external force in the direction oposite gravity is estimated well, despite the contact switch and ongoing squatting task. The estimated external force in the negative z-direction was compensated in the inverse dynamics solver, preventing the COM from moving downwards more than several millimeters despite the significant additional load.

In contrast, Figure IV.16 shows the COM height during the same loading experiment *without* use of the external wrench estimator for compensation. The COM moves down significantly more and eventually the robot goes unstable after being loaded to 100N, unable to continue the squatting task in the presence of the external force.

Second, we performed an experiment in which the robot was perturbed impulsively with a push rod on which a force/torque sensor was attached. The top plot of Figure IV.17 shows the measured disturbance force while the bottom plot shows the COM x-direction tracking without using external wrench compensation. The COM is seen to move on the order of a centimeter in the same pattern as the push force. In an experiment with similar disturbances, external wrench estimation was used to accurately estimate the push force (up to an offset perhaps due to the hydraulic hose forces) as shown in the top of Figure IV.18. The resulting COM tracking improves as a result, as shown in the bottom portion of the figure. Again, note that these results are dependent on the chosen COM control gains; a more dramatic difference in tracking would be seen were the COM control gains lowered.

IV.6 Summary

Overall, the momentum-based filters perform better than the corresponding LIPM-based filters of (Stephens, 2011) with only the disadvantage of being slightly more complex. The estimator to use depends on the application, though we expect both offsets and external

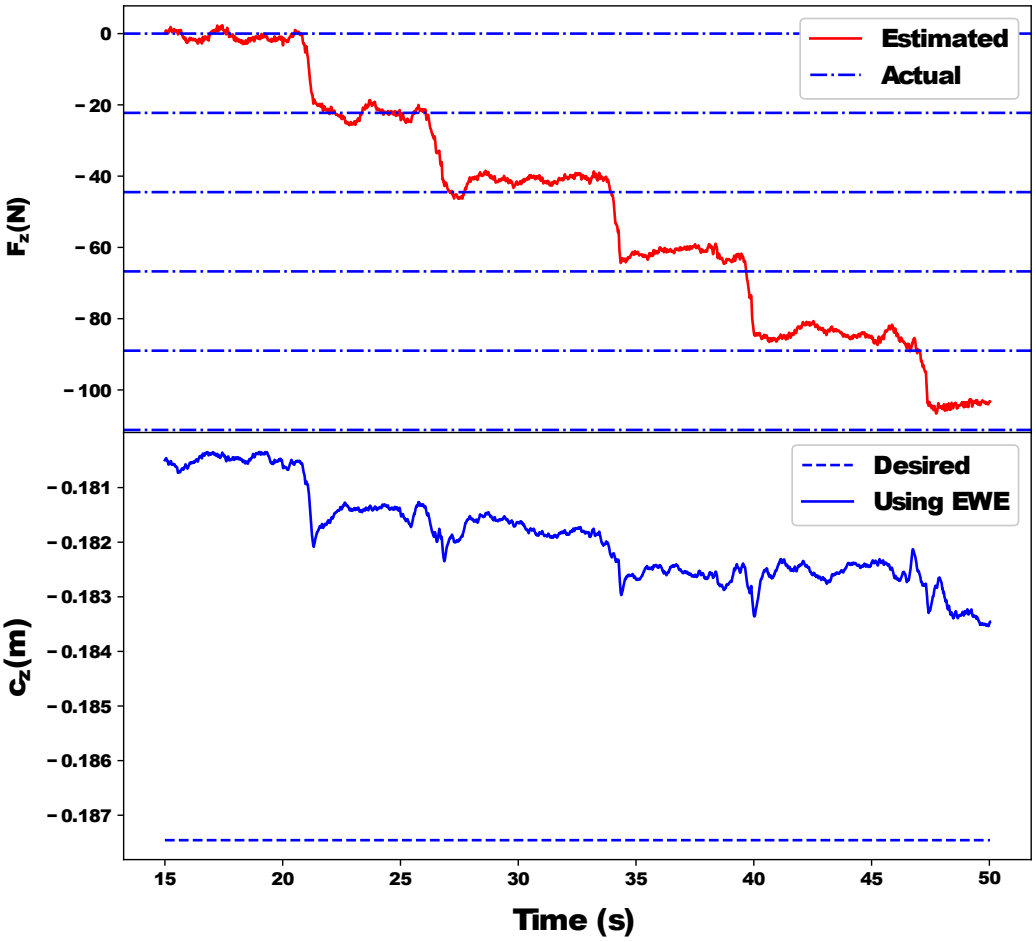


Figure IV.15: Loading of the robot during a single-leg squatting task in five pound (20N) increments while estimating the external force (top) and using it for feedforward compensation to prevent the COM from moving downwards (bottom). Note that the time at which each weight was added is not known precisely, thus the increasing load magnitudes are shown as horizontal lines in the top plot.

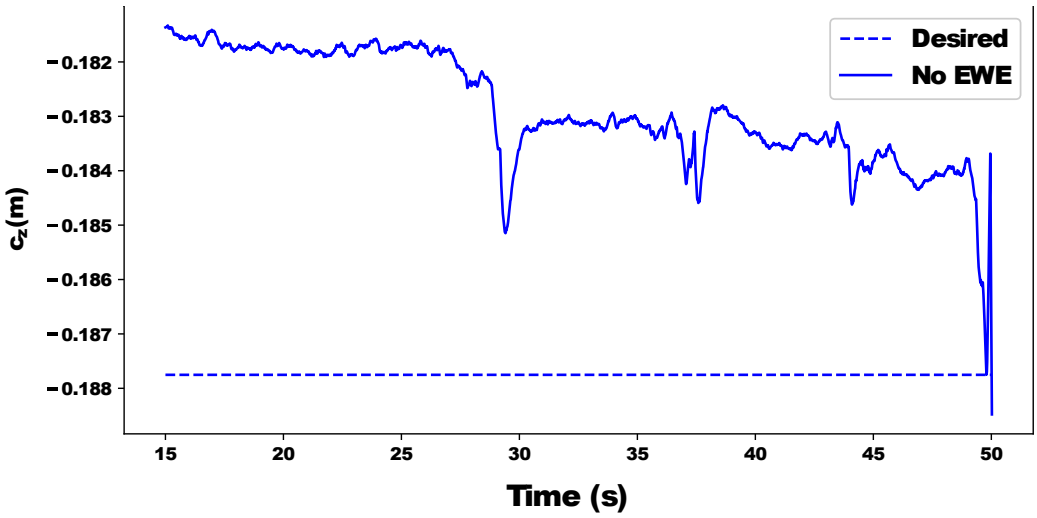


Figure IV.16: External wrench estimation for an increasing simulated load at the robot base in different directions. Load compensation prevents the robot from deviating from its desired COM position.

wrenches to exist on the real robot. Based on the analysis of Section IV.3.5 we expect a combined estimator may work well with proper tuning if the external COM torque is small. Alternatively, we can simplify the problem and treat a COM offset is as an external force as described in Section IV.5, motivating the use of only the external wrench estimator; this was done in order to test the EWE for closed-loop control on the real humanoid.

The results of the experiments performed in the previous section are summarized below.

- The ME and LIPMF both estimate the COM well but the ME performs better at estimating linear momentum.
- LIPMF performance degrades much more rapidly than ME performance for decreasing update rates.
- The ME filters the COM and momentum with much less delay than a typical low-pass filter.

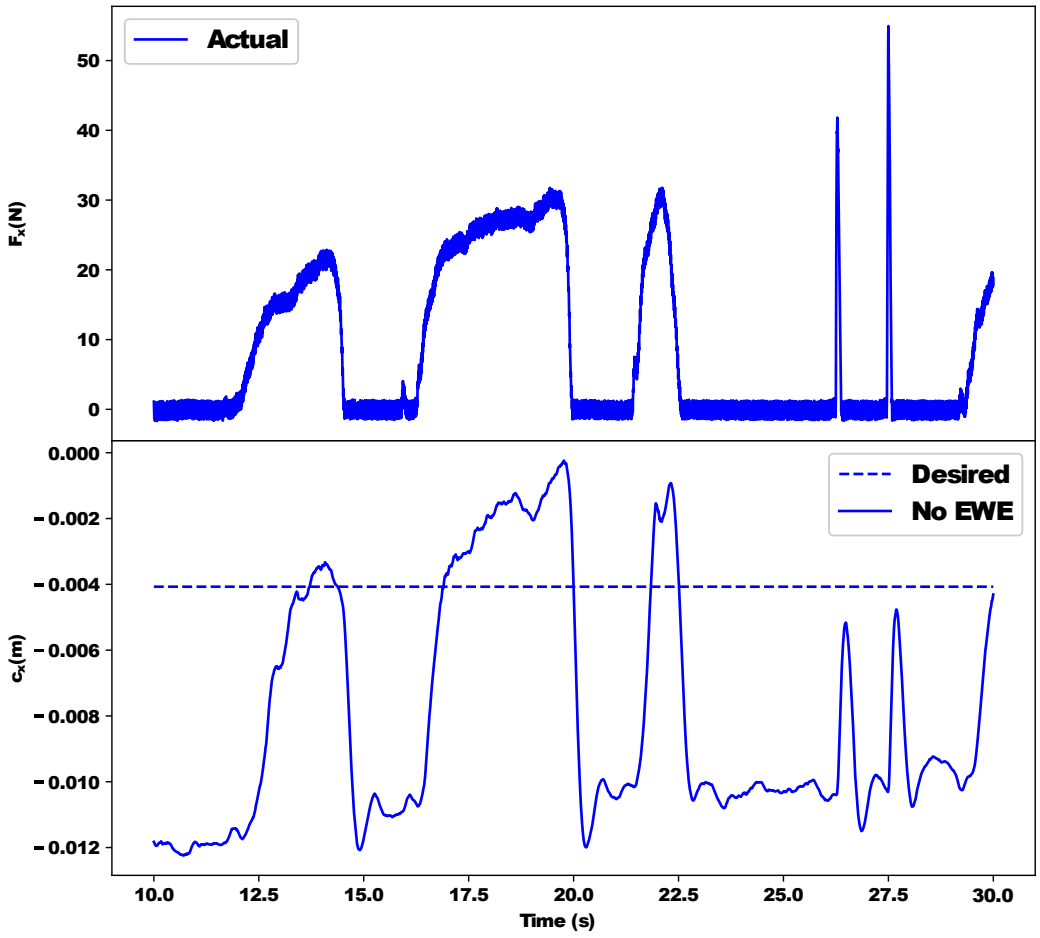


Figure IV.17: Perturbing the robot in the x-direction while measuring the external force (top), resulting in poor COM tracking (bottom).

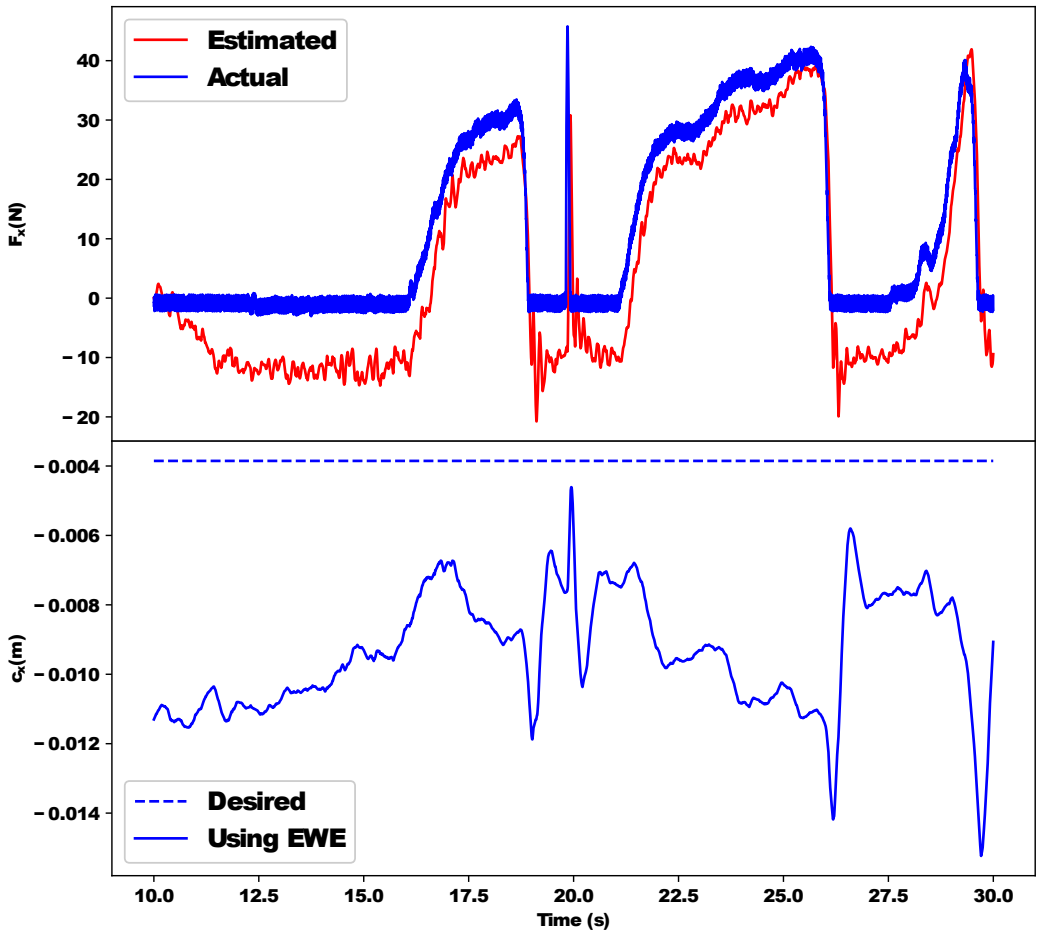


Figure IV.18: Perturbing the robot in the x-direction while both measuring and estimating the external force (top) and using it for feedforward compensation to prevent the COM from moving (bottom).

- Both the OE and LIPMF estimate a constant COM offset well but the OE performs much better for configuration-dependent offsets since it estimates Δl.
- The COE estimates offsets more accurately than the OE in cases of large modeling errors.
- The EWE accurately estimates constant, time-varying and impulsive external wrenches even during walking.
- The EWE can be used in closed-loop, inverse dynamics control in order to effectively compensate for potentially-destabilizing external loads, as shown in both simulated and real-robot experiments.

Chapter V

Joint State Estimation

Feedback control for robots relies on accurate estimates of the joint state. Traditionally, the position of each joint is measured using an angular sensor from which joint velocity and acceleration are computed via numerical differentiation. This produces quantities subject to considerable noise, requiring low-pass filtering for use in control. However, filtering introduces a potentially-destabilizing time delay, preventing the use of feedback gains high enough to achieve satisfactory stiffness and damping; see Figure V.1. Rather than differentiating joint positions, we develop methods for computing joint velocity and acceleration from sensors which measure quantities of the same order. These estimates - derived from gyroscopes and accelerometers - can be used for control directly or through fusion with position sensing in an optimal filtering framework.

While compact and affordable inertial sensors are fairly new, various types of accelerometers have been used in sensor fusion for decades. One of the first works to compute rigid body angular velocity from accelerometers was (Schuler et al., 1967). (Padgaokkar et al., 1975) introduced methods for computing angular acceleration, with (Liu, 1976) demonstrating that nine axes are needed for stable solutions. (Zappa et al., 2001) proved that 12 accelerometer axes are actually required to avoid angular velocity singularities.

Human joint angle estimation has been researched extensively in the biomechanics community. (Williamson and Andrews, 2001) attached IMUs to a subject and integrated gyroscope signals to obtain knee joint angles. (El-Gohary, 2013) developed a human arm model used to derive relations between limb IMU measurements and joint state for use in

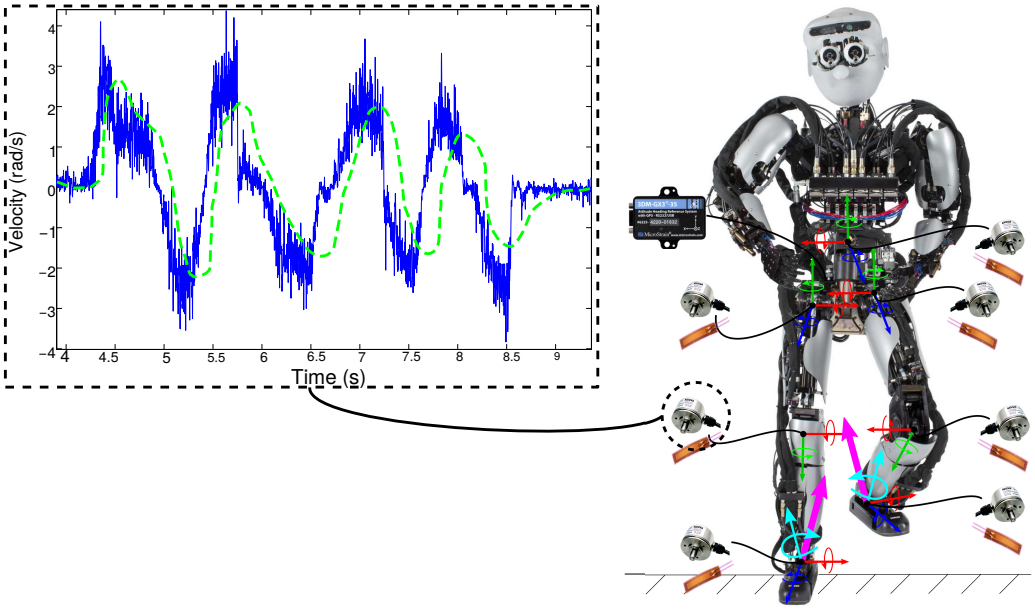


Figure V.1: Illustration of the noisy velocity signal (in blue) computed by differentiating a raw joint potentiometer signal and the corresponding delayed signal (in green) produced by low-pass filtering.

a Kalman Filter. (Seel et al., 2014) determined joint axes and locations from limb IMUs using knowledge of human kinematic constraints.

Research on robot state estimation using multiple IMUs, however, has been limited. For base state estimation, (Lin et al., 2006) developed a 12-axis accelerometer suite for their robot, later adding a three-axis gyroscope to avoid singularities. They also proved a three-axis gyroscope plus six accelerometer axes are sufficient to compute angular acceleration if distributed among three distinct locations (Lin et al., 2012). For joint angle estimation, (Cheng and Oelmann, 2010) surveyed methods for robots lacking angular sensors. (Santaera et al., 2015) sensorized manipulator links and used integrated orientation and kinematics to determine joint angles. (Vihonen et al., 2013) developed an estimator for robots lacking sensors which relied on tilt from accelerometers compensated for inertial effects using gyroscope-based joint velocities, also compensating biases through complementary filtering. In (Vihonen et al., 2013), numerical joint accelerations were replaced

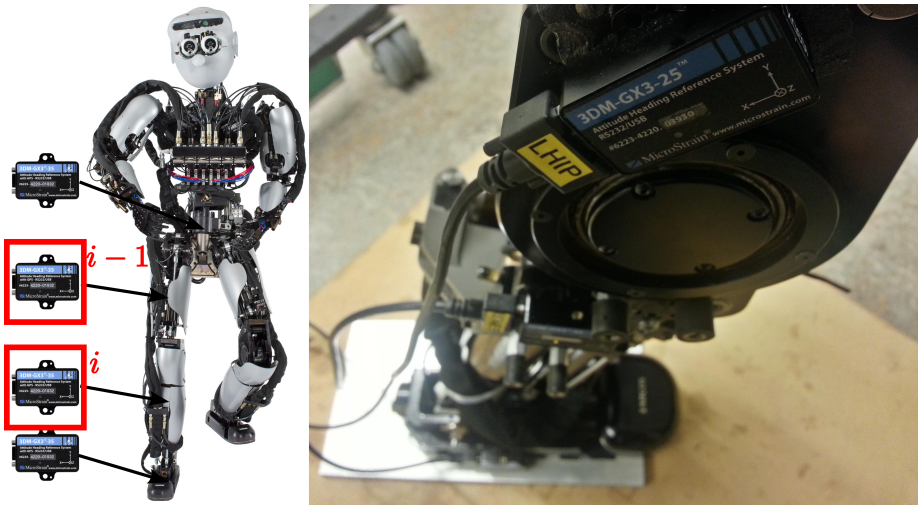


Figure V.2: Inertial Measurement Units (IMUs) attached to the thigh, shank and foot of a Sarcos hydraulic humanoid.

with estimates from accelerometers and then used to estimate velocities in a complementary filter (Vihonen et al., 2014). (Honkakorpi et al., 2013) demonstrated that feedback using IMU-based estimates yields similar performance to using encoder measurements. Recently, (Xinjilefu et al., 2016) fused predicted joint accelerations, angular velocities from link-mounted gyroscopes and joint states measured by encoders. An orientation calibration routine similar to that presented here is detailed, however joint accelerations are not considered (neither is a joint position calibration routine or gyroscope bias estimation). It was shown that using estimated joint velocities, task-level gains (on the COM) could be increased; this is in agreement with the results presented in this chapter, in which we show that joint feedback damping is considerably improved.

Unlike most previous work (with the exception of (Xinjilefu et al., 2016) which was developed at the same time), the theory presented here applies to floating base robots with three degree of freedom joints, does not rely on global link orientations, introduces filters for gyroscope bias compensation and considers the use of accelerometers and their role in joint state estimation. Additionally, we present results on feedback control of individual joints

which demonstrate the ability to increase both stiffness and damping when using gyroscope-based joint velocities. We begin by considering the theory necessary for computing joint velocities and accelerations from IMU sensor measurements.

V.1 Sensor Framework

Assume we have a multibody system composed of N links $L_1, \ldots, L_N$ in series, the first of which is a floating base. These are connected by N joints $J_0, \ldots, J_{N-1}$ where J_0 is the floating base orientation in a minimal set of coordinates. We fix an IMU containing a three-axis gyroscope and a three-axis accelerometer to each link at a known position in the link frame and with the same orientation as the link frame (IMU pose calibration is detailed in Sec. V.2.1 and V.2.2).

Let $\omega_{L_1}^W, \omega_{L_2}^W, \cdots, \omega_{L_N}^W$ denote the angular velocities of each link in the world (global) frame; the gyroscope on L_i thus measures

$$\bar{\omega}_{L_i} = R_W^{L_i}(\omega_{L_i}^W + b_{\omega,L_i} + w_{\omega,L_i}) \tag{V.1}$$

where $R_W^{L_i}$ rotates a quantity in world frame into the local frame of L_i and b_{ω,L_i} and w_{ω,L_i} denote time-varying bias and thermal noise vectors, respectively. Similarly, $a_{L_1}^W, a_{L_2}^W, \cdots, a_{L_N}^W$ denote the true accelerations of the IMU locations on each link in world frame so that the accelerometer on L_i measures

$$\bar{a}_{L_i} = R_W^{L_i}(a_{L_i}^W + b_{a,L_i} + w_{a,L_i} + g) \tag{V.2}$$

where $g = [0, 0, -9.81]$ is the gravity vector and b_{a,L_i} and w_{a,L_i} again denote bias and noise vectors, respectively. For the time being, we will not consider the effects of noise sources; these will be addressed in Sec. V.3.1.

V.1.1 Joint Velocity Computation from Gyroscopes

Assuming in the most general case that every joint has three Degrees of Freedom (DoFs), the vector $\omega^i_{i-1,i}$ measuring the angular velocity of L_i relative to that of L_{i-1} in the L_i frame corresponds to the velocity $\dot{\theta}_{i-1} \in R^3$ of joint J_{i-1}. From this point on, we will drop the use of L and J in subscripts for the sake of brevity. For the floating base (L_1) gyroscope we thus have

$$\bar{\omega}_1 = R^1_W \omega^W_1 = \dot{\theta}_0$$

where $\dot{\theta}_0$ is the base angular velocity in the base link frame. For L_2 we have

$$\bar{\omega}_2 = R^2_W \omega^W_2 = R^2_W(\omega^W_1 + \omega^W_{1,2})$$

where we have used the velocity composition rule. This simplifies further to

$$\bar{\omega}_2 = R^2_1 R^1_W \omega^W_1 + R^2_W \omega^W_{1,2} = R^2_1 \dot{\theta}_0 + \dot{\theta}_1$$

where R^2_1 represents a rotation from frame L_1 to L_2. Solving for the velocity of J_1, we have

$$\dot{\theta}_1 = \bar{\omega}_2 - R^2_1 \dot{\theta}_0$$

Similarly, for L_3 we have (again using velocity and rotation composition rules)

$$\begin{aligned}
\bar{\omega}_3 &= R^3_W \omega^W_3 \\
&= R^3_W(\omega^W_1 + \omega^W_{1,2} + \omega^W_{2,3}) \\
&= R^3_2 R^2_1 R^1_W \omega^W_1 + R^3_2 R^2_W \omega^W_{1,2} + R^3_W \omega^W_{2,3} \\
&= R^3_2 R^2_1 \dot{\theta}_0 + R^3_2 \dot{\theta}_1 + \dot{\theta}_2
\end{aligned}$$

and solving for the velocity of J_2 yields

$$\dot{\theta}_2 = \bar{\omega}_3 - R_2^3 R_1^2 \dot{\theta}_0 - R_2^3 \dot{\theta}_1$$

Continuing in this manner, we form the linear system

$$\begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ R_1^2 & I & 0 & \ddots & \vdots \\ R_1^3 & R_2^3 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & I & 0 \\ R_1^N & R_2^N & \cdots & R_{N-1}^N & I \end{bmatrix} \begin{bmatrix} \dot{\theta}_0 \\ \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_{N-2} \\ \dot{\theta}_{N-1} \end{bmatrix} = \begin{bmatrix} \bar{\omega}_1 \\ \bar{\omega}_2 \\ \vdots \\ \bar{\omega}_{N-1} \\ \bar{\omega}_N \end{bmatrix} \tag{V.3}$$

which we can write more compactly as $T\dot{\theta} = \bar{\omega}$ where $\dot{\theta} \in R^{3N}$ and $\bar{\omega} \in R^{3N}$ are the full vectors of joint velocities and link gyroscope measurements, respectively. Note that T is a function of θ and is composed of relative rotations between links; unlike in other methods, there is no need for global link poses. Additionally, since T is lower-triangular this system always has a unique solution which can be determined efficiently using forward substitution.

Computing Constrained Velocities

The above assumes that every joint has three DoFs; however, it is often the case in for a robot that certain joints have fewer (for example, the knee of a humanoid). In this case, more accurate solutions may result from properly constraining velocities using knowledge of the kinematic structure.

For the L_i gyroscope we again have (neglecting noise)

$$\bar{\omega}_i = R_W^i \omega_i^W = R_W^i J_i^W \dot{q} = J_i^1 \dot{q}$$

where $\dot{q} = [\omega_{base}, \dot{q}_{joints}]$ is the vector of generalized joint velocities and J_i^1 is the Jacobian relating the angular velocity of L_i to generalized joint velocities in the L_i frame. Using this relation for each link, we form the system

$$\begin{bmatrix} J_1^1 \\ J_2^2 \\ \vdots \\ J_N^N \end{bmatrix} \dot{q} = \begin{bmatrix} \bar{\omega}_1 \\ \bar{\omega}_2 \\ \vdots \\ \bar{\omega}_N \end{bmatrix} \tag{V.4}$$

where each Jacobian above is computed using Jacobians relative to the base as $J_i^i = R_1^i J_i^1$. We can write this system as $T_J \dot{q} = \bar{w}$ and solve a least-squares problem for joint velocities. While forming the matrix T_J requires knowledge of the kinematic structure, this method results in velocities which are properly constrained and expressed in the correct frames. We will demonstrate the effect of properly constraining velocities in Sec. V.4.

V.1.2 Joint Acceleration Computation from Accelerometers

We now use the setup detailed * in Sec. V.1 to determine the acceleration $\ddot{\theta}_{i-1}$ of J_{i-1}. The IMU accelerometer on the preceding link measures

$$\bar{a}_{i-1} = R_W^{i-1}(a_{i-1}^W + g)$$

and the accelerometer on the link immediately following J_{i-1} measures

$$\bar{a}_i = R_W^i(a_i^W + g) = R_W^i(a_{i-1}^W + a_{i-1,i}^W + g)$$

where $a_{i-1,i}^W$ is the acceleration of the L_i IMU with respect to the L_{i-1} IMU and is given by

$$a_{i-1,i}^W = (\omega_{i-1,i}^W \times (\omega_{i-1,i}^W \times r_{i-1,i}^W)) + (\alpha_{i-1,i}^W \times r_{i-1,i}^W)$$

*Note that we could again constrain the acceleration using the Jacobian since $\alpha = J\ddot{\theta} + \dot{J}\theta$ but we wish to avoid using the Jacobian derivative, which must be computed numerically and is thus typically very noisy.

where $r^W_{i-1,i}$, $\omega^W_{i-1,i}$ and $\alpha^W_{i-1,i}$ are the position, angular velocity and angular acceleration of the L_i IMU with respect to the L_{i-1} IMU in world frame. Using the notation $a \times b = a^\times b$ where $a^\times \in R^{3\times 3}$, it follows that

$$\begin{aligned}
\bar{a}_i &= R^i_{i-1}\left[R^{i-1}_W(a^W_{i-1}+g)\right] + R^i_W a^W_{i-1,i} \\
&= R^i_{i-1}\bar{a}_{i-1} + R^i_W\left[((\omega^W_{i-1,i})^\times)^2 r^W_{i-1,i} + (\alpha^W_{i-1,i})^\times r^W_{i-1,i}\right] \\
&= R^i_{i-1}\bar{a}_{i-1} + ((\omega^i_{i-1,i})^\times)^2 r^i_{i-1,i} - (r^i_{i-1,i})^\times \alpha^i_{i-1,i} \\
&= R^i_{i-1}\bar{a}_{i-1} + ((\dot{\theta}_{i-1})^\times)^2 r^i_{i-1,i} - (r^i_{i-1,i})^\times \ddot{\theta}_{i-1}
\end{aligned}$$

where we have used the definition of $\bar{a}_{i-1}$, properties of the cross product and the definitions $\dot{\theta}_{i-1} = \omega^i_{i-1,i}$ and $\ddot{\theta}_{i-1} = \alpha^i_{i-1,i}$. We rearrange the above equation to get

$$(r^i_{i-1,i})^\times \ddot{\theta}_{i-1} = R^i_{i-1}\bar{a}_{i-1} - \bar{a}_i + ((\dot{\theta}_{i-1})^\times)^2 \quad \forall i = 2, \ldots, N$$

Note that gravity does not appear in the above, making gravity compensation unnecessary. Also, the joint velocities $\dot{\theta}_i$ are determined from the link gyroscopes as in Sec. V.1.1. However, we cannot solve this equation for $\ddot{\theta}_{i-1}$ because the skew-symmetric matrix $(r^i_{i-1,i})^\times \in R^{3\times 3}$ has rank two.[†] We thus add a second IMU to L_i and solve

$$\begin{bmatrix} (r^i_{i-1,i})^\times \\ (\tilde{r}^i_{i-1,i})^\times \end{bmatrix} \ddot{\theta}_{i-1} = \begin{bmatrix} R^i_{i-1}\bar{a}_{i-1} - \bar{a}_i + ((\dot{\theta}_{i-1})^\times)^2 \\ R^i_{i-1}\bar{a}_{i-1} - \tilde{\bar{a}}_i + ((\dot{\theta}_{i-1})^\times)^2 \end{bmatrix} \tag{V.5}$$

where $\tilde{r}^i_{i-1,i}$ and $\tilde{\bar{a}}_i$ denote the relative position and the acceleration of the additional IMU. The matrix multiplying $\ddot{\theta}_{i-1}$ now has rank three aside from singular cases (for example when $\tilde{r}^i_{i-1,i} = r^i_{i-1,i}$, implying that the IMU positions should be as distinct as possible). Note that rather than use two three-axis accelerometers, we could have equivalently used three two-axis accelerometers to obtain a full-rank matrix; these results are in agreement with Lin et al. (2012).

[†]Real skew-symmetric matrices have purely imaginary eigenvalues, which must come in conjugate pairs; thus, the rank must be even.

V.2 IMU Pose Calibration

To obtain accurate estimates of joint velocities and accelerations, the IMUs must be fixed with known poses. As is often the case for humanoids, the base IMU is rigidly fixed with a known pose; we compute orientation and position offsets for each link relative to this IMU using the following principle. When rotated in the air with locked joints, the entire robot becomes a single rigid body subject to the same angular velocity and angular acceleration. This is the basis for the following offline calibration methods which recover the full pose (orientation and position) of each IMU with respect to its local link frame.

V.2.1 Orientation Calibration

When every link has the same angular velocity, we can equate the velocities of L_i and that of the base to obtain

$$\hat{R}_i \bar{\omega}_i = R_1^i \bar{\omega}_1$$

where $\hat{R}_i$ is the desired rotational correction for the IMU on L_i. Transposing both sides and stacking M consecutive measurements, we obtain

$$\begin{bmatrix} \{\bar{\omega}_i^T\}_1 \\ \{\bar{\omega}_i^T\}_2 \\ \vdots \\ \{\bar{\omega}_i^T\}_M \end{bmatrix} \hat{R} = \begin{bmatrix} \{\bar{\omega}_1^T (R_1^i)^T\}_1 \\ \{\bar{\omega}_1^T (R_1^i)^T\}_2 \\ \vdots \\ \{\bar{\omega}_1^T (R_1^i)^T\}_M \end{bmatrix}$$

where $\{v\}_m$ denotes the m^{th} observation of quantity v. We seek $\hat{R}_i$ as the solution to the problem

$$\hat{R} = \arg\min_X ||AX - B||_F^2$$

subject to $X^T X = I$ where $||A||_F$ denotes the Frobenius matrix norm. This is called the orthogonal Procrustes problem and is solved by computing the SVD of $A^T B = U \Sigma V^T$ and setting $\hat{R} = UV^T$. In order to ensure that the solution is a proper rotation matrix we

also require $\det(\hat{R}) = +1$. This is known as the Kabsch algorithm Kabsch (1976) and is achieved by instead setting $\hat{R} = U\hat{\Sigma}V^T$ where $\hat{\Sigma} = \text{diag}(1, 1, \text{sign}(\det(UV^T)))$.

V.2.2 Position Calibration

Assuming every link has the same world frame angular acceleration α^W, the linear acceleration of the IMU on L_i relative to that of the base link IMU is

$$a_i^W - a_1^W = \omega^W \times \omega^W \times r_{1,i}^W + \alpha^W \times r_{1,i}^W$$

However, we know that $\bar{a}_i = R_W^i(a_i^W + g)$ and thus $a_i^W = R_i^W \bar{a}_i - g$ so

$$a_i^W - a_1^W = R_i^W \bar{a}_i - g - R_1^W \bar{a}_1 + g = R_1^W(R_i^1 \bar{a}_i - \bar{a}_1)$$

Note that the gravity vector again cancels, making compensation unnecessary. Multiplying both sides by R_W^1, we have

$$R_i^1 \bar{a}_i - \bar{a}_1 = \omega^1 \times \omega^1 \times r_{1,i}^1 + \alpha^1 \times r_{1,i}^1$$

which we can rewrite in the form

$$\left[(\bar{\omega}^\times)^2 + \bar{\alpha}^\times\right] r_{1,i}^1 = \left(R_i^1 \bar{a}_i - \bar{a}_1\right)$$

where $\bar{\omega}$ and $\bar{\alpha}$ denote the angular velocity and acceleration measured by the base link IMU. Again, stacking M consecutive measurements results in the linear system

$$\begin{bmatrix} \{(\bar{\omega}^\times)^2 + \bar{\alpha}^\times\}_1 \\ \{(\bar{\omega}^\times)^2 + \bar{\alpha}^\times\}_2 \\ \vdots \\ \{(\bar{\omega}^\times)^2 + \bar{\alpha}^\times\}_M \end{bmatrix} r_{1,i}^1 = \begin{bmatrix} \{R_i^1 \bar{a}_i - \bar{a}_1\}_1 \\ \{R_i^1 \bar{a}_i - \bar{a}_1\}_2 \\ \vdots \\ \{R_i^1 \bar{a}_i - \bar{a}_1\}_M \end{bmatrix}$$

Note that we compute $\bar{\alpha}$ numerically from base gyroscope measurements. Since this is an offline calibration routine, we apply a zero-delay filter to the measurements. Since the matrix multiplying IMU position is the same for every link, we compute its SVD once and find the least-squares solution for each IMU. After solving for the position $r^1_{1,i}$ of the i^{th} IMU in the base frame, we compute its local link position from kinematics.

V.3 Joint State Estimators

In this section, we introduce two Kalman Filters for the joint state which fuse joint velocities and accelerations computed from inertial sensors with joint position sensor measurements. This is advantageous over directly using the computed joint state because it ensures consistency between the joint position and its derivatives. In theory, performing filtering using an accurate process model also creates less delay than simply low-pass filtering computed quantities.

V.3.1 Joint Position and Gyroscope Bias Filter

Assume now that each gyroscope is afflicted by a time-varying bias and thermal noise as in the model given by Eq. (V.1). From Eq. (V.4) we thus have

$$T_J(\theta)\dot{\theta} = \bar{\omega} - b - w$$

where $b, w \in R^{3N}$ are the bias and noise vectors for all gyros, respectively. Combining the joint positions and gyroscope biases into the state vector $x = [\theta^T, b^T]^T \in 6N$, we can write the state dynamics as

$$\dot{x} = f(x, \bar{\omega}, w, w_b)$$

or more specifically as

$$\begin{aligned}\dot{\theta} &= -T_J(\theta)^{-1}b + T_J(\theta)^{-1}(\bar{\omega} - w)\\ \dot{b} &= w_b\end{aligned}$$

where we have modeled the bias dynamics as Brownian motion (the integral of white noise process w_b). Since we have angular position sensors, we can write a measurement of the state as

$$y = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} \theta \\ b \end{bmatrix} + v$$

where $v \in R^{3N}$ is the measurement noise afflicting the joint position sensor measurements. Since the process model is nonlinear, we choose an EKF for implementation.

V.3.2 Acceleration-Based Joint Velocity Filter

In order to filter joint velocities, we need joint accelerations $\ddot{\theta}$ for use in the process model. Defining the state vector $x = [\theta^T, \dot{\theta}^T]^T = [x_1^T, x_2^T]^T \in 6N$ we have the trivial dynamics

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = \ddot{\theta}$$

The joint acceleration vector $\ddot{\theta}$ can be specified in any the following ways, depending on sensor availability:

$$\ddot{\theta} = \begin{cases} M(\theta)^{-1}[\bar{\tau} + J(\theta)_c^T \bar{F} - n(\theta, \dot{\theta})] \\ f(\theta, \dot{\theta}, \bar{a}) \\ \ddot{\theta}_{des} \end{cases}$$

The first method uses measured joint torques $\bar{\tau}$ and endeffector wrenches $\bar{F}$ along with the robot's dynamic model to compute the current joint accelerations. The second method solves for accelerations using link accelerometers with Eq. (V.5). Alternatively, we can simply use the desired or predicted joint accelerations $\ddot{\theta}_{des}$ from the current controller. Note that use of the first two methods results in a nonlinear process model; in this work, we consider only using desired accelerations for simplicity.

We measure the joint positions from angular sensors and the joint velocities from the link IMUs as in Sec. V.1.1, leading to the measurement model $y = x + v$ where $v \in R^{6N}$ is again the measurement noise vector.

V.4 Experiments and Results

The platform used for experiments is the lower body of a Sarcos hydraulic humanoid robot having a total of 14 DoFs (seven per leg). An IMU containing a three-axis gyroscope and three-axis accelerometer was mounted to each link - one on the base and one on the link immediately following the hip, knee and ankle joints as in Figure V.2. We chose Microstrain 3DM-GX3-25 IMUs for their USB interface, low-noise sensors and maximum streaming rate of 1 kHz. Initial gyroscope biases were removed at start-up. The IMUs were fixed using an adhesive; while an effort was made to align the IMU axes with the link frames, using the calibration methods of Sec. V.2.1 and V.2.2 allowed for imprecise sensor placement.

We first evaluated the joint velocity computation detailed in Sec. V.1.1 during a sine tracking task for every joint in the right leg. Joint velocities were computed from measured gyroscope data in three ways: A) using Eq. (V.3) to compute orientation-corrected but unconstrained velocities, B) using Eq. (V.4) to compute constrained but non-calibrated velocities and C) again using Eq. (V.4) to compute velocities both orientation-corrected and constrained to the kinematics. These were compared against velocities computed from potentiometer measurements and filtered using a second-order Butterworth filter with a cutoff of $25Hz$. Fig. V.3 shows the effect of constraining computed joint velocities; the constrained hip velocities more closely match the filtered potentiometer-based velocities. Fig. V.4 shows the effect of performing the offline rotation calibration routine on constrained ankle velocities; the benefit of calibration is apparent.

We next evaluate the IMU position calibration and accelerometer-based joint acceleration computations detailed in Sec. V.1.2. Fig. V.5 compares the potentiometer-based hip accelerations (heavily filtered at a cutoff of $5Hz$) with those computed from IMUs using

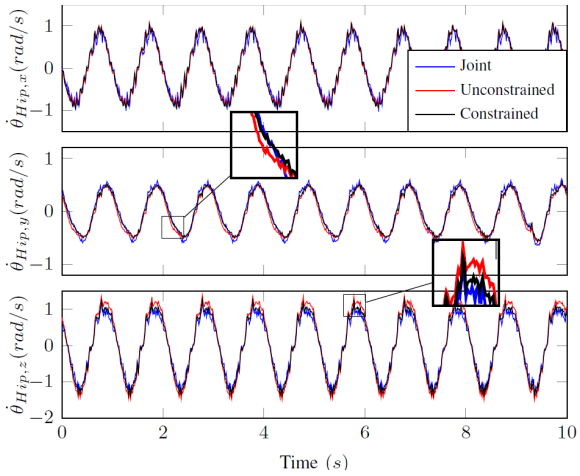


Figure V.3: Comparison between filtered potentiometer-based hip velocities and constrained versus unconstrained hip velocities computed from link gyroscopes.

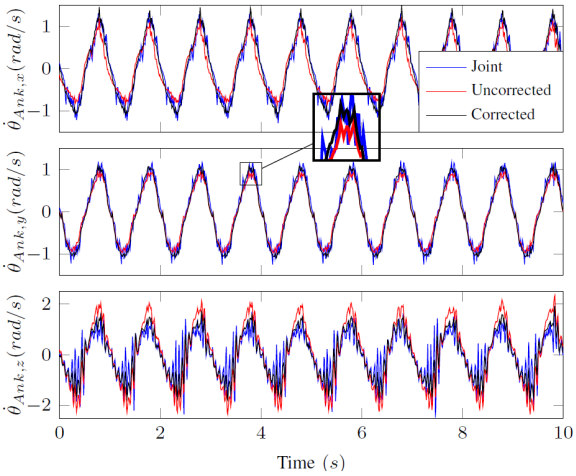


Figure V.4: Comparison between filtered potentiometer-based ankle velocities and rotation-corrected versus uncorrected ankle velocities computed from link gyroscopes.

Eq. (V.5) during a sine task for the hip. We compare the results of computing accelerations using manually-measured IMU positions and automatically-generated positions from the calibration routine of Sec. V.2.2. Both signals estimate the acceleration well and with much less noise than unfiltered, numerically-computed joint accelerations (shown for reference in Fig. V.6). The filtered joint-based accelerations are clearly heavily delayed; this is most evident in the z direction. Because we only have one IMU per link, we compute hip joint accelerations using the base, thigh and shank IMUs with the knee locked. This is not ideal since these IMUs are not truly on the same link; we expect to compute accelerations which are more accurate and less noisy by adding a second IMU to the proper link.

It should be noted that obtaining a good position calibration requires sufficient angular motion of the robot, which can be difficult depending on the setup. Additionally, the position calibration and angular acceleration computations rely on accurate kinematics and a good IMU orientation calibration, else the gravity terms will not cancel; these appear in

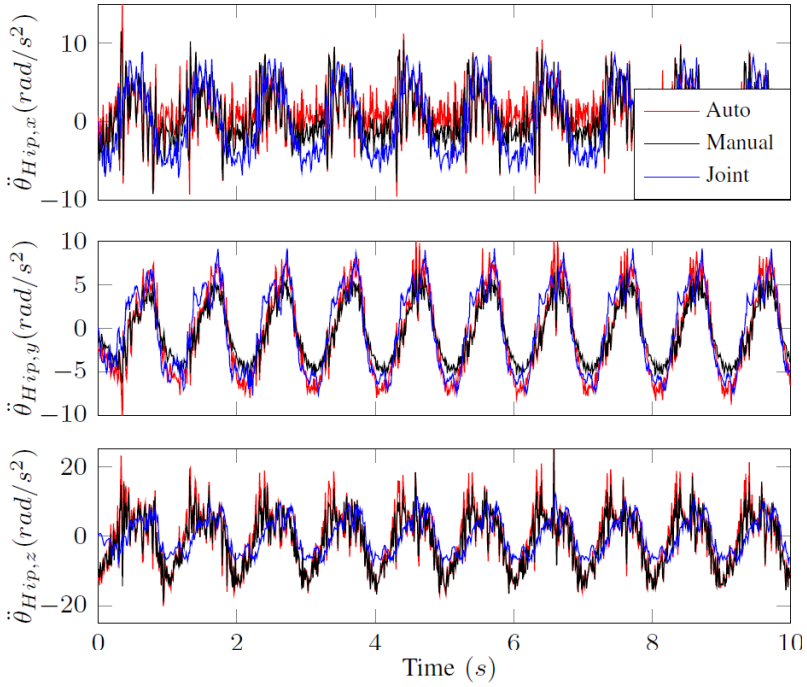


Figure V.5: Filtered potentiometer-based hip joint accelerations versus those computed from inertial sensors with both manually and automatically-generated IMU position information.

Eq. (V.5) to create configuration-dependent acceleration biases. Unmodeled accelerometer biases will have the same effect and should be compensated through calibration, however were neglected here.

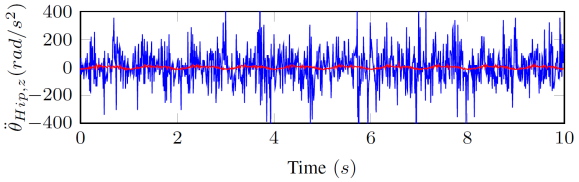


Figure V.6: Raw potentiometer-based hip joint acceleration (blue) versus that computed from inertial sensors (red).

In order to test the gyroscope bias estimator presented in Sec. V.3.1, we performed experiments in the SL simulation environment by simulating gyroscopes subject to the

noise sources in Eq. (V.1). This was done because the IMUs on our robot have a low bias instability; however, this is not the case for inexpensive IMUs which are cost-effective to use in sensorizing every link. We simulate aggressive gyroscope biases which are initially nonzero and drift orders of magnitude faster than those in our IMUs. Fig. V.7 shows the results for one of the simulated IMUs. Despite being run during a full-body sine tracking task with simulated joint sensor noise, the filter manages to track the true biases.

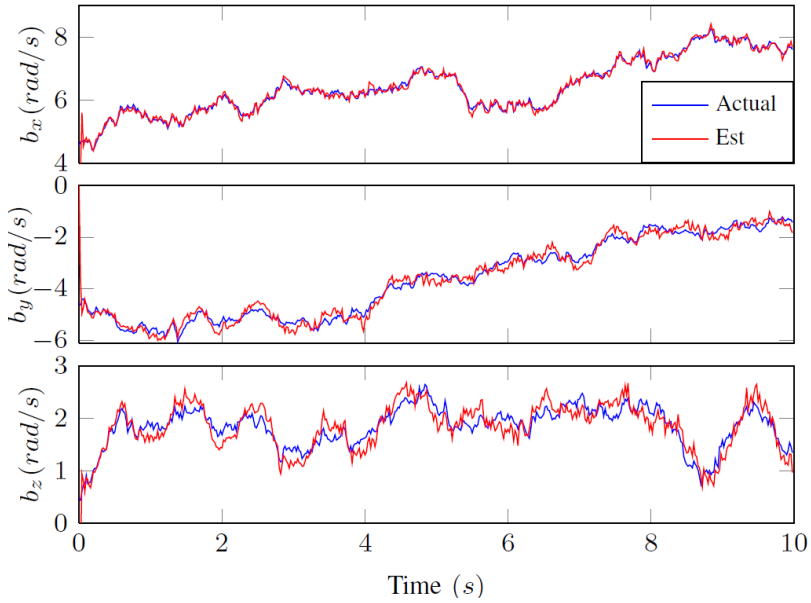


Figure V.7: Simulated gyroscope bias estimation using the joint state filter of Sec. V.3.1.

We also test the joint velocity estimator of Sec. V.3.2 which was implemented for all 14 joint DoFs. Fig. V.8 compares the joint velocity of one hip DoF for the filtered potentiometer-based velocity, the constrained IMU-based velocity from Eq. (V.4), and velocities from the estimator of Sec. V.3.2 both without and with desired joint accelerations as process model inputs. Both filtered velocities are smoother than the IMU-based velocity, however the estimate from the filter which uses desired acceleration in its process model (in black) has tens of milliseconds less delay than the estimate from the filter having a naive process model (in green below). The desired acceleration-based estimator provides a filtered signal with only a slight delay compared to the IMU-computed velocity. Given the

apparent difference between the joint accelerations shown in Fig. V.5 and the sinusoids used to generate them, we expect velocity filter performance to improve considerably by using either sensor-based joint accelerations or accelerations computed using the dynamic model. We leave this investigation to future work.

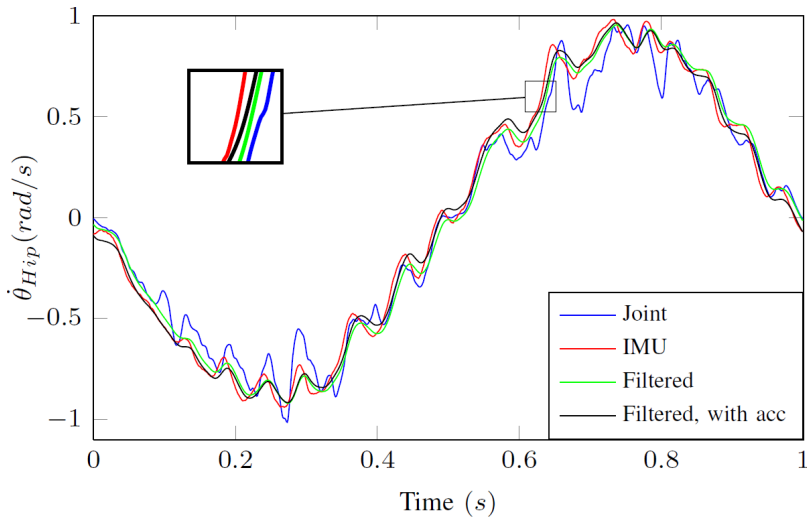


Figure V.8: Hip joint velocity computed from filtered joint sensors, directly from gyroscope velocities, filtered using the estimator of Sec. V.3.2 without and with desired accelerations.

Finally, we perform sine tracking tasks for a single joint in isolation (here the right knee) in order to determine whether using the IMU-based joint velocities allows us to increase controller gains and enable better tracking. A joint PD controller was implemented for the knee using potentiometer-based joint position and velocity (filtered at a $25Hz$ cutoff) with the ability to switch to using the raw joint position and gyroscope-based velocity.

For a $0.5Hz$ sine wave of amplitude $0.25rad$, we were able to increase the position gain to 1000 before the controller using the potentiometer-based velocity went unstable while we could increase the gain to 1600 before the controller using the velocity computed from Eq. (V.4) showed signs of instability. Fig. V.9 shows the tracking using a position gain of 1000 and a velocity gain of 12 for both potentiometer and gyroscope-based joint velocities. RMS tracking error decreases from $0.0103rad$ to $0.0099rad$ in position and from $0.3786rad/s$ to $0.0902rad/s$ in velocity by switching to gyroscope-based velocities. Fig. V.10 shows the

tracking using a position gain of 1500 and a velocity gain of 12 for the gyroscope-based velocities, demonstrating stable position tracking with an RMS error of $0.0077rad$.

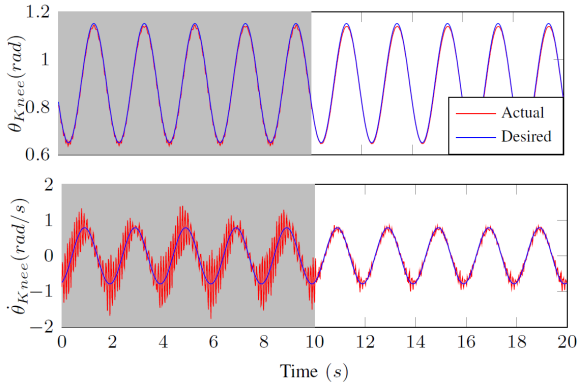


Figure V.9: Knee sine tracking for the gains $P = 1000$ and $D = 12$, switched from potentiometer-based velocities to IMU-based velocities at $t = 10s$.

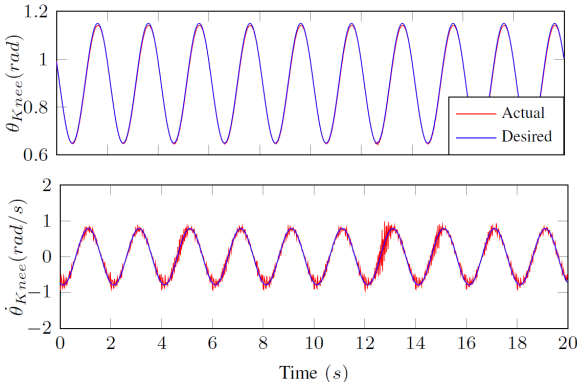


Figure V.10: Knee sine tracking for the gains $P = 1500$ and $D = 12$ for the IMU-based velocities only.

We were also able to independently increase the knee velocity gain from a maximum stable value of 26 using the potentiometer-based velocities to 30 using the gyroscope-based velocities. Fig. V.11 shows the tracking for $D = 26$ using each of these velocities. Position tracking is poor using both since the position gain was held at 250, however it is clear from the figure that damping is improved after the switch.

We finally demonstrate that the above results hold for faster sine tracking by performing a $3Hz$ tracking task using position gains close to the stable limit for each of the potentiometer and gyroscope-based knee velocities. It is evident from Fig. V.12 that both position and velocity tracking are improved after switching to the IMU-based velocities due to the ability to use higher gains.

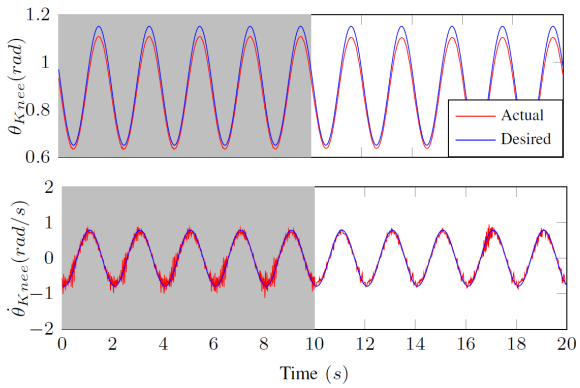


Figure V.11: Knee sine tracking for $P = 250$ and $D = 26$, switched to IMU-based velocities at $t = 10s$.

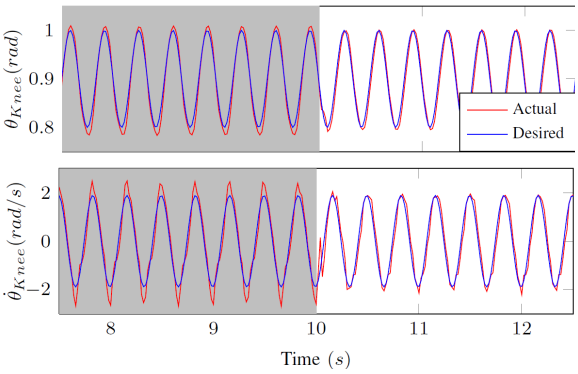


Figure V.12: Knee tracking a 3Hz sine, switched from potentiometer-based velocities with $P = 800$ and $D = 12$ to IMU-based velocities with $P = 1500$ at $t = 10s$.

V.5 Summary

In this chapter, we have:

- Presented methods for computing joint velocities and accelerations directly from inertial sensor measurements.
- Introduced offline calibration procedures which allow for recovery of the pose of the IMUs attached to each link of the robot.
- Proposed filters which fuse measured joint positions with IMU sensor data, allowing for estimation of gyroscopes biases and improvements in the quality of the joint position and velocity estimation.
- Conducted robot experiments using velocity and acceleration estimates, allowing to significantly increase the stiffness and damping of our joint feedback controllers

Now that we have characterized the performance of joint estimation-based feedback control, future work will entail use of these estimators in whole-body control approaches such as the one developed in Section II.3 and in an LQR control framework (Mason et al., 2016) in order to achieve higher damping of both joint and Cartesian quantities (such as momentum, potentially in combination with the approach of Chapter IV). Additionally, we would like to address the hardware issue of potentiometer offset calibration by extending previous work on the topic (Yamane, 2011) to utilize the link IMU sensors.

Chapter VI

Contact State Estimation

VI.1 Introduction

Control and estimation approaches for legged robots rely on assumptions about the contact state of the feet. Floating-base inverse dynamics resolves underactuation by projecting the dynamics into the contact constraints, forcing the endeffector acceleration to be zero Mistry et al. (2010). Locomotion on rough terrain focuses on stabilization through footstep adaptation but ignores the difficulties presented by contact constraint violations Barasuol et al. (2013). Similarly, legged odometry for base state estimation assumes the pose of an endeffector in contact is constant Bloesch et al. (2012). Methods have been introduced to robustify kinematics-based base state estimation, including computing a contact point with minimal instantaneous velocity Masuya and Sugihara (2013) and outlier detection to discard measurements during slip Bloesch et al. (2013), however few consider relaxing the contact assumptions by estimating contact quality in parallel.

Contact estimation is a broad topic which has been investigated in various contexts. Petrovskaya et al.Petrovskaya et al. (2007) were among the first to consider multi-contact force control scenarios in which a manipulator interacts with the environment at points other than its endeffector. Del Prete et al. investigated the effect of contact point estimation error on force control for a humanoid with a capacitive skin Prete et al. (2012). Similar work was done by Manuelli et al. Manuelli and Tedrake (2016) to estimate contact points without a tactile skin by fusing proprioceptive sensing with the dynamic model.

While estimation of contact points has been thoroughly studied, the problem of determining the *quality* of contacts is less well-defined. One aspect of contact quality is the determination of contact constraint directions for a given task. Ortenzi et al. Ortenzi et al. (2016) computed endeffector constraints for a manipulator in contact with a surface using only kinematics, and Nozawa et al. Nozawa et al. (2017) recently presented a similar method for estimating environment constraints in humanoid control tasks.

For humanoids, the quality of a contact is largely determined by friction. Hoepflinger et al. Hoepflinger et al. (2013) investigated foothold quality using unsupervised learning. Terrain elevation map samples were clustered to find a set of primitives which were evaluated for foothold robustness by computing the friction coefficient through exploratory force control. This allowed for prediction of contact quality from visual features for planning. While Focchi et al. Focchi et al. (2017) employed offline friction estimation through for a quadruped walking on steep slopes, Ridgewell et al. Ridgewell (2017) introduced methods for online friction estimation and control adaptation. While the friction coefficient determines linear slip, the center of pressure (CoP) boundaries determine rotational slip/roll. Most controllers assume that the support polygon is the same shape as the foot, however this is invalid on rough terrain where line and even point contacts are encountered. Wiedebach et al. Wiedebach et al. (2016) presented one of the only approaches for online CoP boundary estimation during terrain exploration.

In contrast to approaches which indirectly compute contact quality by computing friction and CoP bounds, we wish to avoid contact models by directly estimating the probability of an endeffector being constrained in each of its six DoFs independently. In this direction, Hwangbo et al. Hwangbo et al. (2016) developed a Hidden Markov Model which uses kinematic and dynamic models to predict contact transitions without force sensing. This one-dimensional approach requires little sensing, however it does not estimate contact quality of the contact nor does it evaluate the classifier in an estimator.

Camurri et al. Camurri et al. (2017) recently developed a method for contact probability estimation using logistic regression. This one-dimensional classifier learns the normal force threshold at which the contact state transitions, ignoring lateral forces under the

assumption that sufficient friction exists to prevent slip. The resulting probability per endeffector is used to weight the corresponding measurement in a base state estimator, and a heuristic for modulating the measurement variance to filter out the effect of impacts is introduced. Although this estimator performs better than one using a fixed threshold, the classifier requires significant effort to train - ground truth is obtained manually as the contact sequence which minimizes estimation error. Further, all results shown are for walking on flat ground where slipping does not occur. Finally, only one dimension (normal force) is considered.

In contrast, we develop a contact estimator which:

- is completely unsupervised and model-free
- uses only common, proprioceptive sensors (endeffector force/torque and IMU)
- estimates the probability of contact in all six endeffector DoFs independently

We test this contact estimator for use in base state estimation by modulating the measurement uncertainty associated with each endeffector DoF using the corresponding estimated probability of contact. The following section details the motivation and setup for this approach.

VI.2 Background

VI.2.1 Motivation

A difficult question arises when designing an estimator for contact state: what does it mean for an endeffector to be *in contact*? Most approaches treat the endeffector as fixed to contact surface if the normal force exceeds a chosen threshold, however contact truly occurs when the assumed endeffector constraints are satisfied - these statements are not always the same. The six DoF endeffector constraints are equivalent to enforcing that the feet cannot *slip* or *rotate*. An endeffector will not slip if the static friction constraint

$$\sqrt{F_x^2 + F_y^2} \leq \mu_{x,y} F_z \tag{VI.1}$$

is satisfied, where F is the contact force and $\mu_{x,y}$ is the translational coefficient of friction. Likewise, the endeffector will not rotate if the CoP and rotational friction constraints

$$\begin{bmatrix} -\tau_y/F_z \\ \tau_x/F_z \end{bmatrix} \leq \begin{bmatrix} CoP_x \\ CoP_y \end{bmatrix} \tag{VI.2}$$

$$|\tau_z| \leq \mu_z F_z \tag{VI.3}$$

are satisfied, where τ is the contact torque, μ_z is the rotational coefficient of friction and CoP_x, CoP_y denote the contact support polygon bounds which are functions of contact surface geometry.

Since a sufficiently-high normal force F_z would guarantee that inequalities (VI.1-VI.3) are satisfied regardless of the other contact wrench dimensions, most estimation approaches simply threshold F_z Bloesch et al. (2012), Fallon et al. (2014), Faraji et al. (2015). However, this is restrictive especially on rough terrain where low friction and difficult surface geometry make slip and rotation likely even at high normal force values. It also results in a one-dimensional contact state estimate as in Hwangbo et al. (2016), Camurri et al. (2017), whereas the contact constraint is truly six-dimensional.

VI.2.2 Sensing for Clustering

As discussed in the previous section, contact constraints are invalid when an endeffector slips and/or rotates, which is caused by a violation of friction and/or CoP constraints; these constraints depend on the contact wrench and surface properties (friction coefficients and geometry). Rather than estimate these properties, we seek to cluster measured contact wrench data to directly learn constraint probabilities.

All experiments in this work are performed in the SL simulation environment Schaal (2007); we add simulated random-walk biases b_F and b_τ, along with simulated Gaussian noise processes w_F and w_τ, to the true force F and torque τ measurements:

$$F = \bar{F} + b_F + w_F \tag{VI.4}$$

$$\tau = \bar{\tau} + b_\tau + w_\tau \tag{VI.5}$$

As low-cost IMUs become available, humanoids are being augmented with additional sensing to improve estimation Rotella et al. (2016), Xinjilefu et al. (2016); in order to give structure to the clustering problem, we add a simulated IMU to each endeffector. We model the sensor outputs subject to simulated random-walk biases and thermal noise processes Woodman (2007) as

$$a^{IMU} = R_W^{IMU}(a^W + g) + b_a + w_a \tag{VI.6}$$

$$\omega^{IMU} = R_W^{IMU}\omega^W + b_\omega + w_\omega \tag{VI.7}$$

where $a \in R^3$ and $\omega \in R^3$ are the linear acceleration and angular velocity, respectively. $R_W^{IMU} \in SO(3)$ is the rotation from world to IMU frame and g is gravity. Sensors are assumed to be aligned with the endeffector frame, however their positions relative to this frame origin are not required.

VI.3 Clustering Setup

Because we seek a continuous measure of contact quality rather than a classifier, we employ Fuzzy C-means (FCM) clustering which results in the soft partitioning of a dataset by allowing each data point to belong to more than one cluster Dunn (1974). This is accomplished by minimizing the cost

$$\sum_{i=1}^{N_p}\sum_{j=1}^{N_c} w_{i,j}^m ||x_i - c_j||^2, \quad m > 1 \tag{VI.8}$$

Table VI.1: Simulated sensor noise standard deviations. Corresponding values for 1kHz sampling rate are shown.

	Continuous	Discrete ($1kHz$)
σ_θ	$0.00000316rad/\sqrt{Hz}$	$0.0001rad$
σ_F	$0.06325N/\sqrt{Hz}$	$2N$
σ_{b_F}	$0.0001N/s/\sqrt{Hz}$	$0.00316N/s$
σ_τ	$0.00316Nm/\sqrt{Hz}$	$0.1Nm$
σ_{b_τ}	$0.0001Nm/s/\sqrt{Hz}$	$0.00316Nm/s$
σ_a	$0.00078m/s^2/\sqrt{Hz}$	$0.02467m/s^2$
σ_{b_a}	$0.0001m/s^3/\sqrt{Hz}$	$0.00316m/s^3$
σ_ω	$0.000523rad/s/\sqrt{Hz}$	$0.01653rad/s$
σ_{b_ω}	$0.000618rad/s^2/\sqrt{Hz}$	$0.01954rad/s^2$

where N_p is the number of data points x_i, N_c is the chosen number of clusters, $w_{i,j}$ is the membership weight of point i belonging to cluster j and m is a constant which can be used to tune the amount of cluster overlap.

This cost is minimized in a manner similar to k-means clustering; first, initial membership weights are randomly assigned. Then, cluster means are computed as

$$c_j = \frac{\sum_{i=1}^{N_p} w_{i,j}^m x_i}{\sum_{i=1}^{N_p} w_{i,j}^m} \tag{VI.9}$$

after which new membership weights are computed with

$$w_{i,j} = \frac{1}{\sum_{k=1}^{N_c} \left(\frac{||x_i - c_j||}{||x_i - c_k||} \right)^{\frac{2}{m-1}}} \tag{VI.10}$$

Eq. (VI.9-VI.10) are iterated until the membership weights converge. Since $\sum_{j=1}^{N_c} w_{i,j} = 1$, we treat $w_{i,j}$ as the probability of point i belonging to cluster j.

We use an FCM implementation from the Python library Scikit-learn Pedregosa et al. (2011) with $N_C = 2$ clusters (corresponding to *contact* and *no contact* states) and default stopping parameters. The "fuzziness" constant is set to $m = 1.2$ which is the default value in most libraries. Increasing this factor can amplify the effect of slip on contact probability, however it also reduces the probability of contact when no slip occurs.

Each data point $x_k \in R^{7T}$ is a time series of the past $T = 20$ samples (at our control rate, $0.020s$). We include a short time-history to improve estimation response time; optimization of this time window is left to future work.

Clustering is performed independently for the six DoF $\{x, y, z, \alpha, \beta, \gamma\}$ of each endeffector, using the full contact wrench and the corresponding IMU dimension from

$$\{a_x^{IMU}, a_y^{IMU}, a_z^{IMU}, \omega_x^{IMU}, \omega_y^{IMU}, \omega_z^{IMU}\}$$

The constraint in the local endeffector frame y direction uses, for example, data points of the form

$$\begin{aligned} x_k = \{&\{F_{x_{k-T}}, \cdots, F_{x_k}\}, \{F_{y_{k-T}}, \cdots, F_{y_k}\}, \{F_{z_{k-T}}, \cdots, F_{z_k}\}, \\ &\{\tau_{x_{k-T}}, \cdots, \tau_{x_k}\}, \{\tau_{y_{k-T}}, \cdots, \tau_{y_k}\}, \{\tau_{z_{k-T}}, \cdots, \tau_{z_k}\}, \\ &\{a_{y_{k-T}}^{IMU}, \cdots, a_{y_k}^{IMU}\}\} \end{aligned} \tag{VI.11}$$

Data from sensors with noise added as in Sec. VI.2.2 is collected and used unfiltered for clustering. Preprocessing entails dimension-wise normalization of all x_k (to ensure that the scale of dimensions such as F_z do not dominate) followed by taking the absolute value (since slip is bi-directional).

VI.4 Base State Estimation

In order to evaluate the utility of the proposed contact estimator, we incorporate it into a base state estimation framework which relies on stationary contact assumptions. In previous work Rotella et al. (2014) we have implemented a kinematics-based estimator

which fuses IMU data and relative base pose measurements to estimate the floating base state of a humanoid (see Chapter III). The estimator measurements take the form

$$s_{p,i} = R(q)(p_i - r) + n_p \tag{VI.12}$$

$$s_{z,i} = \exp(n_z) \otimes q \otimes z_i^{-1} \tag{VI.13}$$

where $R(q)$ denotes the rotation matrix corresponding to the estimated base quaternion q, p_i and z_i are the estimated foot i position and quaternion respectively, and n_p and n_z are position and orientation measurement noise vectors (see Rotella et al. (2014) for more details). In most approaches for legged robots, the variances of n_p and n_z are set to constant, tuned values and the measurements are dropped from the filter when the endeffector loses contact, determined based on a fixed normal force threshold.

In contrast, in this work we set the contact state (which determines active measurements) and measurement noise variance using the output of the probability estimator. When the probability of contact vector $P_{contact} \in R^6$ exceeds $P_i = 0.5$ in every dimension i, we consider the endeffector *in contact* and use the corresponding measurements. Further, we set the measurement noise covariance matrix as

$$\Sigma = E[nn^T] = r^2 I + \alpha(I - \text{diag}(P_{contact})) \tag{VI.14}$$

where r is the nominal measurement noise standard deviation (sometimes tuned separately for position and orientation) and α is a scaling factor for the probability-dependent term (we choose $\alpha = 1$ for simplicity). The covariance thus converges to its constant value as in Rotella et al. (2014) when $P_{contact} \to 1$. This is conceptually similar to the approach of Camurri et al. (2017) but requires less tuning and considers all six contact dimensions.

Since clustering is performed in the endeffector frame, the covariance must be transformed into the base frame where the base state estimator measurement is expressed. This is accomplished with

$$\hat{\Sigma} = R\Sigma R^T, \quad R = \text{blockdiag}(R^{Base}_{Endeff}, R^{Base}_{Endeff}) \in R^{6\times 6}$$

where $R^{Base}_{Endeff} \in R^{3\times3}$ is the rotation from endeffector to base frame (a function of kinematics and joint angles only).

VI.5 Experiments and Results

We perform a number of experiments to evaluate the performance of the proposed estimator and analyze its properties. All experiments are performed in the SL simulator Schaal (2007) during a 60 second rough terrain walking task with simulated joint angle, IMU and contact wrench sensor noise as in Table VI.1; noisy data is used for clustering, contact estimation and base state estimation. Control is computed using non-noisy sensor data and ideal base state estimation, however we investigate using the proposed contact estimator for closed-loop control in Sec. VI.6. Walking velocity commands were recorded from user input and played back, producing repeatable trajectories across experiments. The rough terrain consists of raised patches with a friction coefficient of 0.4 (half the normal friction in SL). To account for the effect of noise and slight contact differences, Root-Mean-Squared Error (RMSE) for experiments in this section was computed by averaging performance across ten trials.

VI.5.1 Contact Clustering Results

Sensor data was recorded from the rough terrain walking task and clustered offline as detailed in Sec. VI.3 (clustering takes on the order of a few seconds). The cluster means were then used to compute contact probability during a similar walking task; the results are shown in Fig. VI.1.

We focus on one contact cycle in the lower portion of Fig. VI.1 to investigate the clustering results more closely. Slip first occurs in the y direction at (1), causing the corresponding contact probability to lag behind the other dimensions. Rotational slip in α is also present during loading, however on a smaller scale. Slip then occurs in x because the foot is not sufficiently loaded while the robot tries to create force in $-x$ to decelerate the center of mass; once F_z increases, there is sufficient friction to stop slipping and a negative

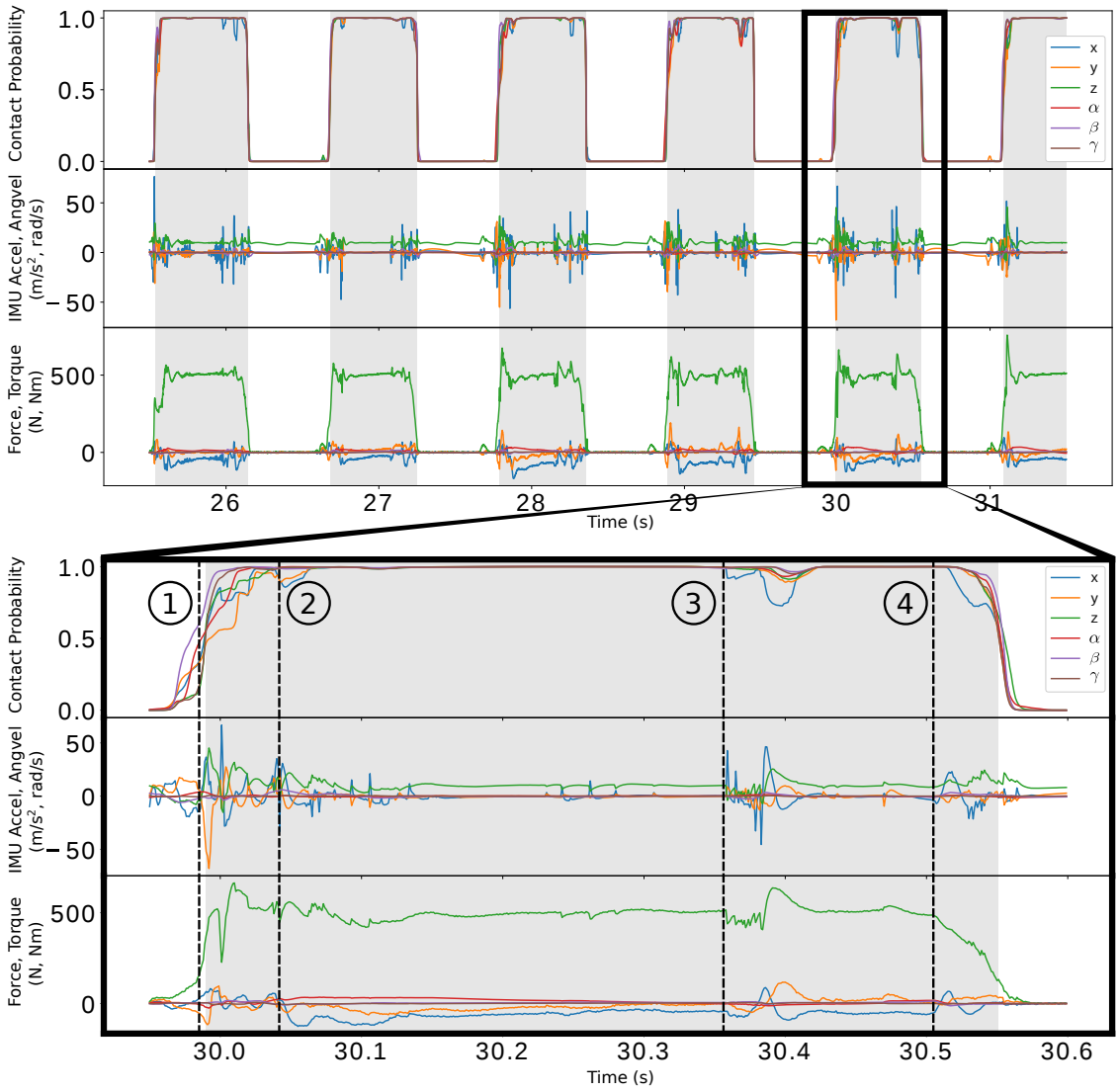


Figure VI.1: The top portion shows the six-dimensional contact probability resulting from a rough terrain walking task (top) along with the measured IMU linear acceleration and angular velocity (middle) and measured contact force and torque (bottom). The portions in gray denote contact according to the probability estimator (all $P_i > 0.5$). The lower portion of the plot shows a zoomed view of one contact cycle with several distinctive contact events highlighted for discussion in Sec. VI.5.1.

F_x is sustained from (2) on. A drop in F_z during single support at (3) again causes slip, leading to a decrease in contact probability in all dimensions. Finally, slip in x again occurs at (4) as the foot is being unloaded. These are only a few highlights of the complex contact interaction shown, however they aid in understanding where/why slip can occur.

VI.5.2 Base State Estimation Threshold

Since we evaluate the proposed contact probability estimator against a typical humanoid base state estimator with a fixed normal force threshold for contact Rotella et al. (2014), we first perform experiments to optimize the chosen threshold. Performance is evaluated by computing the RMSE for the base position and yaw angle as these four states are always unobservable without adding exteroceptive sensing. The normal force thresholds $\{10N, 40N, 100N, 200N, 400N\}$ were tested, with performance averaged across ten trials each; the results are shown in Fig. VI.2. The RMSE mostly decreased for increasing thresholds, with $200N$ resulting in the best performance. A threshold of $400N$ removes the double support period from estimation entirely, resulting in more error. We use a fixed threshold of $200N$ for the baseline estimator in experiments in the remainder of this work.

VI.5.3 Clustering-Based State Estimation

We evaluate the base state estimator detailed in our previous work using both a fixed normal force threshold (as is commonly done) and using the proposed clustering-based contact probability estimator for the same rough terrain walking task. The base state estimators are identical other than the measurement noise covariance matrix modulation of Eq. (VI.14). As shown in Fig. VI.3, using the contact clustering for base state estimation considerably reduces the RMSE.

VI.5.4 Clustering Training Data

In order to test how well the clustering-based estimator generalizes to different types of terrain, we performed clustering using data from two different tasks: one which walks over *rough* terrain (as in all other experiments) and one which walks in place on *flat* ground.

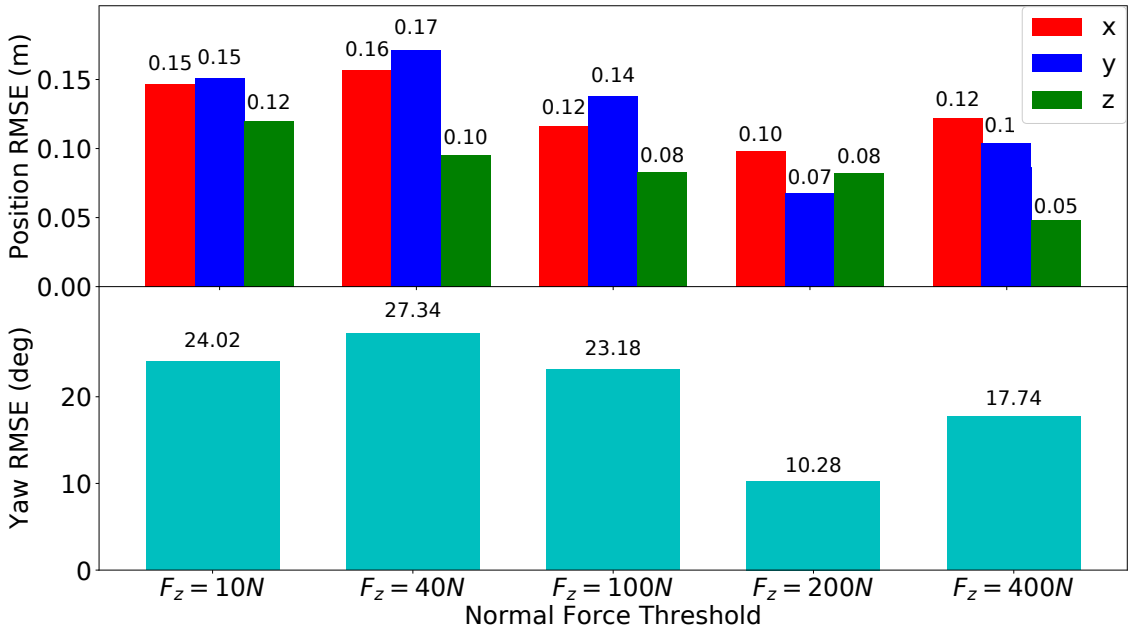


Figure VI.2: Root Mean Squared Error (RMSE) for estimation of the unobservable base position (top) and yaw (bottom) for different normal force thresholds.

We then tested both contact estimators with separate base state estimators on the same rough terrain walking task; the resulting estimation errors are shown in Fig. VI.4.

Surprisingly, the estimator trained on flat ground walking data performs roughly equally-well, despite having been trained on a much different dataset than was used for testing. This is a desirable characteristic because obtaining data from rough terrain walking on a real robot is difficult, especially without accurate state estimation already in place.

We also wish to test how well the clustering-based estimator generalizes to different gaits. We perform clustering using data from flat ground walking in varying directions using three different gaits. The default gait used for walking in this work has a single support period of $0.5s$ and a double support period of $0.05s$; we denote this the *fast* gait. We also perform clustering on *slow* gait data (single support period of $1.0s$, double support period of $0.5s$). Finally, we cluster using data from a *mixed* gait which varies throughout the task between fast and slow. We then test the clustering-based estimators for these gaits

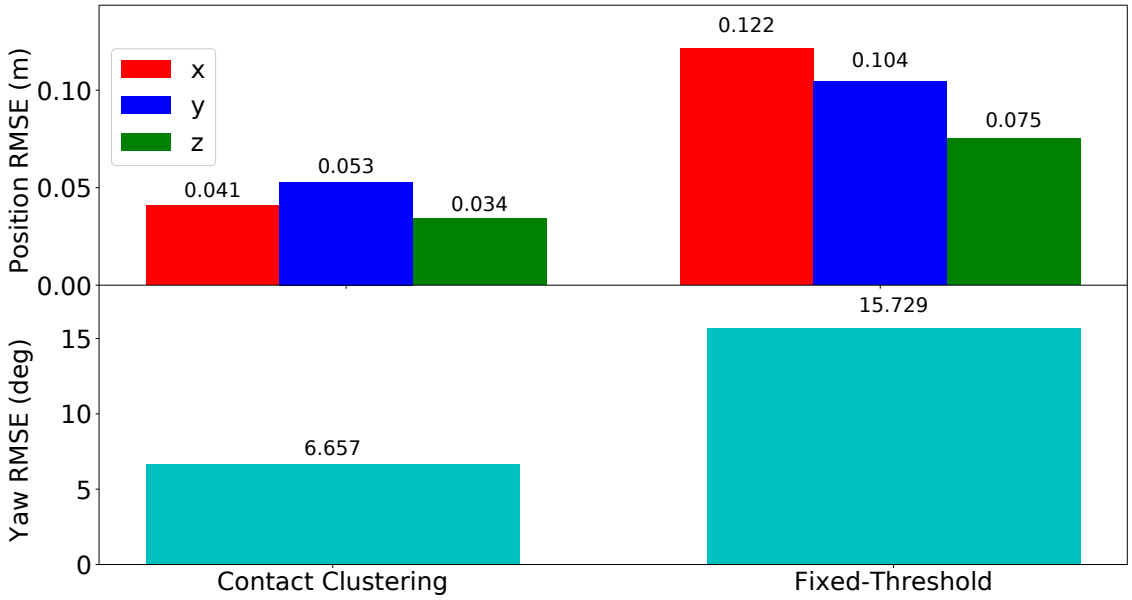


Figure VI.3: Root Mean Squared Error (RMSE) for estimation of the unobservable base position (top) and yaw (bottom) for the contact probability-based base state estimator and the fixed normal force threshold base state estimator.

for a mixed gait walk-in-place task on a patch of rough terrain (varying the gait during a normal walking task over rough terrain is too unstable). The results are shown in Fig. VI.5.

The main conclusion which can be drawn from this study is that the best performance is obtained using the clustering trained on the mixed gait, as expected. However, the slow gait clustering generalizes much better than the fast gait clustering. The fixed-threshold base state estimator (denoted BSE) also performs quite well for this task, however because this was a walk-in-place there was mainly foot rotation and minimal slip; as seen from other tests, the clustering-based base state estimator performs much better when slip occurs. Further investigation into the effect of training data gait is left to future work.

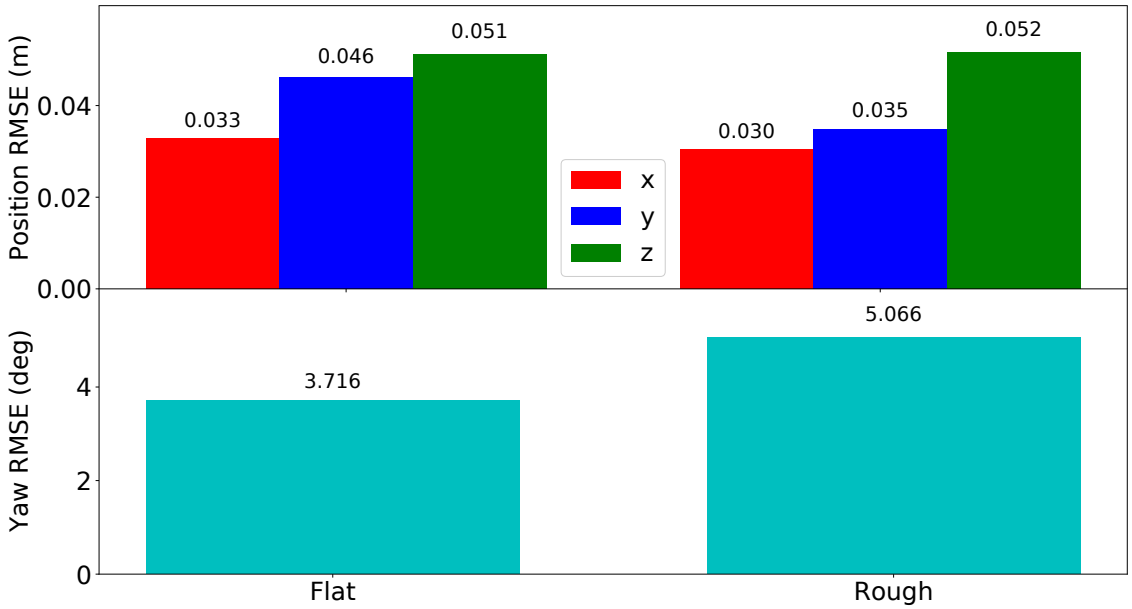


Figure VI.4: Root Mean Squared Error (RMSE) for estimation of the unobservable base position (top) and yaw (bottom) for different training datasets.

VI.5.5 IMUs for Clustering Versus Estimation

As motivated in Sec. VI.2.2, the use of endeffector IMU data in addition to contact wrench data essentially supervises the clustering problem, since the accelerometer and gyroscope capture linear and rotational slip. We expect this sensor data to embed structure in the resulting clusters, meaning that IMUs should not be required when running the contact estimator afterwards. To test this, we cluster using data points as in Eq. (VI.11) but perform clustering-based state estimation with both a) the full data points including IMUs and b) without IMUs (dropping the last portion of Eq. (VI.11)). The resulting estimation errors are shown in Fig. VI.6.

Although the RMSE is slightly lower in all dimensions when using the IMU data, performance is not considerably changed when it is removed. This is a very useful property because it means that the endeffector IMUs can be removed after initially collecting data for clustering. While some robots are designed with endeffector IMUs, most are not; using this

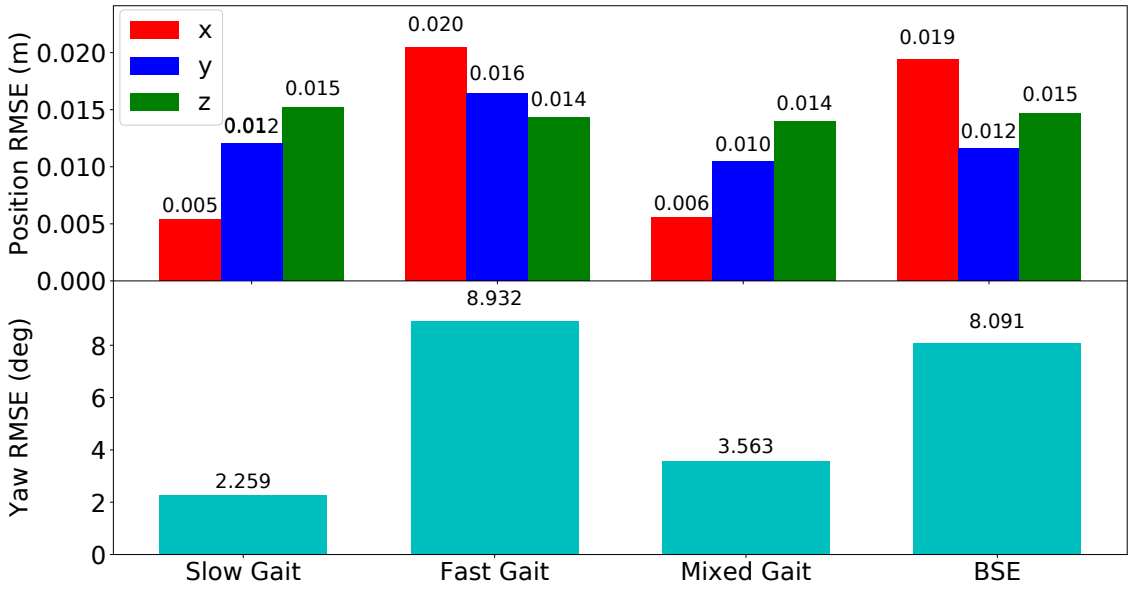


Figure VI.5: Root Mean Squared Error (RMSE) for estimation of the unobservable base position (top) and yaw (bottom) for walking in place on a patch of rough terrain with a varying gait using clustering trained on three different gait types as well as for the fixed-threshold base state estimator (BSE).

clustering method would involve temporarily attaching IMUs as in Rotella et al. (2016), Xinjilefu et al. (2016). This is reasonable for training, however attaching these sensors permanently involves designing rigid mounts, routing cables and protecting them from collisions with the environment. The ability to remove IMUs after training the estimator is highly advantageous when working with real hardware.

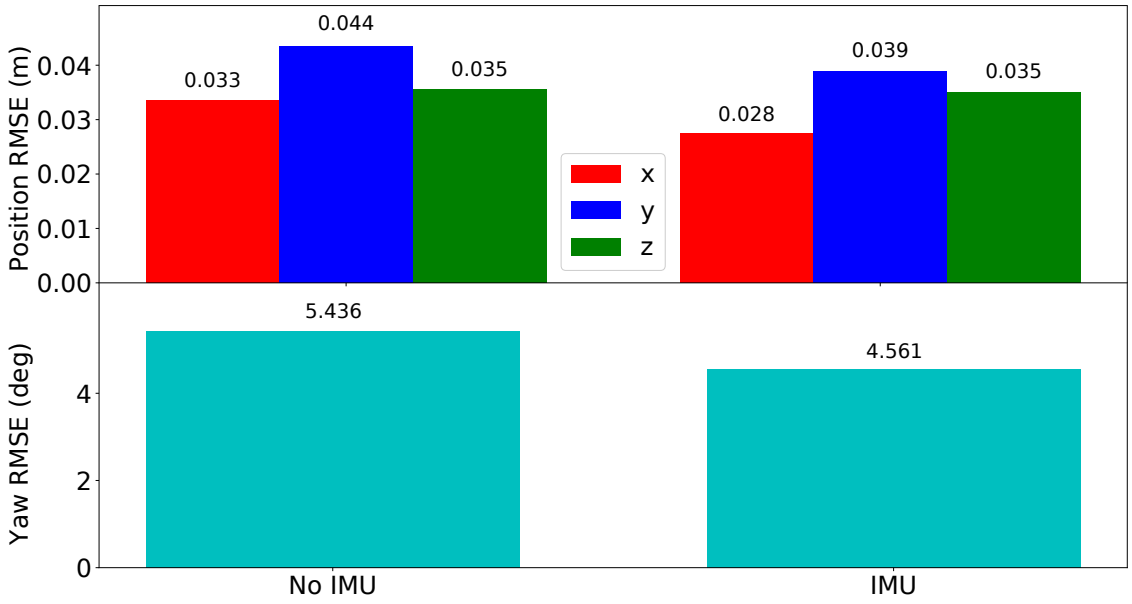


Figure VI.6: Root Mean Squared Error (RMSE) for estimation of the unobservable base position (top) and yaw (bottom) with and without using IMU data for online contact estimation.

VI.6 Clustering-Based Contact Estimation for Control

VI.6.1 Closed-Loop Control using Base State Estimation

While the primary focus of this work has been on the development and evaluation of a contact probability estimator for use in base state estimation, the proposed method has applications in humanoid control as well. The most direct application is the use of an improved base state estimator in a walking controller such as the one used to generate data in this work; see Chapter II for details.

The estimated base pose is crucial in such a control pipeline for computing both dynamic model parameters and feedback control for endeffector tracking, as discussed in III. The use of our contact probability estimator in this context improves control considerably,

allowing the robot to walk on the rough terrain for a longer time before falling due to an accumulation of base state estimation error (see the video accompanying the paper).

VI.6.2 Contact Probability-Based Inverse Dynamics Constraints

The quadratic program-based inverse dynamics solver used for all experiments in this chapter, as detailed in II.3, is written as the optimization problem

$$\min_{\ddot{q},\lambda} ||\ddot{x} - \ddot{x}_{des}||^2_{W_x} + ||P_{null}\ddot{q} - \ddot{q}_{des}||^2_{W_q} + ||\lambda - \lambda_{des}||^2_{W_\lambda} + ||\tau - \tau_{des}||^2_{W_\tau}$$

subject to the equality constraints

$$M_l(q)\ddot{q} + h_l(q,\dot{q}) = J_{c,l}^T\lambda$$
$$J_c\ddot{q} + \dot{J}_c\dot{q} = 0$$

As in base state estimation, the contact state and corresponding constraint dimension change instantaneously based on the measured normal force and a chosen threshold. This is problematic because it results in torque discontinuities in the solution to the above optimization which can destabilize the controller. We thus seek a method of smoothly varying the contact constraints during walking.

Recently, Feng (2016) briefly introduced the concept of structural change smoothing in order to avoid discontinuities when transitioning between contact states. This was accomplished by modifying the above optimization problem as

$$\min_{\ddot{q},\lambda,\tau} ||\ddot{x}-\ddot{x}_{des}||^2_{W_x}+||P_{null}\ddot{q}-\ddot{q}_{des}||^2_{W_q}+||\lambda-\lambda_{des}||^2_{W_\lambda}+||\tau-\tau_{des}||^2_{W_\tau}+||\lambda||^2_{W_{c,\lambda}}+||J_c\ddot{q}+\dot{J}_c\dot{q}||^2_{W_{c,J_c}} \tag{VI.15}$$

where the final two terms represent a cost on creating contact wrenches and a cost on violating contact constraints, respectively. This optimization problem is subject to the modified equality constraints

$$M(q)\ddot{q} + h(q,\dot{q}) = S^T\tau + J^T\lambda$$

Note that the contact equality constraints have been moved to the cost, while the dynamics equality constraint remains however with the important distinction that the *full* endeffector Jacobian J is used rather than the contact Jacobian.

This effectively means that all endeffectors are considered in contact at all times, while the new cost terms actually determine the contact constraints. When an endeffector loses contact, the cost weight W_{c,J_c} is ramped down from a large value to zero (allowing the endeffector to accelerate), while $W_{c,\lambda}$ is simultaneously ramped up from zero to a large value (allowing a wrench to be created). The opposite is done when a contact is gained.

If we change the new cost weights instantaneously, we obtain the same behavior as with hard constraints; instead, we can smoothly ramp the soft constraints on and off using, for example, a low pass filter on the planned contact sequence.

VI.7 Summary

In this chapter, we have:

- Discussed the notion of contact probability as it relates to contact constraints for legged robots.
- Introduced a contact probability estimator based on the Fuzzy C-means soft clustering algorithm using endeffector contact wrench and IMU data.
- Developed a principled method for incorporating contact probabilities into the base state estimator of Chapter III by modulating the measurement noise covariance.
- Evaluated the estimator in simulation using simulated endeffector proprioceptive sensing with realistic added noise, resulting in lower base state estimation RMSE for unobservable states than using a fixed normal force threshold for contact detection.
- Demonstrated that the proposed estimator:

- Can be run without endeffector IMUs after training, making it convenient to run online using only contact wrench data.
- Performs equally-well using training data from either flat ground or rough terrain walking, allowing simple training and generalization to difficult terrain.
- Generalizes to different gait timings as long using a combination of fast and slow gait training data.
- Can be used in an improved base state estimator for closed-loop inverse dynamics control, allowing the robot to remain stable during rough terrain walking for a longer period of time.

Future work will include further analysis of the properties of this contact estimator as well as a more low-level control application in which endeffector constraints in inverse dynamics are smoothly varied according to the contact probability.

Chapter VII

Conclusions

In this work, we have presented solutions to a number of estimation problems which are crucial in attaining robust, general-purpose humanoid robot control. The estimation of the base pose of a floating base robot (Chapter III) is a necessary starting point for investigating such challenges; this quantity cannot be observed directly through proprioceptive sensing, yet it is required to compute the dynamics on which all whole-body controllers are based.

Closely-tied to the base pose rate of change is the momentum of the system, which is used extensively to compute stability criteria based on both simplified and exact models of the robot dynamics. Obtaining low-noise estimates of linear and angular momentum while compensating for the effects of modeling error and externally-applied wrenches is essential for balance and stepping stabilization using model predictive control (Chapter IV).

In a parallel line of research, we have introduced methods for utilizing inexpensive inertial sensors integrated into the links of a humanoid in order to compute joint state derivatives subject to unknown pose offsets which can be compensated through calibration (Chapter V). Joint velocity signals with less noise and delay than filtered, differentiated angle sensors are crucial in obtaining well-damped joint and Cartesian feedback control, resulting in better tracking and disturbance rejection.

Finally, we have investigated contact state estimation, which presents serious challenges to model-based control and estimation methods; both require specifying contact constraint information to render them solvable. Rather than tune heuristics based on closed-loop performance using these constraints, we take a learning-based approach by performing

clustering on proprioceptive sensor data from endeffector contact wrench and inertial sensors (Chapter VI). The resulting contact probability estimates have direct application in base state estimation and inverse dynamics control, providing a continuous measure of contact which helps prevent destabilizing discontinuities.

In the process of developing new estimators for humanoid robots, we have developed a considerable library of contemporary methods for planning and control as well (Chapter II). The optimization-based inverse dynamics solver used in this work is a combination of successful methods used in the DARPA Robotics Challenge, and was implemented specifically in order to test possible control extensions (for example, smooth constraint switching, endeffector impedance control and external wrench compensation). Our joystick-based walking controller combines this solver with a simplified model-based planner in a state machine framework, making it simple to test new estimation and controller modules in a user-controlled, closed-loop walking task. These modules can easily be swapped out and tested against the baseline walking controller and will thus prove useful to others working on the same platform.

Outlook

The work presented in this thesis reflects and builds upon current, state-of-the-art methods for both estimation and control for humanoid robots; while such methods, along with increasing computational power, have advanced the field dramatically over the past few years, much work remains before humanoid robots will be viable options for their target applications. With regards to the work presented in this thesis in particular, we hope to see our estimation methods used in whole-body control for walking in order to overcome the many modeling and sensing challenges which each approach addresses. In addition, we firmly believe that sensing will play an increasingly crucial role in legged locomotion. As sensors become less expensive, we should build as many as possible into our robot designs; this would allow the generation of large amounts of data which can be fed to learning algorithms. Advances in machine learning need not (and may never be able to)

replace model-based methods entirely (nor should they, since rigid body dynamics serves as a great prior), but they should be used more often to augment traditional approaches - the contact estimator introduced in this work is a simple example of this. Generating large amounts of data safely for learning approaches which augment or replace modules of a walking controller is a necessary first step towards potentially "solving" locomotion using deep learning alone. In any case, those of us who work on legged locomotion have plenty of data to work with and little idea how to utilize it - we believe this is a crucial problem which will greatly advance the field when addressed properly.

Appendix A

Kalman Filtering

The Kalman Filter provides an estimate of the state vector x along with a corresponding covariance matrix P which specifies the uncertainty of the estimate. The filter involves 1) propagating the estimates through the system in the *prediction* step to produce *a priori* estimates $\hat{x}_k^-$ and P_k^- and 2) updating the estimates using a measurement in the *update step* to the *a posteriori* estimates $\hat{x}_k^+$ and P_k^+.

However, the standard Kalman Filter is applicable only for state estimation in linear systems. Suppose we have the continuous-time, nonlinear system

$$\dot{x} = f(x, w) \tag{A.1}$$

$$y = h(x, v) \tag{A.2}$$

where $f()$ is the *prediction* model, $h()$ is the *measurement* model and w and v are noise terms. One may simply linearize these models around the current estimate and apply the Kalman Filter equations. This is known as the *Extended Kalman Filter*. While optimality and convergence are no longer guaranteed, this is a common approach for nonlinear estimation.

The EKF algorithm works as follows. First, (A.1) and (A.2) are discretized and the *a priori* state and measurement are computed using the resulting discretized models as follows.

$$\hat{x}_k^- = f(\hat{x}_{k-1}^+) \tag{A.3}$$

$$\hat{y}_k = h(\hat{x}_k^-) \tag{A.4}$$

Next, the process and measurement models (A.1) and (A.2) are linearized around the *a priori* state estimate using a first-order Taylor series expansion to produce the process and measurement Jacobians which are discretized to form F_k and H_k. The prediction step is completed by computing the *a priori* covariance using the linearized prediction model.

$$F_k = \exp\left(\frac{\partial f}{\partial x}|_{\hat{x}_k^-}\right) \tag{A.5}$$

$$H_k = \frac{\partial h}{\partial x}|_{\hat{x}_k^-} \tag{A.6}$$

$$P_k^- = F_k P_{k-1}^+ F_k^T + Q_k \tag{A.7}$$

where Q_k is the discrete process noise covariance and $\exp(\dot{})$. Finally, when a measurement y_k becomes available the state and covariance matrix are updated using the linearized model as follows.

$$S_k = H_k P_k^- H_k^T + R_k \tag{A.8}$$

$$K_k = P_k^- H_k^T S_k^{-1} \tag{A.9}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - \hat{y}_k) \tag{A.10}$$

$$P_k^+ = (I - K_k H_k) P_k^- \tag{A.11}$$

where R_k is the discrete measurement noise covariance. The filter is initialized with an estimate $\hat{x}_0^+$ of the state and corresponding covariance matrix P_0^+ which represents the uncertainty in the chosen initial state.

A.0.1 Handling of Rotational Quantities

The unit quaternion was chosen to represent the base orientation in the original filter due to its theoretical and computational advantages. However, since the quaternion is a non-minimal representation of $SO(3)$, special care must be taken in handling rotational quantities in the EKF.

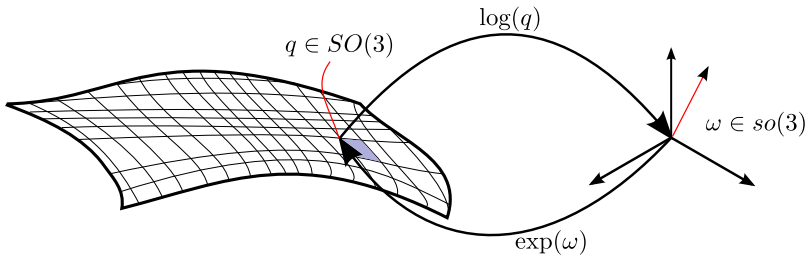


Figure A.1: Illustration of the mapping between the manifold of rotations $SO(3)$ and its tangent space $so(3)$.

First, the exponential map

$$\exp(\omega) = \begin{pmatrix} \sin(\frac{||\omega||}{2})\frac{\omega}{||\omega||} \\ \cos(\frac{||\omega||}{2}) \end{pmatrix} \tag{A.12}$$

is used to relate a quaternion at times k and $k+1$ given an incremental rotation of magnitude $||\omega||$ about the unit vector $\omega/||\omega||$. That is,

$$q_{k+1} = \exp(\omega) \otimes q_k \tag{A.13}$$

where $\otimes$ denotes quaternion multiplication. Roughly speaking, the exponential map is used for "addition" of rotational quantities. Note that the first entry of (A.12) is the vector portion of the quaternion while the second entry is the scalar portion. Also note that there are two self-consistent conventions for quaternion algebra; to avoid such issues, we employ the quaternion conventions of (Trawny and Roumeliotis, 2005).

In the EKF state vector, a quaternion is represented by its corresponding three-dimensional *error rotation vector* $\phi \in so(3)$. The covariance of the orientation represented by the quaternion is thus defined with respect to this minimal representation.

During the update step, the innovations vector e corresponding to a quaternion-valued measurement is computed as the three dimensional rotation vector extracted from the difference between the actual measurement quaternion s and the expected measurement quaternion z. That is,

$$e = \log\left(s \otimes z^{-1}\right) \tag{A.14}$$

where $\log()$ denotes the logarithm mapping an element of $SO(3)$ to its corresponding element of $so(3)$ (the inverse of the exponential map). The above operation extends the notion of subtraction to rotational quantities.

Finally, using the innovations vector as computed above, the state correction vector Δx is computed during the update step as $\Delta x = Ke$ where K is the Kalman gain. While all non-rotational states are updated simply as $\hat{x}_k^+ = \hat{x}_k^- + \Delta x$, a quaternion state is updated using (A.12) as follows.

$$\hat{q}_k^+ = \exp\left(\Delta\phi\right) \otimes \hat{q}_k^- \tag{A.15}$$

For more information on quaternion convention and use, see Appendix B.

Appendix B

Base State EKF Linearization

B.1 Quaternion Review and Conventions

It is important to note that there are two different conventions for using quaternions, each of which is self-consistent; unfortunately, the quaternion algebra used in these conventions is often mixed up in the literature, resulting in inconsistent implementations (Shuster, 1993). We use that proposed as a standard by JPL (Breckenridge, 1979) which was also used in (Trawny and Roumeliotis, 2005) on which this work and that in (Bloesch et al., 2012) was based. A review of quaternions using this convention is given in this section.

Briefly, quaternions are one of several choices for representing $SO(3)$, the Lie group of rotations. The advantage of quaternions over other parameterizations is their numerical properties, efficiency and lack of singularities. We denote a quaternion by

$$q = \begin{pmatrix} q_x \\ q_y \\ q_z \\ q_w \end{pmatrix}$$

where q_w and $q_v = [q_x, q_y, q_z]^T$ are the scalar and vector components, respectively. Note that the ordering of these components does not depend on any conventions and is purely a matter of preference. We choose this ordering here to match the literature but actually implement quaternions in the reverse order to match the SL simulation environment.

All rotations are active, meaning that they act to rotate vectors; the quaternion representing the base orientation is written as $q = q_W^B$ which specifies a rotation from the world frame W to the base frame B. This quaternion corresponds to the rotation matrix $C = C[q]$ which rotates vectors defined in the world frame into the base frame.

Successive rotations about local axes are composed via left-multiplication, ie $R_A^C = R_B^C R_A^B$ represents a rotation from frame A to frame B (given in terms of the frame A basis) followed by a rotation from frame B to frame C (given in terms of the frame B basis). Analogously, we have $q_A^C = q_B^C \otimes q_A^B$ for quaternions where $\otimes$ denotes quaternion multiplication.

A vector in frame A is rotated into frame C as $v^C = R_A^C v^A$, with the reverse transformation given by $v^A = (R_A^C)^{-1} v^C$ where $(R_A^C)^{-1} = R_C^A = (R_A^C)^T$ since this is a rotation matrix (orthogonal matrix with det $= +1$). Vector rotations using quaternions are achieved through conjugation as

$$v^C = q_A^C \otimes \begin{pmatrix} v^A \\ 0 \end{pmatrix} \otimes (q_A^C)^{-1}$$

where $[v^A, 0]^T$ is called a pure quaternion and $(q_A^C)^{-1} = q_C^A = [-q_x, -q_y, -q_z, q_w]$ is the inverse or conjugate quaternion satisfying $q_A^C \otimes (q_A^C)^{-1} = (q_A^C)^{-1} \otimes q_A^C = q_I$ where $q_I = [0, 0, 0, 1]$ is the identity quaternion.

Quaternion multiplication is defined by

$$q \otimes p = \begin{pmatrix} q_w p_x + q_z p_y - q_y p_z + q_x p_w \\ -q_z p_x + q_w p_y + q_x p_z + qy p_w \\ q_y p_x - q_x p_y + q_w p_z + q_z p_w \\ -q_x p_x - q_y p_y - q_z p_z + q_w p_w \end{pmatrix}$$

This can be written more concisely as the matrix vector multiplication

$$q \otimes p = L(q)p = \begin{pmatrix} q_w I - q_v^{\times} & q_v \\ -q_v^T & q_w \end{pmatrix} \begin{pmatrix} p_v \\ p_w \end{pmatrix}$$
$$= R(p)q = \begin{pmatrix} p_w I + p_v^{\times} & p_v \\ -p_v^T & p_w \end{pmatrix} \begin{pmatrix} q_v \\ q_w \end{pmatrix}$$

where $q_v = [q_x, q_y, q_z]^T$ is the vector part of q and

$$q_v^{\times} = \begin{pmatrix} 0 & -q_z & q_y \\ q_z & 0 & -q_x \\ -q_y & q_x & 0 \end{pmatrix}$$

is the skew-symmetric matrix corresponding to the vector q_v.

The rotation matrix corresponding the quaternion q is given by

$$C[q] = (2q_w^2 - 1)I - 2q_w q_v^{\times} + 2q_v q_v^T$$
$$= \begin{pmatrix} 2q_x^2 + 2q_w^2 - 1 & 2(q_x q_y + q_z q_w) & 2(q_x q_z - q_y q_w) \\ 2(q_x q_y - q_z q_w) & 2q_y^2 + 2q_w^2 - 1 & 2(q_y q_z + q_x q_w) \\ 2(q_x q_z + q_y q_w) & 2(q_y q_z - q_x q_w) & 2q_z^2 + 2q_w^2 - 1 \end{pmatrix}$$

where the first equation can be seen as equivalent to Rodrigues' identity using the quaternion exponential map

$$\exp(\omega) = \begin{pmatrix} \sin(\frac{||\omega||}{2}) \frac{\omega}{||\omega||} \\ \cos(\frac{||\omega||}{2}) \end{pmatrix}$$

which represents a rotation of $||\omega||$ about an axis $\omega/||\omega||$ as a quaternion. Letting $\delta\phi$ be an infinitesimal rotation, we see that

$$\delta q = \exp(\delta\phi) \approx \begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix}$$

is the first-order approximation of an incremental quaternion. It follows from the definition of $C[q]$ that we have the first-order approximation

$$C[\delta q] \approx I - \delta\phi^{\times}$$

This can also be seen as the first-order approximation of the exponential map for rotation matrices

$$\exp(\delta\phi^{\times}) = \sum_{i=0}^{\infty} \frac{(-\delta\phi^{\times})^i}{i!} \approx I - \delta\phi^{\times}$$

It follows that the first-order expansion of q about a nominal quaternion $\bar{q}$ can be written in matrix form as

$$C[\delta q \otimes \bar{q}] = C[\delta q]C[\bar{q}] = (I - \delta\phi^{\times})\bar{C}$$

where $\bar{C} = C[\bar{q}]$. This approximation will be used in the derivations of the linearized filter dynamics in the next section.

The derivative of a quaternion is related to the angular velocity ω by the equation

$$\dot{q} = \frac{1}{2}\begin{pmatrix} \omega \\ 0 \end{pmatrix} \otimes q$$

and the first-order approximation of $\dot{\delta q}$ is given by

$$\dot{\delta q} \approx \begin{pmatrix} \frac{1}{2}\dot{\delta\phi} \\ 0 \end{pmatrix}$$

B.2 Linearization

Linearization of both the process and measurement models is performed analytically by expanding the filter states about their expected values using first-order Taylor series approximations. Products of small deviations are considered to be negligible, ultimately resulting in linear equations in terms of state deviations.

B.2.1 Process Model

Position

The original process model for the time-evolution of the position is

$$\dot{r} = v$$

Letting $r \approx \bar{r} + \delta r$ and $v \approx \bar{v} + \delta v$ leads to

$$\frac{d}{dt}(r + \delta r) = \dot{r} = \bar{v} + \delta v$$

From which it follows that

$$\dot{\bar{r}} + \dot{\delta r} = \bar{v} + \delta v$$

However, we know that the expected value of $\dot{r}$ is $\dot{\bar{r}} = \bar{v}$, so we finally have

$$\dot{\delta r} = \delta v$$

Velocity

The original process model for the time-evolution of the velocity is

$$\dot{v} = C^T(\tilde{f} - b_f - w_f) + g$$

Let $C \approx (I - \delta\phi^{\times})\bar{C}$, $v \approx \bar{v} + \delta v$, and $b_f \approx \bar{b}_f + \delta b_f$ so that

$$\frac{d}{dt}(\bar{v} + \delta v) = \dot{v} = \bar{C}^T(I + \delta\phi^{\times})(\tilde{f} - (\bar{b}_f + \delta b_f) - w_f) + g$$

Simplifying yields

$$\dot{\bar{v}} + \dot{\delta v} = \bar{C}^T(I + \delta\phi^{\times})(\bar{f} + \delta f) + g$$

where we have defined $\bar{f} = \tilde{f} - \bar{b}_f$ and $\delta f = -\delta b_f - w_f$ to be the "large-signal" and "small-signal" accelerations as in (Sola, 2014). Expanding the right hand side yields

$$\dot{\bar{v}} + \dot{\delta v} = \bar{C}^T\bar{f} + \bar{C}^T\delta f + \bar{C}^T\delta\phi^{\times}\bar{f} + \bar{C}^T\delta\phi^{\times}\delta f + g$$

Recognizing that $\dot{\bar{v}} = \bar{C}^T\bar{f} + g$ is the expected value of the velocity process model equation, we are left with

$$\dot{\delta v} = \bar{C}^T\delta f + \bar{C}^T\delta\phi^{\times}\bar{f} + \bar{C}^T\delta\phi^{\times}\delta f$$

The last term on the right hand side is the (cross) product of two small vectors and can thus be neglected. This leaves

$$\dot{\delta v} = \bar{C}^T\delta f + \bar{C}^T\delta\phi^{\times}\bar{f}$$

Using the fact that $a^{\times}b = -b^{\times}a$ and expanding the expression for δf finally results in

$$\dot{\delta v} = -\bar{C}^T\bar{f}^{\times}\delta\phi - \bar{C}^T\delta b_f - \bar{C}^T w_f$$

B.2.2 Quaternion

The original process model for the time-evolution of the quaternion is

$$\dot{q} = \frac{1}{2}\begin{pmatrix}\tilde{\omega} - b_\omega - w_\omega \\ 0\end{pmatrix} \otimes q$$

We define a deviation δq from the expected value $\bar{q}$ of the pose q by $q = \delta q \otimes \bar{q}$ and let $b_\omega \approx \bar{b}_\omega + \delta b_\omega$ so that

$$\dot{q} = \frac{d}{dt}(\delta q \otimes \bar{q})$$
$$\frac{1}{2}\begin{pmatrix}\tilde{\omega} - (\bar{b}_\omega + \delta b_\omega) - w_\omega \\ 0\end{pmatrix} \otimes q = \dot{\delta q} \otimes \bar{q} + \delta q \otimes \dot{\bar{q}}$$

Simplifying and using the fact that $\dot{\bar{q}} = \frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix} \otimes \bar{q}$ yields

$$\frac{1}{2}\begin{pmatrix}\tilde{\omega} - (\bar{b}_\omega + \delta b_\omega) - w_\omega \\ 0\end{pmatrix} \otimes q = \dot{\delta q} \otimes \bar{q} + \delta q \otimes \left(\frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix} \otimes \bar{q}\right)$$

Multiplying on the right of both sides by $\bar{q}^{-1}$ and recognizing that $\delta q = q \otimes \bar{q}^{-1}$ yields

$$\frac{1}{2}\begin{pmatrix}\tilde{\omega} - (\bar{b}_\omega + \delta b_\omega) - w_\omega \\ 0\end{pmatrix} \otimes \delta q = \dot{\delta q} + \delta q \otimes \frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix}$$

Solving for $\dot{\delta q}$ yields

$$\begin{aligned}\dot{\delta q} &= \frac{1}{2}\begin{pmatrix}\tilde{\omega} - (\bar{b}_\omega + \delta b_\omega) - w_\omega \\ 0\end{pmatrix} \otimes \delta q - \delta q \otimes \frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix} \otimes \delta q - \delta q \otimes \frac{1}{2}\begin{pmatrix}\tilde{\omega} - \bar{b}_\omega \\ 0\end{pmatrix} + \frac{1}{2}\begin{pmatrix}-\delta b_\omega - w_\omega \\ 0\end{pmatrix} \otimes \delta q\end{aligned}$$

where the second step results from the fact that we can split a pure quaternion into the sum of multiple pure quaternions (since they are just vectors) and distribute them over quaternion multiplication. For the quaternion δq corresponding to the small rotation $\delta\phi$, we may write $\delta q \approx \begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix}$ and thus $\dot{\delta q} \approx \begin{pmatrix} \frac{1}{2}\dot{\delta\phi} \\ 0 \end{pmatrix}$ so that

$$\begin{pmatrix} \frac{1}{2}\dot{\delta\phi} \\ 0 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} \tilde{\omega} - \bar{b}_\omega \\ 0 \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix} - \begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix} \otimes \frac{1}{2}\begin{pmatrix} \tilde{\omega} - \bar{b}_\omega \\ 0 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} -\delta b_\omega - w_\omega \\ 0 \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix}$$

Using the fact that quaternion products can be written as matrix-vector products as shown in the previous section and letting $\hat{\omega} = \tilde{\omega} - \bar{b}_\omega$, we can write the above as

$$\begin{aligned} \begin{pmatrix} \frac{1}{2}\dot{\delta\phi} \\ 0 \end{pmatrix} &= \frac{1}{2}\left[\begin{pmatrix} -\hat{\omega}^\times & \hat{\omega} \\ -\hat{\omega}^T & 0 \end{pmatrix} - \begin{pmatrix} \hat{\omega}^\times & \hat{\omega} \\ -\hat{\omega}^T & 0 \end{pmatrix}\right]\begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} -(-\delta b_\omega - w_\omega)^\times & (-\delta b_\omega - w_\omega) \\ -(-\delta b_\omega - w_\omega)^T & 0 \end{pmatrix}\begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix} -2\hat{\omega}^\times & 0 \\ 0^T & 0 \end{pmatrix}\begin{pmatrix} \frac{1}{2}\delta\phi \\ 1 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} -\delta b_\omega - w_\omega \\ 0 \end{pmatrix} \end{aligned}$$

where the second step results from simplifying and eliminating products of small quantities (products of deviations and products of noise with deviations) in the second term. Multiplying out this expression yields the equations

$$\begin{aligned} \frac{1}{2}\dot{\delta\phi} &= -\frac{1}{2}\omega^\times\delta\phi - \frac{1}{2}(\delta b_\omega + w_\omega) \\ 0 &= 0 \end{aligned}$$

where the first equation results in the linearized orientation dynamics

$$\dot{\delta\phi} = -\omega^\times\delta\phi - \delta b_\omega - w_\omega$$

and the second equation ensures consistency.

Foot Position

The original process model for the time-evolution of the position of the i^{th} foot is

$$\dot{p} = C^T w_p$$

Again, let $C = (I - \delta\phi^\times)\bar{C}$ so that we have $C^T \approx \bar{C}^T(I + \delta\phi^\times)$. Also let $p \approx \bar{p} + \delta p$ so that

$$\frac{d}{dt}(\bar{p} + \delta p) = \dot{p} = \bar{C}^T(I + \delta\phi^\times)w_p$$

Simplifying the above yields

$$\dot{\bar{p}} + \dot{\delta p} = \bar{C}^T w_p + \bar{C}^T \delta\phi^\times w_p$$

The second term is the cross product of a state deviation and a (small) noise vector and thus is neglected. Further, we know that

$$\dot{\bar{p}} = E[\dot{p}] = E[C^T w_p] = E[w_p] = 0$$

since w_p is zero-mean and since the rotation matrix C does not alter the statistics of w_p. It directly follows that

$$\dot{\delta p} = \bar{C}^T w_p$$

Accelerometer/Gyroscope Bias

The original process model for the time-evolution of the accelerometer bias is

$$\dot{b}_f = w_{b_f}$$

Letting $b_f \approx \bar{b}_f + \delta b_f$ yields

$$\frac{d}{dt}(\bar{b}_f + \delta b_f) = \dot{b}_f = w_{b_f}$$

It follows that

$$\dot{\bar{b}}_f + \delta\dot{b}_f = w_{b_f}$$

However, we know that $\dot{\bar{b}}_f = E[\dot{b}_f] = E[w_{b_f}] = 0$ since w_{b_f} is zero mean and thus

$$\delta\dot{b}_f = w_{b_f}$$

Likewise, for the gyroscope bias we have

$$\delta\dot{b}_\omega = w_{b_\omega}$$

Foot Quaternion

The continuous process model for the time-evolution of the foot quaternion is

$$\dot{z} = \frac{1}{2}\begin{pmatrix} w_z \\ 0 \end{pmatrix} \otimes z$$

Let $z \approx \delta z \otimes \bar{z}$ so that $\dot{z} = \delta\dot{z} \otimes \bar{z} + \delta z \otimes \dot{\bar{z}}$. Then we have

$$\delta\dot{z} \otimes \bar{z} + \delta z \otimes \dot{\bar{z}} = \frac{1}{2}\begin{pmatrix} w_z \\ 0 \end{pmatrix} \otimes z$$

However, $\dot{\bar{z}} = 0$ is the expected value of $\dot{z}$ since $E[w_z] = 0$. We are thus left with

$$\delta\dot{z} \otimes \bar{z} = \frac{1}{2}\begin{pmatrix} w_z \\ 0 \end{pmatrix} \otimes z$$

Multiplying both sides on the left by $\bar{z}^{-1}$ and using the fact that $\delta z = z \otimes \bar{z}^{-1}$, we have

$$\delta\dot{z} = \frac{1}{2}\begin{pmatrix} w_z \\ 0 \end{pmatrix} \otimes \delta z$$

Again, writing the quaternion product as a matrix-vector product leads to

$$\begin{pmatrix} \frac{1}{2}\delta\dot{\theta} \\ 0 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} -w_z^{\times} & w_z \\ w_z^T & 0 \end{pmatrix}\begin{pmatrix} \delta\theta \\ 0 \end{pmatrix}$$

Multiplying out the above yields

$$\begin{pmatrix} \frac{1}{2}\delta\dot{\theta} \\ 0 \end{pmatrix} = \begin{pmatrix} -\frac{1}{4}w_z^{\times}\delta\theta + \frac{1}{2}w_z \\ \frac{1}{4}w_z^T\delta\theta \end{pmatrix}$$

After eliminating products of small quantities, the first equation yields

$$\delta\dot{\theta} = w_z$$

as desired, and the second yields $0 = 0$ ensuring consistency.

B.2.3 Measurement Model

Relative Foot Position

The relative foot position measurement is given by (for a single foot)

$$s_p = C(p - r) + n_p$$

Letting $s_p \approx \bar{s}_p + \delta s_p$, $C \approx (I - \delta\phi^{\times})\bar{C}$, $p \approx \bar{p} + \delta p$ and $r \approx \bar{r} + \delta r$ we have

$$\bar{s}_p + \delta s_p = s_p = (I - \delta\phi^{\times})\bar{C}((\bar{p} + \delta p) - (\bar{r} + \delta r)) + n_p$$

Expanding the above yields

$$\bar{s}_p + \delta s_p = \bar{C}(\bar{p} - \bar{r}) + \bar{C}(\delta p - \delta r) - \delta\phi^{\times}\left(\bar{C}(\bar{p} - \bar{r})\right) - \delta\phi^{\times}\left(\bar{C}(\delta p - \delta r)\right) + n_p$$

Recognizing that $\bar{s}_p = \bar{C}(\bar{p} - \bar{r}) + n_p$ this simplifies to

$$\delta s_p = \bar{C}(\delta p - \delta r) - \delta\phi^{\times}\left(\bar{C}(\bar{p} - \bar{r})\right) - \delta\phi^{\times}\left(\bar{C}(\delta p - \delta r)\right)$$

The last term above is the cross product of state deviations and is thus neglected. It follows that, after using the fact that $a^{\times}b = -b^{\times}a$, we have

$$\delta s_p = -\bar{C}\delta r + \bar{C}\delta p + \left(\bar{C}(\bar{p} - \bar{r})\right)^{\times}\delta\phi$$

Relative Foot Quaternion

The relative foot quaternion measurement is given by (for a single foot)

$$s_z = n_z \otimes q \otimes z^{-1}$$

Letting $s_z \approx \delta s_z \otimes \bar{s}_z$, $q \approx \delta q \otimes \bar{q}$ and $z \approx \delta z \otimes \bar{z}$ we have

$$\delta s_z \otimes \bar{s}_z = s_z = n_z \otimes (\delta q \otimes \bar{q}) \otimes (\delta z \otimes \bar{z})^{-1}$$

After expanding the right hand side and regrouping, we have

$$\delta s_z \otimes \bar{s}_z = n_z \otimes \delta q \otimes (\bar{q} \otimes \bar{z}^{-1}) \otimes \delta z^{-1}$$

Substituting $\bar{s}_z = \bar{q} \otimes \bar{z}^{-1}$ in the right hand side and multiplying on the right of both sides by $\bar{s}_z^{-1}$ yields

$$\delta s_z = n_z \otimes \delta q \otimes \left(\bar{s}_z \otimes \delta z^{-1} \otimes \bar{s}_z^{-1}\right)$$

In (Trawny and Roumeliotis, 2005) was shown that a triple product of quaternions can be written as

$$\left(q \otimes p \otimes q^{-1}\right) = \begin{pmatrix} C[q]p_v \\ p_s \end{pmatrix}$$

Since δz^{-1} is the quaternion corresponding to the small rotation $-\delta\theta$, we have the approximation

$$\delta z^{-1} \approx \begin{pmatrix} -\frac{1}{2}\delta\theta \\ 1 \end{pmatrix}$$

and thus

$$\left(\bar{s}_z \otimes \delta z^{-1} \otimes \bar{s}_z^{-1}\right) \approx \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta \\ 1 \end{pmatrix}$$

Assuming that the rotations corresponding to δs_z and n_z are small and rewriting the quaternion products as matrix-vector products yields

$$\begin{pmatrix} \frac{1}{2}\delta s_z \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2}n_z \\ 1 \end{pmatrix} \otimes \left[\begin{pmatrix} I - \frac{1}{2}\delta\phi^{\times} & \frac{1}{2}\delta\phi \\ -\frac{1}{2}\delta\phi^T & 1 \end{pmatrix} \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta \\ 1 \end{pmatrix} \right]$$

$$= \begin{pmatrix} \frac{1}{2}n_z \\ 1 \end{pmatrix} \otimes \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta + \frac{1}{4}\delta\phi^{\times}C[\bar{s}_z]\delta\theta + \frac{1}{2}\delta\phi \\ \frac{1}{4}\delta\phi^T C[\bar{s}_z]\delta\theta + 1 \end{pmatrix}$$

$$= \begin{pmatrix} I - \frac{1}{2}n_z^{\times} & \frac{1}{2}n_z \\ -\frac{1}{2}n_z^T & 1 \end{pmatrix} \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta + \frac{1}{4}\delta\phi^{\times}C[\bar{s}_z]\delta\theta + \frac{1}{2}\delta\phi \\ \frac{1}{4}\delta\phi^T C[\bar{s}_z]\delta\theta + 1 \end{pmatrix}$$

where δs_z is the vector corresponding to the measurement quaternion deviation, n_z is the measurement noise vector and $\delta\phi$ is the vector corresponding to the deviation in the base

pose. The expression on the right can be simplified by eliminating terms involving products of small quantities. This yields

$$\begin{pmatrix} \frac{1}{2}\delta s_z \\ 1 \end{pmatrix} = \begin{pmatrix} I - \frac{1}{2}n_z^{\times} & \frac{1}{2}n_z \\ -\frac{1}{2}n_z^T & 1 \end{pmatrix} \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta + \frac{1}{2}\delta\phi \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} -\frac{1}{2}C[\bar{s}_z]\delta\theta + \frac{1}{2}\delta\phi + \frac{1}{4}n_z^{\times}C[\bar{s}_z]\delta\theta - \frac{1}{4}n_z^{\times}\delta\phi + \frac{1}{2}n_z \\ \\ \frac{1}{4}n_z^T C[\bar{s}_z]\delta\theta - \frac{1}{4}n_z^T\delta\phi + 1 \end{pmatrix}$$

After again eliminating terms involving products of small quantities and simplifying (using the fact that $C^T[q] = C[q^{-1}]$), the first equation yields the linearized measurement

$$\delta s_z = -C[\bar{q} \otimes \bar{z}^{-1}]\delta\theta + \delta\phi + n_z$$

The second equation becomes $1 = 1$, ensuring consistency.

Appendix C

Observability

A linear system is said to be *observable* if there exists a finite $t_f > t_0$ such that for any initial state $x(t_0)$, knowledge of the input $u(t)$ and the measured output $y(t)$ over the interval $[t_0, t_f]$ is sufficient to determine $x(t)$ on the interval.

C.1 Linear Systems

A linear (in general, time-varying) system is *observable* on the interval $[t_0, t_f]$ if and only if the *observability Grammian*

$$W_o(t_0, t_f) = \int_{t_0}^{t_f} \Phi^T(\tau, t_0) C^T(\tau) C(\tau) \Phi(\tau, t_0) d\tau$$

is nonsingular. However, for a linear, time-invariant (LTI) system it can be shown that the condition for observability is

$$\text{rank } O(A, C) = \text{rank} \begin{bmatrix} C \\ CA \\ CA^2 \\ \cdots \\ C^{n-1}A \end{bmatrix} = n$$

where $O(A, C)$ is called the *observability matrix*. If this condition does not hold, then the unobservable subspace is given by the nullspace of $O(A, C)$.

C.2 Nonlinear Observability

For nonlinear systems, observability isn't as well-defined. Essentially, we want to know how much the output $y = h(x)$ changes with a change in the state $\dot{x} = f(x)$. Equivalently, we want to know the derivative of vector field $h(x)$ along the flow of vector field $f(x)$. We can write:

$$\begin{aligned} y &= h(x) \\ \dot{y} &= \frac{dh(x)}{dx}\frac{dx}{dt} = \mathscr{L}_1(x) \\ \ddot{y} &= \frac{d\mathscr{L}_1(x)}{dx}\frac{dx}{dt} = \mathscr{L}_2(x) \end{aligned}$$

and so on where $\mathscr{L}_i$ is called the i^{th} Lie derivative of $h(x)$. We can then write:

$$\begin{aligned} \frac{\partial}{\partial t}y &= \frac{\partial h}{\partial x}\frac{\partial x}{\partial t} = \nabla h\frac{\partial x}{\partial t} \\ \frac{\partial}{\partial t}\dot{y} &= \frac{\partial \mathscr{L}_1}{\partial x}\frac{\partial x}{\partial t} = \nabla \mathscr{L}_1\frac{\partial x}{\partial t} \\ \frac{\partial}{\partial t}\ddot{y} &= \frac{\partial \mathscr{L}_2}{\partial x}\frac{\partial x}{\partial t} = \nabla \mathscr{L}_2\frac{\partial x}{\partial t} \end{aligned}$$

and so on. Now, "multiply" both sides by ∂t to get

$$\begin{pmatrix} \partial y \\ \partial \dot{y} \\ \partial \ddot{y} \\ \vdots \end{pmatrix} = \begin{pmatrix} \nabla h \\ \nabla \mathscr{L}_1 \\ \nabla \mathscr{L}_2 \\ \vdots \end{pmatrix} \partial x$$

where we define the nonlinear observability matrix to be

$$\mathscr{O} = \begin{pmatrix} \nabla h \\ \nabla \mathscr{L}_1 \\ \nabla \mathscr{L}_2 \\ \vdots \end{pmatrix}$$

So $\mathscr{O}$ relates a perturbation in the state to changes in the output and its derivatives.

We investigate the observability of an estimator having nonlinear prediction and/or measurement models by forming the nonlinear observability matrix introduced in (Hermann and Krener, 1977). As in the linear case, the state is observable if O has full rank; unobservable state combinations are parameterized by the nullspace of O. Given a system with nonlinear process model $\dot{x} = f(x)$ and measurement model $y = h(x)$,

$$O = \begin{bmatrix} \nabla h(x) \\ \nabla(\nabla h \bullet f(x)) \\ \nabla(\nabla(\nabla h \bullet f(x)) \bullet f(x)) \\ \vdots \end{bmatrix}$$

where ∇ denotes the Jacobian with respect to x. In the linear case, $\nabla h = C$ and $f(x) = Ax$ so $\nabla(\nabla h \bullet f(x)) = \nabla(CAx) = CA$, $\nabla(\nabla(\nabla h \bullet f(x)) \bullet f(x)) = CA^2$ and so on. Unlike in the linear case, there is no condition limiting the size of O; however, only a finite number of derivatives usually need be taken before successive rows become zero (and thus no longer affect the rank). Essentially, O illustrates that states are observable because they are measured directly or because they appear in the dynamics (of some order) of a state which is measured directly. Note that since O is state-dependent in general, this procedure investigates *local* observability.

Appendix D

Inverse Dynamics Details

For the purposes of implementation, we must write the inverse dynamics quadratic program of Section II.3 in the form

$$\min_x x^T H x + 2g^T x$$

subject to constraints

$$A_{eq}x + b_{eq} = 0$$
$$A_{ineq}x + b_{ineq} \leq 0$$

Expanding the cost function, we have

$$\begin{aligned}
&(\ddot{x} - \ddot{x}_{des})^T W_x (\ddot{x} - \ddot{x}_{des}) + (T_{null}\ddot{q} - \ddot{q}_{des})^T W_q (T_{null}\ddot{q} - \ddot{q}_{des}) + \cdots \\
&\qquad + (\lambda - \lambda_{des})^T W_\lambda (\lambda - \lambda_{des}) + (\tau - \tau_{des})^T W_\tau (\tau - \tau_{des}) \\
&= \left[\ddot{x}^T W_x \ddot{x} - 2\ddot{x}_{des}^T W_x \ddot{x} + \ddot{x}_{des}^T W_x \ddot{x}_{des}\right] + \left[\ddot{q}^T T_{null}^T W_q T_{null}\ddot{q} - 2\ddot{q}_{des}^T W_x T_{null}\ddot{q} + \ddot{q}_{des}^T W_q \ddot{q}_{des}\right] + \cdots \\
&\qquad + \lambda^T W_\lambda \lambda^T - 2\lambda_{des}^T W_\lambda \lambda + \lambda_{des} W_\lambda \lambda_{des} + \cdots \\
&\qquad + \tau^T W_\tau \tau^T - 2\tau_{des}^T W_\tau \tau + \tau_{des} W_\tau \tau_{des}
\end{aligned}$$

However, we must use the fact that $\ddot{x} = J_u\ddot{q} + \dot{J}_u\dot{q}$; focusing for now on just the first term above,

$$\begin{aligned}
&\ddot{x}^T W_x \ddot{x} - 2\ddot{x}_{des}^T W_x \ddot{x} + \ddot{x}_{des}^T W_x \ddot{x}_{des} \\
&= (J_u\ddot{q} + \dot{J}_u\dot{q})^T W_x (J_u\ddot{q} + \dot{J}_u\dot{q}) - 2\ddot{x}_{des}^T W_x (J_u\ddot{q} + \dot{J}_u\dot{q}) + \ddot{x}_{des}^T W_x \ddot{x}_{des} \\
&= \ddot{q}^T J_u^T W_x J_u \ddot{q} + 2\dot{q}^T \dot{J}_u^T W_x J_u \ddot{q} + \dot{q}^T \dot{J}_u^T W_x \dot{J}_u \dot{q} - 2\ddot{x}_{des}^T W_x J_u \ddot{q} - 2\ddot{x}_{des}^T W_x \dot{J}_u \dot{q} + \ddot{x}_{des}^T W_x \ddot{x}_{des}
\end{aligned}$$

Any terms which do not depend on $\ddot{q}$, λ or τ drop out of the cost, so we finally have

$$\begin{aligned}
\left[\ddot{q}^T J_u^T W_x J_u \ddot{q} + 2\dot{q}^T \dot{J}_u^T W_x J_u \ddot{q} - 2\ddot{x}_{des}^T W_x J_u \ddot{q}\right] + \left[\ddot{q}^T T_{null}^T W_q T_{null} \ddot{q} - 2\ddot{q}_{des}^T W_x T_{null} \ddot{q}\right] + \cdots \\
+ \left[\lambda^T W_\lambda \lambda^T - 2\lambda_{des}^T W_\lambda \lambda\right] + \left[\tau^T W_\tau \tau^T - 2\tau_{des}^T W_\tau \tau\right]
\end{aligned}$$

which, after rearranging into the correct form and simplifying using $T_{null}^T = T_{null}$ since this is an orthogonal projection, becomes

$$\begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix}^T \begin{bmatrix} J_u^T W_x J_u + T_{null} W_q T_{null} & 0 & 0 \\ 0 & W_\lambda & 0 \\ 0 & 0 & W_\tau \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix} + 2 \begin{bmatrix} J_u^T W_x (\dot{J}_u \dot{q} - \ddot{x}_{des}) - T_{null} W_q \ddot{q}_{des} \\ -W_\lambda \lambda_{des} \\ -W_\tau \tau_{des} \end{bmatrix}^T \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix}$$

Equality Constraints

There are two sets of equality constraints which must be enforced: dynamics constraints and contact constraints. Recall that the dynamics of the floating base robot are

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T \tau + J_c^T \lambda$$

We can write this equation linearly in terms of the optimization variables as

$$\begin{bmatrix} M(q) & -J_c^T & -S^T \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix} + h(q, \dot{q}) = 0$$

Similarly, the contact constraint is simply

$$J_c \ddot{q} + \dot{J}_c \dot{q} = 0$$

which can be written in the form

$$\begin{bmatrix} J_c & 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix} + (-\dot{J}_c \dot{q}) = 0$$

Inequality Constraints

Torque Limits

Imposing limits on allowable joint torques is accomplished with the constraints

$$\tau \leq \tau_{max}$$
$$\tau \geq \tau_{min} \rightarrow -\tau \leq -\tau_{min}$$

and in matrix form

$$\begin{bmatrix} 0 & 0 & I \\ 0 & 0 & -I \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix} + \begin{bmatrix} -\tau_{max} \\ \tau_{min} \end{bmatrix} \leq 0$$

Acceleration Limits

Unlike actuator limits which can be determined from datasheets, acceleration limits must be chosen or tuned. Since the purpose of these limits is mainly to prevent hitting joint stops, we limit the acceleration to the value that would drive the joint into an endstop in Δt seconds, which can be the actual timestep of the controller or set to something larger (to be more conservative).

$$q + \dot{q}\Delta t + \frac{1}{2}\ddot{q}\Delta t^2 \leq q_{max}$$
$$q + \dot{q}\Delta t + \frac{1}{2}\ddot{q}\Delta t^2 \geq q_{min} \rightarrow -q - \dot{q}\Delta t - \frac{1}{2}\ddot{q}\Delta t^2 \leq -q_{min}$$

which in matrix form is

$$\begin{bmatrix} \frac{1}{2}\Delta t^2 I & 0 & 0 \\ -\frac{1}{2}\Delta t^2 I & 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \\ \tau \end{bmatrix} + \begin{bmatrix} q + \dot{q}\Delta t - q_{max} \\ -q - \dot{q}\Delta t + q_{min} \end{bmatrix} \leq 0$$

Center of Pressure Limits

The center of pressure $p \in R^2$ is given by

$$p_x = -\frac{\lambda_{M,y}}{\lambda_{F,z}}$$
$$p_y = \frac{\lambda_{M,x}}{\lambda_{F,z}}$$

Eliminating Joint Torques

It has been noted that the dynamics of a floating base robotic manipulator can be decomposed into two parts as

$$M_u(q)\ddot{q} + h_u(q,\dot{q}) = \tau + J_{c,u}^T\lambda$$
$$M_l(q)\ddot{q} + h_l(q,\dot{q}) = J_{c,l}^T\lambda$$

where the first set of n equations describes the motion of the joints while the second set of 6 equations describes the motion of the unactuated floating base of the robot. This latter set of equations is also known as the momentum or Newton-Euler dynamics of the system; it relates momentum rate of change of the mechanical system to the external forces applied to the robot at its endeffectors. Because these equations are linear in the variables $\ddot{q}$, λ and τ we can simply use the second set of equations to constrain the optimization to the robot dynamics and then compute the joint torques afterward as

$$\tau = M_u(q)\ddot{q} + h_u(q,\dot{q}) - J_{c,u}^T\lambda$$

The optimization problem thus becomes

$$\min_{[\ddot{q},\lambda]} ||\ddot{x} - \ddot{x}_{des}||_{W_x}^2 + ||T_{null}\ddot{q} - \ddot{q}_{des}||_{W_q}^2 + ||\lambda - \lambda_{des}||_{W_\lambda}^2 + ||(M_u(q)\ddot{q} + h_u(q,\dot{q}) - J_{c,u}^T\lambda) - \tau_{des}||_{W_\tau}^2$$

subject to the constraints

$$M_l(q)\ddot{q} + h_l(q,\dot{q}) = J_{c,l}^T\lambda$$
$$J_c\ddot{q} + \dot{J}_c\dot{q} = 0$$

Because the last term in the above cost now depends on both $\ddot{q}$ and λ after eliminating τ, the hessian and gradient will change accordingly. Expanding just this term (ignoring

dependencies on $q, \dot{q}$ and immediately dropping any terms which do not depend on the optimization variables) yields

$$
\begin{aligned}
&(M_u\ddot{q} + h_u - J_{c,u}^T\lambda - \tau_{des})^T W_\tau (M_u\ddot{q} + h_u - J_{c,u}^T\lambda - \tau_{des}) \\
&= \ddot{q}^T M_u^T W_\tau M_u \ddot{q} + \ddot{q}^T M_u^T W_\tau h_u - \ddot{q}^T M_u^T W_\tau J_{c,u}^T \lambda - \ddot{q}^T M_u^T W_\tau \tau_{des} + \cdots \\
&+ h_u^T W_\tau M_u \ddot{q} - h_u^T W_\tau J_{c,u}^T \lambda + \cdots \\
&- \lambda^T J_{c,u} W_\tau M_u \ddot{q} - \lambda^T J_{c,u} W_\tau h_u + \lambda^T J_{c,u} W_\tau J_{c,u}^T \lambda + \lambda^T J_{c,u} W_\tau \tau_{des} + \cdots \\
&- \tau_{des}^T W_\tau M_u \ddot{q} + \tau_{des}^T W_\tau J_{c,u}^T \lambda
\end{aligned}
$$

which can be put into matrix form as

$$
\begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix}^T \begin{bmatrix} J_u^T W_x J_u + T_{null} W_q T_{null} + M_u^T W_\tau M_u & -M_u^T W_\tau J_{c,u}^T \\ -J_{c,u} W_\tau M_u & W_\lambda + J_{c,u} W_\tau J_{c,u}^T \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix}
$$
$$
+ 2 \begin{bmatrix} J_u^T W_x (\dot{J}_u \dot{q} - \ddot{x}_{des}) - T_{null} W_q \ddot{q}_{des} + M_u^T W_\tau (h_u - \tau_{des}) \\ -W_\lambda \lambda_{des} + J_{c,u} W_\tau (\tau_{des} - h_u) \end{bmatrix}^T \begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix}
$$

Bibliography

Satnam Alag, Alice M Agogino, and Mahesh Morjaria. A methodology for intelligent sensor measurement, validation, fusion, and fault detection for equipment monitoring and diagnostics. *AI EDAM*, 15(04):307–320, 2001.

P. b. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 137–142, Dec 2006. doi: 10.1109/ICHR.2006.321375.

V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. R. De Pieri, and D. G. Caldwell. A reactive controller framework for quadrupedal locomotion on challenging terrain. In *2013 IEEE International Conference on Robotics and Automation*, pages 2554–2561, May 2013. doi: 10.1109/ICRA.2013.6630926.

M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart. State estimation for legged robots on unstable and slippery terrain. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6058–6064, Nov 2013. doi: 10.1109/IROS.2013.6697236.

Michael Bloesch, Marco Hutter, Mark Hoepflinger, Stefan Leutenegger, Christian Gehring, C. David Remy, and Roland Siegwart. State estimation for legged robots - consistent fusion of leg kinematics and IMU. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

T. Boaventura, C. Semini, J. Buchli, M. Frigerio, M. Focchi, and D. G. Caldwell. Dynamic torque control of a hydraulic quadruped robot. In *Robotics and Automation (ICRA),*

2012 IEEE International Conference on, pages 1889–1894, May 2012. doi: 10.1109/ICRA.2012.6224628.

W. G. Breckenridge. Quaternions proposed standard conventions. Technical report, NASA Jet Propulsion Laboratory, October 1979.

M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini. Probabilistic contact estimation and impact detection for state estimation of quadruped robots. *IEEE Robotics and Automation Letters*, 2(2):1023–1030, April 2017. ISSN 2377-3766. doi: 10.1109/LRA.2017.2652491.

J. Carpentier, M. Benallegue, N. Mansard, and J. P. Laumond. Center-of-mass estimation for a polyarticulated system in contact - a spectral approach. *IEEE Transactions on Robotics*, 32(4):810–822, Aug 2016. ISSN 1552-3098. doi: 10.1109/TRO.2016.2572680.

P. Cheng and B. Oelmann. Joint-angle measurement using accelerometers and gyroscopesa survey. *Instrumentation and Measurement, IEEE Transactions on*, 59(2), Oct 2010. ISSN 0018-9456. doi: 10.1109/TIM.2009.2024367.

Stefano Chiaverini, Bruno Siciliano, and Luigi Villani. A survey of robot interaction control schemes with experimental comparison. *IEEE/ASME Transactions on mechatronics*, 4 (3):273–285, 1999.

A. Chilian, H. Hirschmuller, and M. Gorner. Multisensor data fusion for robust pose estimation of a six-legged walking robot. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2497–2504, Sept 2011. doi: 10.1109/IROS.2011.6094484.

S. Chitta, P. Vernaza, R. Geykhman, and D.D. Lee. Proprioceptive localization for a quadrupedal robot on known terrain. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4582–4587, April 2007. doi: 10.1109/ROBOT.2007.364185.

J. A. Cobano, J. Estremera, and P. Gonzalez de Santos. Location of legged robots in outdoor environments. *Robot. Auton. Syst.*, 56(9):751–761, September 2008. ISSN 0921-8890. doi: 10.1016/j.robot.2007.12.003. URL http://dx.doi.org/10.1016/j.robot.2007.12.003.

Mathew DeDonato, Velin Dimitrov, Ruixiang Du, Ryan Giovacchini, Kevin Knoedler, Xianchao Long, Felipe Polido, Michael A. Gennert, Takn Padr, Siyuan Feng, Hirotaka Moriguchi, Eric Whitman, X. Xinjilefu, and Christopher G. Atkeson. Human-in-the-loop control of a humanoid robot for disaster response: A report from the darpa robotics challenge trials. *Journal of Field Robotics*, 32(2):275–292, 2015. ISSN 1556-4967. doi: 10.1002/rob.21567. URL http://dx.doi.org/10.1002/rob.21567.

J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4(1):95–104, 1974. doi: 10.1080/01969727408546059. URL http://dx.doi.org/10.1080/01969727408546059.

M.A. El-Gohary. *Joint Angle Tracking with Inertial Sensors*. PhD thesis, Portland State University, 2013.

J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schffer. Overview of the torque-controlled humanoid robot toro. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 916–923, Nov 2014. doi: 10.1109/HUMANOIDS.2014.7041473.

M. F. Fallon, M. Antone, N. Roy, and S. Teller. Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 112–119, Nov 2014. doi: 10.1109/HUMANOIDS.2014.7041346.

Maurice Fallon, Scott Kuindersma, Sisir Karumanchi, Matthew Antone, Toby Schneider, Hongkai Dai, Claudia Prez D'Arpino, Robin Deits, Matt DiCicco, Dehann Fourie, Twan Koolen, Pat Marion, Michael Posa, Andrs Valenzuela, Kuan-Ting Yu, Julie Shah, Karl Iagnemma, Russ Tedrake, and Seth Teller. An architecture for online affordance-based perception and whole-body planning. *Journal of Field Robotics*, 32(2):229–254, 2015.

ISSN 1556-4967. doi: 10.1002/rob.21546. URL http://dx.doi.org/10.1002/rob.21546.

S. Faraji, L. Colasanto, and A. J. Ijspeert. Practical considerations in using inverse dynamics on a humanoid robot: Torque tracking, sensor fusion and cartesian control laws. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 1619–1626, Sept 2015. doi: 10.1109/IROS.2015.7353584.

S. Feng. *Online Hierarchical Optimization for Humanoid Control*. PhD thesis, Carnegie Mellon University, 2016.

Siyuan Feng, Eric Whitman, X. Xinjilefu, and Christopher G. Atkeson. Optimization-based full body control for the darpa robotics challenge. *Journal of Field Robotics*, 32 (2):293–312, 2015. ISSN 1556-4967. doi: 10.1002/rob.21559. URL http://dx.doi.org/10.1002/rob.21559.

Michele Focchi, Andrea del Prete, Ioannis Havoutis, Roy Featherstone, Darwin G. Caldwell, and Claudio Semini. High-slope terrain locomotion for torque-controlled quadruped robots. *Autonomous Robots*, 41(1):259–272, Jan 2017. ISSN 1573-7527. doi: 10.1007/s10514-016-9573-1. URL https://doi.org/10.1007/s10514-016-9573-1.

Bernard Friedland. *Advanced Control System Design*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995. ISBN 0130140104.

B. Gassmann, F. Zacharias, J.M. Zollner, and R. Dillmann. Localization of walking robots. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 1471–1476, April 2005. doi: 10.1109/ROBOT.2005.1570322.

Philippe Gerum. Xenomai-implementing a rtos emulation framework on gnu/linux. Technical report, Xenomai Project, 2004.

Ambarish Goswami. Postural stability of biped robots and the foot-rotation indicator (fri) point. *The International Journal of Robotics Research*, 18(6):523–533, 1999.

doi: 10.1177/02783649922066376. URL http://ijr.sagepub.com/content/18/6/523.abstract.

I. Hashlamon and K. Erbatur. Center of mass states and disturbance estimation for a walking biped. In *Mechatronics (ICM), 2013 IEEE International Conference on*, pages 248–253, Feb 2013. doi: 10.1109/ICMECH.2013.6518544.

A. Herdt, N. Perrin, and P. B. Wieber. Walking without thinking about it. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 190–195, Oct 2010. doi: 10.1109/IROS.2010.5654429.

R. Hermann and Arthur J. Krener. Nonlinear controllability and observability. *Automatic Control, IEEE Transactions on*, 22(5):728–740, Oct 1977. ISSN 0018-9286. doi: 10.1109/TAC.1977.1101601.

A. Herzog, L. Righetti, F. Grimminger, P. Pastor, and S. Schaal. Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pages 981–988, September 2014a. doi: 10.1109/IROS.2014.6942678.

A. Herzog, N. Rotella, S. Schaal, and L. Righetti. Trajectory generation for multi-contact momentum control. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 874–880, Nov 2015. doi: 10.1109/HUMANOIDS.2015.7363464.

Alexander Herzog, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti. Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid. *CoRR*, abs/1410.7284, 2014b.

M. A. Hoepflinger, M. Hutter, C. Gehring, M. Bloesch, and R. Siegwart. Unsupervised identification and prediction of foothold robustness. In *2013 IEEE International Conference on Robotics and Automation*, pages 3293–3298, May 2013. doi: 10.1109/ICRA.2013.6631036.

Neville Hogan. Impedance control: An approach to manipulation: Part itheory. *Journal of Dynamic Systems, Measurement, and Control*, 107, 1985. ISSN 0022-0434. doi: 10.1115/1.3140702. URL http://dx.doi.org/10.1115/1.3140702.

J. Honkakorpi, J. Vihonen, and J. Mattila. Mems-based state feedback control of multi-body hydraulic manipulator. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4419–4425, Nov 2013. doi: 10.1109/IROS.2013.6696991.

M. Hutter, M. Hoepflinger, C. Gehring, M. Bloesch, C. Remy, and R. Siegwart. Hybrid operational space control for compliant legged systems. In *Proc. of the 8th Robotics: Science and Systems Conference (RSS), 2012*, 2012a. doi: 10.1109/IROS.2010.5648837.

Marco Hutter, Christian Gehring, Michael Bloesch, Mark A Hoepflinger, C David Remy, and Roland Siegwart. Starleth: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In *15th International Conference on Climbing and Walking Robot-CLAWAR 2012*, 2012b.

J. Hwangbo, C. D. Bellicoso, P. Fankhauser, and M. Huttery. Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3872–3878, Oct 2016. doi: 10.1109/IROS.2016.7759570.

Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Tingfan Wu, Daniel Duran, Marshall Floyd, Peter Abeles, Douglas Stephen, Nathan Mertins, Alex Lesman, John Carff, William Rifenburgh, Pushyami Kaveti, Wessel Straatman, Jesper Smith, Maarten Griffioen, Brooke Layton, Tomas de Boer, Twan Koolen, Peter Neuhaus, and Jerry Pratt. Team ihmc's lessons learned from the darpa robotics challenge trials. *Journal of Field Robotics*, 32(2):192–208, 2015. ISSN 1556-4967. doi: 10.1002/rob.21571. URL http://dx.doi.org/10.1002/rob.21571.

W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, Sep 1976. doi: 10.1107/S0567739476001873. URL `http://dx.doi.org/10.1107/S0567739476001873`.

S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1620–1626 vol.2, Sept 2003. doi: 10.1109/ROBOT.2003.1241826.

Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 239–246. IEEE, 2001.

Svetoslav Kolev and Emanuel Todorov. Physically consistent state estimation and system identification for contacts. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 1036–1043. IEEE, 2015.

Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016. ISSN 1573-7527. doi: 10.1007/s10514-015-9479-3. URL `http://dx.doi.org/10.1007/s10514-015-9479-3`.

SangJoo Kwon and Yonghwan Oh. Estimation of the center of mass of humanoid robot. In *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, pages 2705–2709, Oct 2007. doi: 10.1109/ICCAS.2007.4406826.

Sung-Hee Lee and Ambarish Goswami. A momentum-based balance controller for humanoid robots on non-level and non-stationary ground. *Autonomous Robots*, 33(4): 399–414, 2012. ISSN 0929-5593. doi: 10.1007/s10514-012-9294-z.

P.-C. Lin, H. Komsuoglu, and D.E. Koditschek. Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits. *Robotics, IEEE Transactions on*, 22(5): 932–943, Oct 2006. ISSN 1552-3098. doi: 10.1109/TRO.2006.878954.

Pei-Chun Lin, Jau-Ching Lu, Chia-Hung Tsai, and Chi-Wei Ho. Design and implementation of a nine-axis inertial measurement unit. *Mechatronics, IEEE/ASME Transactions on*, 17(4):657–668, Aug 2012. ISSN 1083-4435. doi: 10.1109/TMECH.2011.2111378.

Y. King Liu. Discussion: measurement of angular acceleration of a rigid body using linear accelerometers. *ASME Journal of Applied Mechanics*, 43(2):377–378, 1976. doi: 10.1115/1.3423861. URL `http://dx.doi.org/10.1115/1.3423861`.

Kendall Lowrey, Svetoslav Kolev, Yuval Tassa, Tom Erez, and Emanuel Todorov. Physically-consistent sensor fusion in contact-rich behaviors. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1656–1662. IEEE, 2014.

L. Manuelli and R. Tedrake. Localizing external contact using proprioceptive sensors: The contact particle filter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5062–5069, Oct 2016. doi: 10.1109/IROS.2016.7759743.

S. Mason, N. Rotella, S. Schaal, and L. Righetti. Balancing and walking using full dynamics lqr control with constraints. In *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on (ACCEPTED)*, 2016.

K. Masuya and T. Sugihara. Dead reckoning of biped robots with estimated contact points based on the minimum velocity criterion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3637–3642, Nov 2013. doi: 10.1109/IROS.2013.6696875.

M. Mistry, S. Schaal, and K. Yamane. Inertial parameter estimation of floating base humanoid systems using partial force sensing. In *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pages 492–497, Dec 2009. doi: 10.1109/ICHR.2009.5379531.

M. Mistry, J. Buchli, and S. Schaal. Inverse dynamics control of floating base systems using orthogonal decomposition. In *2010 IEEE International Conference on Robotics and Automation*, pages 3406–3412, May 2010. doi: 10.1109/ROBOT.2010.5509646.

Yoshihiko Nakamura, Hideo Hanafusa, and Tsuneo Yoshikawa. Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, 6(2):3–15, 1987. doi: 10.1177/027836498700600201. URL `http://ijr.sagepub.com/content/6/2/3.abstract`.

Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008. doi: 10.1177/0278364908091463. URL `http://ijr.sagepub.com/content/27/6/737.abstract`.

S. Nozawa, S. Noda, M. Murooka, K. Okada, and M. Inaba. Online estimation of object-environment constraints for planning of humanoid motion on a movable object. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1291–1298, May 2017. doi: 10.1109/ICRA.2017.7989152.

V. Ortenzi, H. C. Lin, M. Azad, R. Stolkin, J. A. Kuo, and M. Mistry. Kinematics-based estimation of contact constraints using only proprioception. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1304–1311, Nov 2016. doi: 10.1109/HUMANOIDS.2016.7803438.

A.J. Padgaokkar, K. W. Krieger, and A. I. King. Measurement of angular acceleration of a rigid body using linear accelerometers. *ASME Journal of Applied Mechanics*, 42(3): 552–556, 1975. doi: 10.1115/1.3423640. URL `http://dx.doi.org/10.1115/1.3423640`.

Sangbum Park, Youngjoon Han, and Hernsoo Hahn. Balance control of a biped robot using camera image of reference object. *International Journal of Control, Automation and Systems*, 7(1):75–84, 2009. ISSN 1598-6446. doi: 10.1007/s12555-009-0110-2. URL `http://dx.doi.org/10.1007/s12555-009-0110-2`.

P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal. Online movement adaptation based on previous sensor experiences. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 365–371, 2011.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

A. Petrovskaya, J. Park, and O. Khatib. Probabilistic estimation of whole body contacts for multi-contact robot control. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 568–573, April 2007. doi: 10.1109/ROBOT.2007.363047.

A. Del Prete, F. Nori, G. Metta, and L. Natale. Control of contact forces: The role of tactile feedback for contact localization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4048–4053, Oct 2012. doi: 10.1109/IROS.2012.6385803.

Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

M. Reinstein and M. Hoffmann. Dead reckoning in a dynamic quadruped robot: Inertial navigation system aided by a legged odometer. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 617–624, May 2011. doi: 10.1109/ICRA.2011.5979609.

C. Ridgewell. *Humanoid Robot Friction Estimation in Multi-Contact Scenarios*. PhD thesis, Virginia Polytechnic Institute and State University, 2017.

Ludovic Righetti, Jonas Buchli, Michael Mistry, Mrinal Kalakrishnan, and Stefan Schaal. Optimal distribution of contact forces with inverse-dynamics control. *The International*

Journal of Robotics Research, 32(3):280–298, March 2013. ISSN 0278-3649, 1741-3176. doi: 10.1177/0278364912469821. URL http://ijr.sagepub.com/content/32/3/280.

G. P. Roston and Eric Krotkov. Dead reckoning navigation for walking robots. Technical Report CMU-RI-TR-91-27, Robotics Institute, Pittsburgh, PA, November 1991.

N. Rotella, M. Bloesch, L. Righetti, and S. Schaal. State estimation for a humanoid robot. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 952–958, Sept 2014. doi: 10.1109/IROS.2014.6942674.

N. Rotella, S. Mason, S. Schaal, and L. Righetti. Inertial sensor-based humanoid joint state estimation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1825–1831, May 2016. doi: 10.1109/ICRA.2016.7487328.

N. Rotella, S. Schaal, and L. Righetti. Unsupervised contact learning for humanoid estimation and control. In *Robotics and Automation (ICRA), 2018 IEEE International Conference on (SUBMITTED)*, 2017.

G. Santaera, E. Luberto, A. Serio, M. Gabiccini, and A. Bicchi. Low-cost, fast and accurate reconstruction of robotic and human postures via imu measurements. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2728–2735, May 2015. doi: 10.1109/ICRA.2015.7139569.

Stefan Schaal. The sl simulation and real-time control software package. Technical report, University of Southern California, Dept. of Comp. Sci., June 2007.

Alfred R. Schuler, Anthony Grammatikos, and Kenneth A. Fegley. Measuring rotational motion with linear accelerometers. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-3(3):465–472, May 1967. ISSN 0018-9251. doi: 10.1109/TAES.1967.5408811.

Thomas Seel, Jrg Raisch, and Thomas Schauer. Imu-based joint angle measurement for gait analysis. *Sensors*, 14(4):6891, 2014. ISSN 1424-8220. doi: 10.3390/s140406891. URL http://www.mdpi.com/1424-8220/14/4/6891.

Claudio Semini. Hyqdesign and development of a hydraulically actuated quadruped robot. *Doctor of Philosophy (Ph. D.), University of Genoa, Italy*, 2010.

M. D. Shuster. Survey of attitude representations. *Journal of the Astronautical Sciences*, 41:439–517, October 1993.

Bruno Siciliano, Lorenzo Sciavicco, and Luigi Villani. *Robotics : modelling, planning and control*. Advanced Textbooks in Control and Signal Processing. Springer, London, 2009. ISBN 1-8462-8641-7. URL `http://opac.inria.fr/record=b1129198`. 013-81159.

Joan Sola. Quaternion kinematics for the error-state KF. Technical report, Universitat Politcnica de Catalunya, February 2014.

B.J. Stephens. State estimation for force-controlled humanoid balance using simple models in the presence of modeling error. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3994–3999, May 2011. doi: 10.1109/ICRA.2011.5980358.

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*, Oct 2010. doi: 10.1109/IROS.2010.5648837.

Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913, Oct 2012. doi: 10.1109/IROS.2012.6386025.

Nikolas Trawny and Stergios I. Roumeliotis. Indirect Kalman filter for 3D attitude estimation. Technical Report 2005-002, University of Minnesota, Dept. of Comp. Sci. & Eng., March 2005.

Nikolaos G Tsagarakis, Giorgio Metta, Giulio Sandini, David Vernon, Ricardo Beira, Francesco Becchi, Ludovic Righetti, Jose Santos-Victor, Auke Jan Ijspeert, Maria Chiara

Carrozza, et al. icub: the design and realization of an open humanoid platform for cognitive and neuroscience research. *Advanced Robotics*, 21(10):1151–1175, 2007.

J. Vihonen, J. Honkakorpi, J. Mattila, and A. Visa. Geometry-aided mems motion state estimation for multi-body manipulators. In *Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on*, pages 341–347, July 2013. doi: 10.1109/AIM.2013.6584115.

J. Vihonen, J. Honkakorpi, J. Koivumaki, J. Mattila, and A. Visa. Geometry-aided low-noise angular velocity sensing of rigid-body manipulator using mems rate gyros and linear accelerometers. In *Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference on*, pages 570–575, July 2014. doi: 10.1109/AIM.2014.6878139.

Liyang Wang, Zhi Liu, C.L.P. Chen, Yun Zhang, Sukhan Lee, and Xin Chen. A ukf-based predictable svr learning controller for biped walking. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, 43(6):1440–1450, Nov 2013. ISSN 2168-2216. doi: 10.1109/TSMC.2013.2242887.

Patrick M Wensing and David 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. IEEE, 2013.

G. Wiedebach, S. Bertrand, T. Wu, L. Fiorio, S. McCrory, R. Griffin, F. Nori, and J. Pratt. Walking on partial footholds including line contacts with the humanoid robot atlas. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1312–1319, Nov 2016. doi: 10.1109/HUMANOIDS.2016.7803439.

R. Williamson and B.J. Andrews. Detecting absolute human knee angle and angular velocity using accelerometers and rate gyroscopes. *Medical and Biological Engineering and Computing*, 39(3):294–302, 2001. ISSN 0140-0118. doi: 10.1007/BF02345283. URL `http://dx.doi.org/10.1007/BF02345283`.

Oliver J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, August 2007. URL `http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf`.

Xinjilefu and C.G. Atkeson. State estimation of a walking humanoid robot. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3693–3699, Oct 2012. doi: 10.1109/IROS.2012.6386070.

X. Xinjilefu, S. Feng, and C. G. Atkeson. Dynamic state estimation using quadratic programming. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 989–994, Sept 2014a. doi: 10.1109/IROS.2014.6942679.

X. Xinjilefu, S. Feng, W. Huang, and C. G. Atkeson. Decoupled state estimation for humanoids using full-body dynamics. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 195–201, May 2014b. doi: 10.1109/ICRA.2014.6906609.

X Xinjilefu, Siyuan Feng, and Christopher G Atkeson. Center of mass estimator for humanoids and its application in modelling error compensation, fall detection and prevention. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 67–73. IEEE, 2015.

X. Xinjilefu, S. Feng, and C. G. Atkeson. A distributed mems gyro network for joint velocity estimation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1879–1884, May 2016. doi: 10.1109/ICRA.2016.7487334.

K. Yamane. Practical kinematic and dynamic calibration methods for force-controlled humanoid robots. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 269–275, Oct 2011. doi: 10.1109/Humanoids.2011.6100803.

B. Zappa, G. Legnani, and A. J. van den Bogert. On the number and placement of accelerometers for angular velocity and acceleration determination. *Journal of Dynamic Systems, Measurement, and Control*, 123:552–554, 2001. doi: 10.1115/1.1386649.