Balance control and locomotion planning for humanoid robots using nonlinear centroidal models

by

Frans Anton Koolen

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2020

Author .....
Department of Electrical Engineering and Computer Science
September 30, 2019

Certified by .....
Russ Tedrake
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Balance control and locomotion planning for humanoid robots using nonlinear centroidal models

by

Frans Anton Koolen

Submitted to the Department of Electrical Engineering and Computer Science
on September 30, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Balance control approaches for humanoid robots have traditionally relied on low-dimensional models for locomotion planning and reactive balance control. Results for the low-dimensional model are mapped to the full robot, and used as inputs to a whole-body controller. In particular, the linear inverted pendulum (LIP) has long been the de facto standard low-dimensional model used in balance control. The LIP has its limitations, however. For example, it requires that the robot's center of mass move on a plane and that the robot's contact environment be planar. This thesis presents control and planning approaches using nonlinear low-dimensional models, aimed at mitigating some of the limitations of the LIP. Specifically, the contributions are: 1) a closed-form controller and region of attraction analysis for a nonlinear variable-height inverted pendulum model, 2) a trajectory optimization approach for humanoid robot locomotion over moderately complex terrain based on mixed-integer nonlinear programming with a low-dimensional model, and 3) a quadratic-programming based controller that uses the the results from these low-dimensional models to control a simulation model of the Atlas humanoid robot.

Thesis Supervisor: Russ Tedrake
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

First, I'd like to thank Russ for giving me the opportunity to be a part of the Robot Locomotion Group, for the freedom, and for the advice throughout the years. Thank you to my committee members, Tomás Lozano-Pérez and Alberto Rodriguez. We only had the opportunity to meet a couple of times, but I'm very thankful for your suggestions.

Thanks to all of my labmates. You've made all the difference in making my time at MIT a good experience. Thanks first and foremost for the great discussions, but also for the lighter conversations, the many lunches we shared, the ping-pong games (which Ani always won), the runs, and the occasional beer. A special thank you to my main collaborators, Robin Deits, Michael Posa, and Sabrina Neuman, and to the members of the DRC team, who made my first couple of years at MIT an intense and exciting experience. Thank you to Steve Proulx, Mieke Moran, and Brytt Bradley. And thanks to Tobia Marcucci for reviewing parts of this thesis, the discussions, and the laughs.

Thank you to Martijn Wisse and the members of the Delft Robotics Laboratory, for lighting the spark by making my introduction to robotics so much fun.

A big thanks to Jerry Pratt, for being a great example and friend, and for helping me throughout my career. Thank you to all my friends and former colleagues at IHMC, for fanning the flame and for all the good times.

Thank you to all my great friends at MIT, especially Affi, Albert, Alin, Ari, Ben, Brett, Eric, Colm, Danielle, Deborah, Jess, Julie, G, Gustavo, Lindsay, Marek, Mark, Mike G., Michel, Sara, and Thomas.

And finally, thank you to my sister and my parents. Pap, mam, thank you again for always being there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Humanoid robots have long captured the imagination of researchers and the general public alike. However, robots with a humanoid form factor have yet to find widespread adoption in the real world, despite decades of research. Classically, good application areas for robots in general have consisted of dirty, dull, and dangerous jobs. While these application areas could also apply to humanoid robots, there is a serious question of whether the humanoid form factor is ever the right choice, perhaps outside of the entertainment industry.

A key argument in favor of the humanoid form factor has always been that it is well-suited to manipulation tasks in a large variety of environments, especially those built for humans. Examples of environments in which humanoid robots could potentially outperform *e.g.* wheeled mobile manipulators include buildings with stairs, very rough outdoor terrain, cluttered areas, and environments with sparse available contact surfaces.

However, the most popular algorithms for balance and locomotion control of humanoid robots have relied on certain key assumptions that are violated by the very environments in which the humanoid form factor is supposed to shine. For example, standard control algorithms typically assume that the ground is flat and level.

In recent years, research focus in the field of humanoid robotics has started to shift away from this simple 'lab' environment, and towards having humanoid robots do useful work in the real world, as witnessed by the DARPA Robotics Challenge

(DRC) and its disaster response scenario [9]. The first requirement that must be met for a humanoid robot to be able to operate in these environments is that it be able to robustly maintain its balance, avoiding costly falls which could damage the robot or its environment. This thesis focuses on balance control approaches that avoid the typical assumptions of classical balance controllers. Specifically, we explore the use of centroidal models where the dynamics are either explicitly nonlinear in the state and control input, or are subject to nonlinear state constraints.

The remainder of this thesis is structured as follows. Chapter 2 establishes some preliminaries and gives a sense of the state of the art in balance control for humanoid robots, including the limiting assumptions that most current control approaches make. Chapter 3 presents a control framework that is used throughout to map results from various centroidal models to the full robot. Chapter 4 presents an analysis of a 2D nonlinear centroidal model that foregoes one of the standard assumptions, namely that the center of mass moves on a plane. It demonstrates how center of mass height variation can instead be used to improve balance. Chapter 5 presents a planner for 3D locomotion over terrain with sparse footholds based on a nonlinear centroidal model. The planning problem is formulated as a mixed-integer nonlinear program.

While an effort has been made to consolidate the mathematical notation used throughout this thesis, notation conventions should generally be considered chapter-specific.

Chapter 2

Preliminaries and related work

This chapter reviews the balance control problem for humanoid robots and provides a literature review of the approaches used to address it.

The chapter is structured as follows. Section 2.1 provides a high-level overview of the balance control problem for legged robots. Section 2.2 discusses some relatively generic control approaches that can be applied to this balance control problem. Section 2.3 briefly recalls properties of a legged robot's dynamics, focusing on the dynamics of its centroidal momentum. How these centroidal dynamics relate to the balance control problem is discussed in section 2.4. Section 2.5 presents perhaps the most popular approach to balance control for humanoids, and section 2.6 discusses limitations of this approach, framing the work presented in this thesis.

2.1 The balance control problem

Perhaps stating the obvious, the primary objective of balance control for legged robots is avoiding falls. Formalizing this statement is tricky, however. In an attempt to do so, we will consider a state-space model of the robot's dynamics of the form[1]

[1]Other formalisms may also be used to model legged robots, most notably hybrid dynamical systems. However, (2.1) suffices for the current discussion.

$$\dot{x}(t) = f(x(t), u(t)) \tag{2.1}$$
$$u(t) \in U(x(t))$$

where $x \in \mathbb{R}^p$ is the robot's state (*e.g.*, joint positions and velocities) and $u \in \mathbb{R}^q$ is its control input (*e.g.*, joint torques), confined to the state-dependent set of admissible control inputs $U(x(t)) \subseteq \mathbb{R}^q$.

To formalize the concept of a 'fall', assume that a subset of state space is designated that consists of failed (fallen) states,[2] $X_{\text{failed}} \subset \mathbb{R}^p$. Then the problem of avoiding this set of failed states can be formalized using the theory of *viability* [10], first applied to the field of legged robotics in [11]. A state x_0 is called viable if and only if there exists at least one admissible input trajectory that results in a state trajectory $x(t)$ starting from x_0 that stays outside of the set of failed states for all times $t \geq 0$. The set of all viable states is called the viability kernel (see Fig. 2-1 for a visualization). The viability kernel is also known as the maximum controlled invariant set [12].

If a legged robot ever finds itself in a state outside of the viability kernel, then it either has already fallen or is doomed to fall, and a fall damage minimization strategy should be employed. For states inside the viability kernel, maintaining balance amounts to selecting any control action that keeps the state inside the viability kernel. As such, under certain continuity conditions of the dynamics (2.1), only states on the boundary of the viability kernel restrict the choice of control actions [10]. In the interior of the viability kernel, there is freedom to choose control actions that achieve a secondary task (*e.g.*, walking to a goal location), or optimize a performance metric (*e.g.*, energy expenditure).

[2]Note that defining such a set may already be a difficult endeavor for a detailed state-space model of the robot.

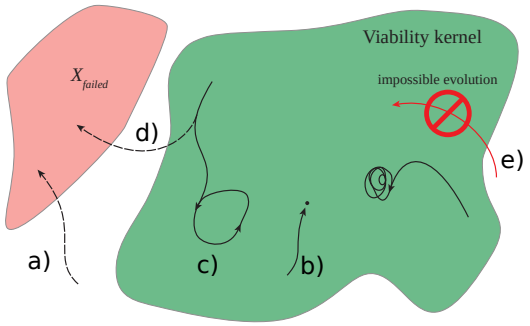


Figure 2-1: Cartoon demonstrating the concept of a viability kernel. Several state evolutions are shown: a) an evolution starting outside the viability kernel inevitably ends up in the set of failed states X_{failed}; b) the system starts in the viability kernel and comes to a rest at a fixed point; c) an evolution that converges to a limit cycle; d) an evolution that has the same initial state as c), but ends up in the set of failed states, *e.g.* because the input trajectory was different; e) impossible evolution: by definition, it is impossible to enter the viability kernel if the initial state is outside the viability kernel. Adapted from [1].

2.2 Generic approaches to balance control and analysis

Unsurprisingly, computing viability kernels and associated control laws for nontrivial systems is intractable. A generally applicable approach to attacking the balance control is to replace viability with a strictly more conservative condition that is easier to work with. For example, for a system with (hybrid) polynomial dynamics and a given polynomial control law $u = k(x)$, sum-of-squares (SOS) optimization [13] may be used to find a Lyapunov-like polynomial barrier function of limited degree whose zero-level set separates the set of failed states from all trajectories starting from a given set of initial states [14]. Doing so certifies that these initial states are a subset of the viability kernel. Other SOS-based computational approaches may be used to search for outer approximations of the viability kernel [15].

Another generic way to somewhat simplify the problem in strictly conservative fashion is to add the constraint that trajectories must reach a pre-specified known-viable state. For example, one may require that trajectories converge to a given final

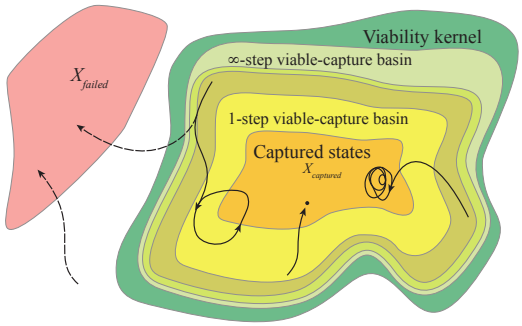


Figure 2-2: Cartoon demonstrating the concept of N-step capturability, formally introduced in [1]. The set of captured states, or the 0-step viable-capture basin, consists of those states from which a static equilibrium is reachable without changing the current contact situation by taking a step. The N-step viable-capture basin may be defined recursively as the set of states from which a viable state trajectory exists that reaches the $(N-1)$-step viable-capture basin in finite time while taking a single step. By definition, each N-step viable-capture basin is a subset of the viability kernel. As $N \to \infty$, the N-step viable-capture basins approach the viable-capture basin. There may be a gap between the viable-capture basin and the viability kernel, for example in the case of a passive mechanism 'walking' down a slope but unable to come to a stop [2, 3, 4]. However, for simple models of walking with a moderate amount of actuation, the argument has been made that the gap between the viability kernel and the 2-step viable-capture basin is small [5]. Adapted from [1].

state x_f that is a fixed point of the dynamics (2.1) for some allowable control input $u_\mathrm{f} \in U(x_\mathrm{f})$. Adding this constraint corresponds to turning the problem of finding the viability kernel into the problem of finding the backwards reachable set of x_f. Again in the context of (hybrid) polynomial dynamics, algorithmic approaches have been developed that search for a polynomial control law that maximizes a measure of the certified region of attraction to a fixed point subject to state and input constraints [16].

Yet another approach is to require that a pre-specified *set* of states be reachable from the initial state in *finite time* without visiting a failed state along the way. The set of initial states that satisfy this property is known as the viable-capture basin (see *e.g.* [17])[3]. In [1], we computed viable-capture basins in closed form for various simplified linear models of legged locomotion. This paper also introduced

[3]The term 'backwards reachable set' may also applied to this concept.

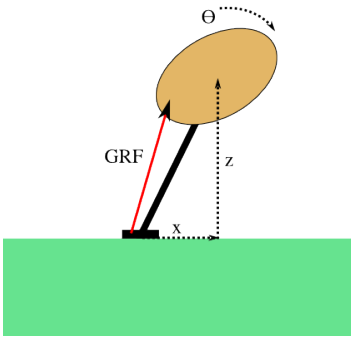


Figure 2-3: One of the simple polynomial models we studied in [6]. The model includes stepping, center of pressure regulation, center of mass height variation, and a lumped inertia which roughly models upper body angular momentum effects.

the notion of N-step capturability, visualized and briefly summarized in Fig. 2-2. Subsequent studies have employed numerical techniques to analyze more complicated models approximately [5, 18, 19].

Although we have applied some of these relatively generic, strictly conservative approaches to simplified polynomial models of walking (see Fig. 2-3) with moderate success [6], none of these methods scale to the level of detailed dynamical models of legged robots. As a result, research in synthesis and analysis of balance controllers has in practice focused on using approximate (non-conservative) criteria for balance maintenance. These criteria are often specific to the particularities of legged robot dynamics. The following section provides a brief overview of these dynamics.

2.3 Dynamics of a legged robot

A legged robot is often modeled as an underactuated, articulated rigid body mechanism with joint configuration vector $q \in \mathbb{R}^{n_q}$, joint velocity vector $v \in \mathbb{R}^{n_v}$, and joint torque vector $\tau \in \mathbb{R}^{n_v}$.[4] Its dynamics are described by the well-known equations of motion [20, 21, 22],

[4]The joint torque vector is collocated with the joint velocity vector, in the sense that $v \cdot \tau$ has the interpretation of mechanical power.

$$M(q)\dot{v} + b(q,v) = \tau + J_c(q)^T f_c \tag{2.2}$$
$$\dot{q} = V(q)\, v$$

where:

- $M(q)$ is the joint-space mass matrix;
- $b(q,v)$ is a bias term consisting of velocity-dependent terms and gravitational effects;
- $f_c = \begin{pmatrix} f_{c,1}^T & f_{c,2}^T & \cdots \end{pmatrix}^T$, with $f_{c,i} \in \mathbb{R}^3$, is a vector of contact forces acting on the robot;
- $J_c(q)$ is the contact Jacobian, and
- $V(q)$ relates the velocity vector to the time derivative of the configuration vector.[5]

The joint torque vector τ is constrained to be of the form

$$\tau = S^T u$$

where $u \in \mathbb{R}^{n_v - 6}$ and $S = \begin{bmatrix} 0_{n_v \times 6} & I_{n_v \times n_v} \end{bmatrix}$ is a selection matrix that characterizes the underactuation.

While the 'full' nonlinear dynamics (2.2) are useful for *e.g.* simulation purposes, most controller design and analysis approaches for humanoid robots have relied on lower-dimensional models. These models are often rooted in the dynamics of the

[5]Many discussions of dynamics in the robotics literature implicitly use $V(q) = I$, so that $v = \dot{q}$. Allowing a more general configuration-dependent linear map between v and $\dot{q}$ has the advantage that redundant parameterizations of configuration (*e.g.*, quaternions for orientation) may be used to avoid artificial singularities and associated numerical problems, while retaining a minimal representation of v.

robot's total centroidal momentum [23],

$$h = \begin{pmatrix} k \\ l \end{pmatrix} \in \mathbb{R}^6 \tag{2.3}$$

where $k \in \mathbb{R}^3$ is the angular momentum about the robot's center of mass and $l \in \mathbb{R}^3$ is the robot's linear momentum, expressed in an inertial frame of reference. Equivalently, the momentum, interpreted as a spatial force vector [21], can be said to be expressed in a *centroidal frame* [24] Φ_c, *i.e.*, an inertial frame with its origin instantaneously at the center of mass position c and its axes aligned with those of a fixed world frame Φ_{w}.

Euler's laws of motion tell us that the sum of the wrenches applied to the robot is equal to the rate of change of momentum, regardless of the complexity of the 'full' robot:

$$\dot{h} = \sum_i w_i \tag{2.4}$$

where each $w_i = \left(\begin{smallmatrix} \tau_i \\ f_i \end{smallmatrix}\right) \in \mathbb{R}^6$ is a wrench consisting of a torque component τ_i and a force component f_i. It should be stressed that only wrenches acting externally on the robot can cause a change in momentum, as wrenches acting between the robot's links cancel in the sum on the right hand side of (2.4) due to Newton's third law.

Not only are the centroidal dynamics (2.4) low-dimensional, they are also *linear* in the external wrenches acting on the robot. Furthermore, linear momentum l is just the product of the robot's total mass m and its center of mass velocity $\dot{c}$,

$$l = m\dot{c} \tag{2.5}$$

and so the dynamics of the center of mass are themselves (second-order) linear,

$$m\ddot{c} = \sum_i f_i.$$

In the absence of external disturbances, the wrenches w_i externally applied to a

humanoid robot can be split up into a wrench due to gravity, w_g, and and ground contact wrenches, $w_{c,i}$:

$$\dot{h} = w_g + \sum_i w_{c,i}, \tag{2.6}$$

where the gravitational wrench may be written in terms of the gravitational acceleration vector $g = \begin{pmatrix} 0 \\ 0 \\ -g_z \end{pmatrix}$ as

$$w_g = \begin{pmatrix} 0 \\ \\ mg \end{pmatrix}.$$

The contact wrenches $w_{c,i} = \begin{pmatrix} \tau_{c,i} \\ f_{c,i} \end{pmatrix} \in \mathbb{R}^6$ are typically modeled as being subject to Coulomb's friction law. In the special case of a point contact in static friction, where the relative velocity between the body-fixed point and the world-fixed environment is initially zero, the contact force must satisfy the friction cone constraint

$$\| f_{c,i} - (n \cdot f_{c,i})\, n \| \leq \mu n \cdot f_{c,i} \tag{2.7}$$

in order to prevent slip. Here, n is the contact normal, and μ is the coefficient of static friction. Expressed in a centroidal frame, the torque component of the contact wrench depends on the relative position of the application point p_i and the center of mass c:[6]

$$\tau_{c,i} = (p_i - c) \times f_{c,i}. \tag{2.8}$$

Centroidal momentum h is also fundamentally linked to the robot's generalized velocity vector, v. It can be shown that this relationship is also linear,[7] i.e. there exists a configuration-dependent matrix $A(q) \in \mathbb{R}^{6\times(n+6)}$, called the centroidal momentum matrix [24, 23], such that

$$h = A(q)\, v. \tag{2.9}$$

[6]Note that in the case of multiple point contacts or distributed loading, the torque may also have a component normal to the contact surface.

[7]This can be seen as follows. Centroidal momentum h is the sum of the momenta of each of the links (expressed in the common centroidal frame [24]), where each link momentum is the product of the link's (velocity-independent) spatial inertia matrix and the twist (spatial velocity) of the link. In turn, the twist is linearly related to the joint velocity vector v through the link's geometric Jacobian [20]. Since each step in this chain of operations acting on the velocity vector v is linear, h is a linear function of v.

Note that as a result of (2.5), the bottom three rows of $A(q)$ are identical to the center of mass Jacobian scaled by m.

Differentiating (2.9) and combining the result with (2.6), we find the relationship

$$A(q)\,\dot{v} + \dot{A}(q)\,v = w_g + \sum_i w_{c,i}, \tag{2.10}$$

which represents a fundamental constraint on the allowable motions of the 'full' robot model as a result of its contact situation, given the friction cone constraints (2.7).

2.4 Centroidal dynamics applied to balance control

The centroidal dynamics (2.6) capture the salient features of the balance control problem for a humanoid robot, namely the gross movement of the robot as a function of gravity and contact forces [25]. They can be used to explain the balance control mechanisms most commonly used in humanoid robots and humans alike [26, 25]:

- taking a recovery step, which corresponds to changing the locations p_i at which the robot can push off against its environment, resulting in a different set of possible contact wrenches;[8]

- the use of ankle torques, which changes the magnitude and perceived point of application of the total contact force applied to a foot (*i.e.*, the foot's center of pressure), and consequently changes the contact wrench applied at the foot;

- windmilling/lunging behaviors, which correspond to a nonzero rate of change of centroidal angular momentum and an associated nonzero total torque about the center of mass. This in turn corresponds to a change of the direction of the total contact force force and associated change in linear momentum.

[8] In addition, taking a step may result in an impact with the ground, which results in an impulsive force that can also be used to achieve balance [25].

Despite the many attractive properties of the centroidal dynamics, there are several complicating factors for their use in control and planning approaches.

First, the constraints on the contact wrenches are nontrivial. As a result of friction cone constraints (2.7), the sum of the contact wrenches at a given time must lie within the so-called contact wrench cone (CWC) [27, 28, 29]. In addition, the contact force $f_{c,i}$ itself and its application point p_i jointly determine the torque component of a contact wrench about the center of mass c, via the relationship (2.8). This relationship is bilinear in contact forces and the positions of their application points,[9] which results in significant issues with optimization-based planning approaches that use both forces and positions as decision variables.

Second, kinematic limitations are not captured very well by a centroidal model. Such limitations may be approximated using constraints on the position of the center of mass relative to the force application points (e.g. in [30, 31]), but cannot fully capture the kinematic workspace of the center of mass. Trajectory optimization approaches that combine the centroidal dynamics with the full kinematics of the robot have been proposed [32], but such approaches are currently fairly slow. In addition, there is no angular equivalent of the center of mass that can be used to summarize the aggregate rotational motion of the robot as a whole for use as a proxy in kinematic constraints.

Third, centroidal models do not capture limitations on joint torques directly. However, this problem has not been as immediate in practice as the preceding two, in part due to the availability of high-performance robots with high-torque actuators, such as the Boston Dynamics Atlas robot [33].

Despite these limitations, centroidal models have been very widely used to inform the design of balance controllers and locomotion planners, leading to a kind of 'standard' control approach.

[9] Specifically, a vector constraint like $c = a \times b$ results in scalar constraints like $c_x = a_y b_z - a_z b_y$, containing products of decision variables.

2.5 A standard balance control approach

This section discusses a commonly used control approach based on the Linear Inverted Pendulum (LIP), a particularly simple centroidal model. Section 2.5.1 briefly describes the model, and Section 2.5.2 discusses how control techniques based on the LIP are typically mapped to the full robot.

2.5.1 The linear inverted pendulum

The (3D) linear inverted pendulum (LIP) [34, 35] (see Fig. 2-4) is based on the centroidal dynamics, (2.3), together with the following assumptions:

1. centroidal angular momentum remains constant (typically zero);[10]

2. the ground is a (typically horizontal) plane;

3. the center of mass moves on a plane parallel to the ground.

These assumptions sidestep the complicating factors described in the previous section to a certain extent.

The first assumption is used to roughly limit angular excursion of the robot's upper body, effectively prohibiting lunging behaviors. Studies have also shown that humans tightly regulate centroidal angular momentum around zero while walking normally [40]. Constant centroidal angular momentum implies that the sum of the torques about the center of mass is zero, in accordance with (2.4) and (2.8).

The second assumption allows for an unambiguous definition of an overall *center of pressure* (CoP), also known as *zero* (tangential) *moment point* (ZMP) p on the ground plane, so that the sum of the contact forces can be thought of as acting at this CoP (see Fig. 2-4). Furthermore, assumptions 1 and 2 together imply that the total contact force acting at the CoP passes through the CoM (see Fig. 2-4). It can be shown that the CoP must always lie within the convex hull of the positions of

[10]Note that there is an extension to the linear inverted pendulum that does allow for a nonzero centroidal angular momentum rate [36, 37, 38, 1] while retaining the same form of the dynamics, (2.16), but with the point p now interpreted as the centroidal moment pivot (CMP) [39].

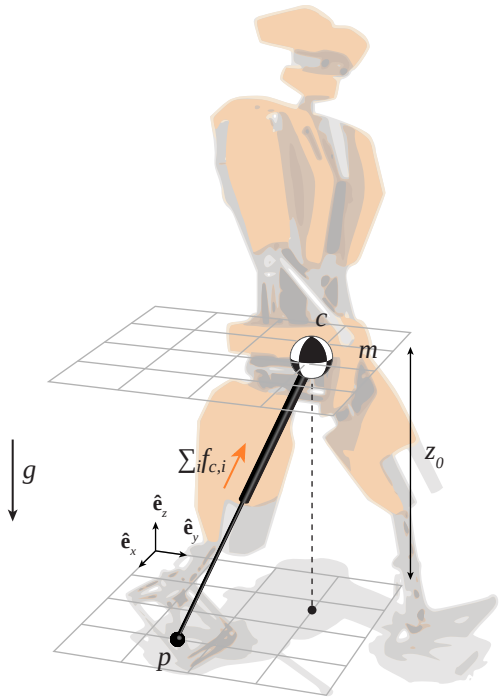


Figure 2-4: The 3D linear inverted pendulum (LIP), with center of pressure p and center of mass c at height z_0 above the ground plane. The total contact force $\sum_i f_{c,i}$ can be thought of as acting at the CoP and passing through the CoM.

the robot's contacts with the flat ground, usually referred to as the robot's support polygon. In the context of optimization-based control techniques, this is a linear constraint that can be seen as a relaxation of the friction cone constraints.

The third assumption makes it so that the dynamics that relate horizontal center of mass motion to *center of pressure location* (as opposed to the contact forces) are second-order linear and decoupled in the x and y directions. When combined with limits on the horizontal motion of the center of mass, the restriction to planar motion also acts as an approximate kinematic limit and as a way to keep the total contact force in check, which in turn acts as a proxy for leg joint torque limits.

We recall from section 2.3 that the rate of change of the robot's total linear momentum can be written as

$$m\ddot{c} = mg + \sum_i f_{c,i} \tag{2.11}$$

The three assumptions imply that the sum of the contact forces can be parameterized

in terms of the center of pressure (CoP) as

$$\sum_i f_{c,i} = \lambda (c - p).$$

In particular, due to the vertical force balance associated with maintaining a constant height, we can write

$$\sum_i f_{c,i} = \frac{m g_z}{z_0} (c - p) \tag{2.12}$$

where g_z is the magnitude of gravitational acceleration and z_0 is the vertical separation between the ground plane and the CoM plane.

Substituting (2.12) into (2.11), using the projection matrix $P = \left(\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right)$ to select the horizontal part and simplifying, we obtain the LIP dynamics

$$P\ddot{c} = \frac{g_z}{z_0} P (c - p) \tag{2.13}$$

or

$$\ddot{c}_{xy} = \frac{g_z}{z_0} (c_{xy} - p_{xy}), \tag{2.14}$$

where $c_{xy} = \left(\begin{smallmatrix} c_x \\ c_y \end{smallmatrix}\right)$, $p_{xy} = \left(\begin{smallmatrix} p_x \\ p_y \end{smallmatrix}\right)$.

The linearity and extremely low dimensionality of the LIP dynamics have often been used as a basis for joint CoM/CoP trajectory planners [41, 42, 43] and feedback control schemes [41], or combined planning and control schemes in the form of model-predictive control (MPC) [44, 45]. Typically, these trajectory planners assume a pre-specified contact sequence and step timing, which are often found using heuristics and/or graph search algorithms [46, 47].

Some of these approaches are based on a change of coordinates that separates the horizontal motion of the CoM into an unstable (divergent) component and a stable (convergent) component [48, 49]. It can be shown that the divergent component is described by the motion of the instantaneous capture point (ICP) [48, 1], defined as

$$\xi := Pc + \frac{\dot{c}}{\omega_0}. \tag{2.15}$$

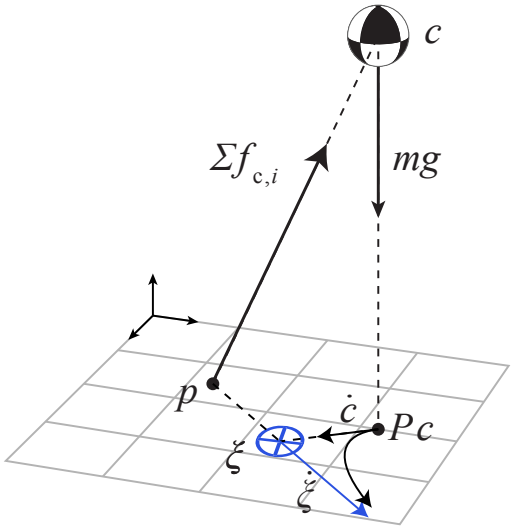


Figure 2-5: Forces acting on a legged robot: gravitational force mg and total contact force $\sum_i f_{c,i}$ acting at the CoP. Motion of the instantaneous capture point ξ is also shown. The ICP diverges away from the CoP in a straight line.

Differentiating this definition and comparing with (2.13) shows that the ICP dynamics are described by

$$\dot{\xi} = \sqrt{\frac{g_z}{z}}\,(\xi - p)\,. \tag{2.16}$$

Fig. 2-5 visualizes the forces acting on a general robot, as well as the ICP dynamics.

The divergent component of motion ξ is also known as the (instantaneous) capture point [36, 50, 1] or extrapolated center of mass [51], and has a very useful interpretation as the point on the ground at which the center of pressure p should be maintained such that the center of mass asymptotically converges to a static equilibrium over p. Whenever the center of pressure is not located at the instantaneous capture point ($p \neq \xi$), (2.16) means that ξ diverges away from p exponentially. As a result, if ξ ever leaves the support polygon (*i.e.*, the set of all possible overall center of pressure locations in the current contact configuration), the robot must modify its support polygon by taking a step in order to avoid exponential divergence. We used this property in [1] to explicitly compute N-step viable-capture basins for the LIP.

In the context of the LIP, planning and control approaches that do not prioritize regulating ξ risk not maximizing the set of states from which balance can be achieved

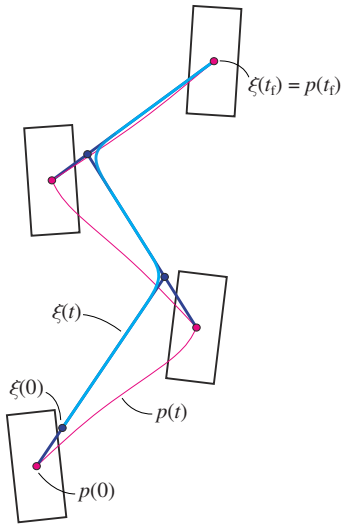


Figure 2-6: Example result of an LIP-based approach to simultaneously planning the trajectory of the CoP p and the the instantaneous capture point (divergent component of motion) ξ from [7]. Overhead view with (preplanned) footsteps shown as rectangles. The instantaneous capture point ξ is repelled by the ground projection of the CoP p_{xy} at all times according to (2.16). At all times, $p(t)$ must remain within the active support polygon formed from the active foot polygons. At the final time t_f, $\xi(t_f) = p_{xy}(t_f)$, which means that the CoM will asymptotically approach a static equilibrium above $p(t_f)$. Adapted from [7].

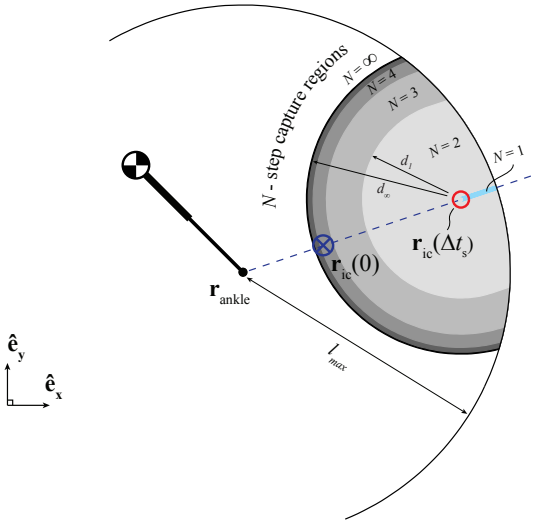


Figure 2-7: Overhead view of N-step capture regions for the linear inverted pendulum with minimum step time and maximum step length constraints in an example state, as derived in [1]. $r_{ic} \equiv \xi$ represents the instantaneous capture point.

for the LIP [1, 52]. See Fig. 2-6 for an example output of a planner using the coordinate change (2.15), resulting in trajectories $\xi(t)$ and $p(t)$. For step recovery, the instantaneous capture point may also be used to compute the regions on the ground to which the robot must step in order to be able to come to a stop in N steps or fewer, called the N-step capture regions [36, 1] (see Fig. 2-7).

Regardless of whether the change of coordinates (2.15) is used, LIP-based feedback control schemes compute a desired trajectory for the rate of change of linear momentum $l(t)$ and corresponding trajectories for the total contact force $\sum_i f_{c,i}(t)$ and center of pressure $p(t)$. A simple linear feedback controller may then be used to track these trajectories, resulting in a reference linear momentum $l_r(t)$ or CoP $p_r(t)$ at every control time t.

2.5.2 Mapping to the full robot

The LIP may be used as a *template model* to control the 'full' robot in various ways. While the LIP has been the go-to template model for decades, up until recently there has been more variation in *whole-body control* approaches that map control actions stemming from strategies designed for the LIP (e.g., $l_r(t)$ or $p_r(t)$) to control actions for the full robot.

One link between the centroidal dynamics (2.6) and the full dynamics (2.2) is formed by the contact forces, which appear in both. This link has been exploited in the *virtual model control* approach, where approximate center of pressure tracking is achieved in torque-controlled walking robots using the concept of a virtual toe point [53, 42]. Virtual model control is essentially a Jacobian-transpose approach.

Another fundamental link is the relationship (2.9). This relationship has been used in the *resolved momentum control* approach to implement various whole-body behaviors on a position-controlled humanoid robot [54]. Achieving a desired momentum can more generally be seen as one example of a *motion task* in the sense of [55], i.e. a linear constraint on the joint velocity vector v. Additional motion tasks may be specified in the nullspace of the momentum control task, and used to achieve e.g. trunk orientation control and swing foot trajectory tracking. The latter task, for

example, may be accomplished by constructing a spline trajectory for the foot and then using a Cartesian PD control law to compute a desired foot twist, which then becomes the right-hand side in a linear constraint on v.

Similar control algorithms may be used for torque-controlled robots, where tasks are typically specified at the acceleration/ force level rather than at the velocity level. For the specific case of momentum tracking, an acceleration-level task suitable for use in a torque-controlled setting can be found by differentiating (2.9):

$$\dot{h} = A(q)\,\dot{v} + \dot{A}(q)\,v.$$

This affine relationship between $\dot{h}$ and $\dot{v}$ may be used to enforce a desired rate of change of centroidal momentum (or just linear momentum) at every control time step, thereby 'mapping' results from a centroidal model to the full robot. The control framework presented in chapter 3 of this thesis is one example of a momentum-based control approach for torque-controlled robots. We refer to this chapter for more details on how results from a centroidal model can be mapped to the full robot. However, for the purposes of the current discussion, performing this mapping can be seen as a more-or-less solved problem with a standard solution.

2.6 Moving beyond the standard approach

This section discusses limitations of the 'standard' approach and some attempts from the literature to avoid these limitations.

While the standard approach has many advantages, including its versatility and relatively low computational requirements, there are also significant limitations. These stem from the fact that there is no notion of the future in the whole-body control approaches discussed in the previous section: the full-dimensional dynamics are never used for planning into the future; rather they are only used to perform an instantaneous transformation from low-dimensional quantities to high-dimensional ones. All of the locomotion planning must be done in terms of the low-dimensional LIP model.

The assumptions used to arrive at the LIP (see Section 2.5.1) thus translate into limitations on the motion of the full robot.

The assumption of constant centroidal angular momentum prohibits lunging or windmilling behaviors that may be used to regain balance in an emergency. The effects of the ability to change centroidal angular momentum on balancing capability are known when the LIP is extended to include a rough model of the rotational inertia of the robot's entire body, approximated as a single lumped rigid body [37, 38, 1], and they are not insignificant. It remains difficult, however, to translate angular excursion limits of the robot's joints into constraints on the behavior of a centroidal template model, such as orientation constraints on the lumped body. Some implementations simply omit constraints on angular momentum or body orientation in the QP formulation while using a low-weight objective term to regulate towards a nominal body pose. This allows QP solutions that result in an 'emergent' lunging behavior. However, without reasoning about joint limits and future posture, this approach could result in hard joint limit impacts, extreme postures, and unintended collisions between the robot's bodies or with the environment.

While properly addressing the issue of non-constant centroidal angular momentum remains an open problem, we will not address it further in this thesis. Instead, the focus will be on the other LIP assumptions, namely that the ground is a flat plane and that the CoM moves on a plane parallel to the ground. These assumptions are problematic in the challenging environments in which the humanoid form factor is supposed to outperform wheeled robots. Control algorithms employing the LIP for planning can typically tolerate ground height variation to a certain extent, but more challenging terrain with sparse available footholds or handholds remains a problem.

Addressing some of the problems with the use of the LIP as a template model will be the focus of chapters 4 and 5. However, we will first discuss the whole-body control framework that may be used to map results of arbitrary centroidal models to the full robot.

Chapter 3

Momentum-based whole-body control

3.1 Introduction

This chapter describes a framework that may used to control a complex humanoid robot based on results for a centroidal model, such as the linear inverted pendulum (LIP). The framework may for example be used to implement a walking behavior for a high-degree-of-freedom humanoid robot such as Boston Dynamics' Atlas robot. The presented control framework originated from the requirements of the DARPA Robotics Challenge (DRC) and its disaster response scenario. This competition required the implementation of a wide variety of behaviors in an environment consisting of not just flat ground, but also tougher terrain including slanted cinder blocks and stairs.

The presented control framework was summarily introduced in [56]. A more detailed description, including hardware experiments on a physical Atlas robot, was presented in [57]. The control framework was also used by Team IHMC to secure second place in both the DRC Trials (December 2013), and the DRC finals (June 2015).

While this chapter is heavily based on [57], the current objective is more to give a brief introduction to the framework within the broader scope of this thesis, and this

chapter has been slimmed down and modified as a result. In particular, results of hardware experiments were omitted from this chapter, and the example application to walking was simplified and updated. The framework will be employed as a tool throughout this thesis to map results from various centroidal models to the full robot.

The presented control framework exploits the fact that the rate of change of whole-body centroidal momentum $\dot{h} \in \mathbb{R}^6$ is simultaneously affine in the joint acceleration vector and linear in the external wrenches applied to the robot [24]. This fact is used to formulate a compact quadratic program, solved at every control time step, which reconciles desired motions with the available frictional contacts between the robot and its environment.

The remainder of this chapter is structured as follows. Section 3.2 discusses related work. Section 3.3 introduces the general control framework. Section 3.4 presents a specific application of the framework to humanoid walking based on the LIP. Section 3.5 details the implementation of the controller used in this thesis (which is different from the implementation used during the DRC). Section 3.6 summarily presents results of flat-ground LIP-based walking in simulation using the controller implementation. Section 3.7 provides a brief discussion, and section 3.7 concludes the chapter.

3.2 Related work

The presented control framework can be classified as a model-based torque control scheme. One of the first such schemes is due to Khatib [58], and was later extended and applied to humanoid robots [59, 60, 61]. It allows the specification of (a hierarchy of) multiple motion tasks, and can handle the case of multiple non-coplanar ground contacts. Joint torques are essentially found by solving an unconstrained linear least squares optimization, or a hierarchy of such problems (*i.e.*, a lexicographic multi-objective [62, 63], or multilevel [64] optimization problem). Due to the lack of inequality constraints, the approach may result in contact forces that violate friction cone constraints. Other examples of this type of approach include the work of Hyon et al. [65, 66] and of Mistry, Buchli, Righetti and Schaal [67, 68].

Taking unilateral ground contact into account naturally leads to the use of constrained optimization. Although the friction force limitations stemming from the Coulomb friction model are naturally represented as second-order cone constraints, the cones are often approximated using linear constraints, reducing the problem of finding joint torques given motion tasks to a quadratic program (QP). The use of quadratic programming in online control algorithms has seen a rise in popularity in the last decade, at least in part due to the availability of fast and reliable QP solver implementations [69, 70, 71, 72] and more powerful CPUs. An early example of such QP-based control schemes is the work of Kudoh et al. [73]. Similar approaches include the work of Macchietto [74], Stephens and Atkeson [75], Saab et al. [76], Righetti et al. [77], Kuindersma et al. [78], Herzog et al. [79], and Feng et al. [80].

Momentum-based control has also gained popularity in recent years. Kajita et al. were the first to propose a momentum-based whole-body control scheme, called Resolved Momentum Control [41]. The authors recognized the fact that the whole-body momentum of a general robot is linear in the joint velocities, and used this to track a desired momentum reference while achieving other motion tasks. The Resolved Momentum Control scheme relied on unconstrained linear least squares to find joint velocity references to be tracked by a low level controller.

The relationship between momentum and joint velocities was studied in more detail by Orin and Goswami [24]. Based on this work, Lee, Goswami, and Orin presented a momentum-based control framework for torque-controlled robots [81, 82, 23]. This work most directly inspired the presented control framework. As opposed to Resolved Momentum Control, the authors employed constrained optimization to take the unilaterality of ground contacts into account. However, the framework is limited in the types of motion tasks that can be expressed. The authors additionally chose to split up the computation of the desired joint accelerations and contact forces into separate optimization problems. Each of these optimization problems is easier to solve than a general QP, resulting in reduced computation time, but also results the approach results in a less general and somewhat less elegant control scheme.

Similar to the work of Lee and Goswami, the presented framework exploits the

properties of whole-body momentum, but recombines the problems of finding joint accelerations and matching contact forces into a single, relatively compact QP. It also enables a simple way of specifying various types of motion tasks. As an example application, the framework ties in neatly with walking control based on instantaneous capture point / divergent component of motion dynamics [1, 42, 43], as will be shown in section 3.4.

3.3 Control framework

Fig. 3-1 shows the high level flow of information in the control framework. A high-level controller, which implements *e.g.* a walking behavior, sets up a quadratic program (QP) based on the following data:

1. Desired motions, in the form of *motion tasks*.

2. Limitations stemming from available contacts.

The QP reconciles the desired motions with the available contacts by exploiting the relationship between whole-body momentum, joint velocities, and externally applied wrenches due to gravity and contacts, (2.10),

$$A(q)\,\dot{v} + \dot{A}(q)\,v = w_g + \sum_i w_{\mathrm{c},i} \tag{3.1}$$

where we recall that $A(q)$ is the centroidal momentum matrix, v is the joint velocity vector, w_g is the wrench due to gravity, and the $w_{\mathrm{c},i}$ are contact wrenches. Quantities are expressed in a centroidal frame, an inertial frame aligned with the global frame, but with its origin instantaneously at the location of the center of mass, c.

The QP solver outputs desired joint accelerations, $\dot{v}_{\mathrm{d}}$, and associated feasible contact wrenches, $w_{\mathrm{c},i}$, that are in agreement with these joint accelerations according to (3.1). This guarantees that the motion is realizable within the context of the friction-limited rigid body model. The joint accelerations and contact wrenches are

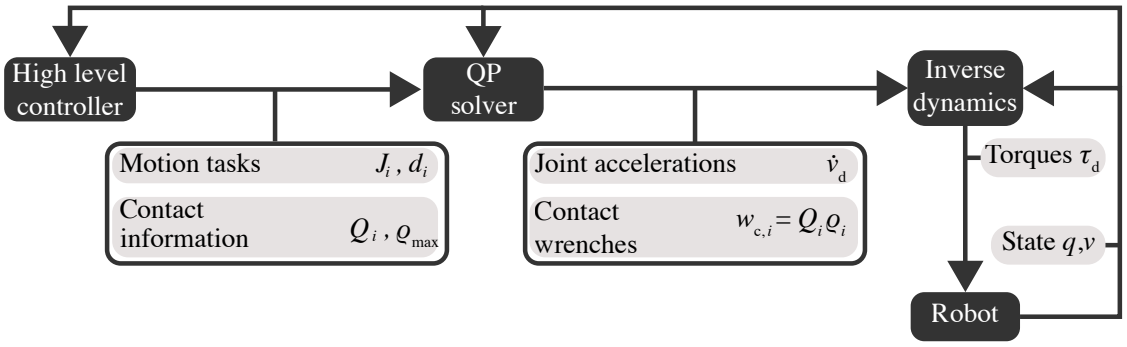


Figure 3-1: High level overview of information flow in controller framework.

then used to compute desired joint torques using an inverse dynamics algorithm. The computed torques are in turn used to determine the motor commands for the robot.

The remainder of this section is structured as follows. Section 3.3.1 discusses the way in which desired motions may be specified. Contact wrenches are addressed in section 3.3.2. The quadratic program is formulated in section 3.3.3. Finally, the inverse dynamics step is discussed in section 3.3.4.

3.3.1 Desired motions

In the proposed framework, desired robot motions are represented in the form of acceleration-level *motion tasks*. Motions appear on the left hand side of the momentum rate balance, (3.1). We define a motion task as an equation that is linear in the desired robot joint acceleration vector, $\dot{v}_\mathrm{d} \in \mathbb{R}^{n_v}$.[1] Here, n_v is the number of velocity degrees of freedom. A motion task with index i can be written as

$$J_i \dot{v}_\mathrm{d} = d_i. \tag{3.2}$$

Motion tasks will appear in the QP formulation either as (hard) constraints, or as penalty terms in the objective function; see section 3.3.3 for more details.

We will first define some basic types of motion tasks, followed by a description of how we typically use these tasks to track reference trajectories.

[1]Note that $\dot{v}_\mathrm{d}$ includes the spatial acceleration of the floating base joint with respect to the world.

Basic types

We mainly use four basic types of motion tasks: 1) joint space acceleration tasks, 2) spatial acceleration tasks, 3) point acceleration tasks, and 4) momentum rate tasks.

In the case of joint space acceleration tasks, $J_i \in \mathbb{R}^{n_{v_j} \times n_v}$ is simply a selection matrix, where n_{v_j} is the number of degrees of freedom of joint j, and d_i is the desired joint acceleration $\dot{v}_{\mathrm{d},j}$:

$$J_i = \begin{pmatrix} 0 & \cdots & 0 & I & 0 & \cdots & 0 \end{pmatrix}, \quad d_i = \dot{v}_{\mathrm{d},j} \tag{3.3}$$

To express a spatial acceleration task, we note that the twist

$$\vartheta_k^{m,j} = \begin{pmatrix} \omega_k^{m,j} \\ v_k^{m,j} \end{pmatrix} \in \mathbb{R}^6 \tag{3.4}$$

of body k with respect to body j, expressed in body m's reference frame (notation adapted from [22]) can be written as

$$\vartheta_k^{m,j} = J_k^{m,j} v \tag{3.5}$$

where $J_k^{m,j} \in \mathbb{R}^{6 \times n_v}$ is a geometric Jacobian [83, 22] (or basic Jacobian [58]). Differentiating and rearranging, we find

$$J_k^{m,j} \dot{v} = \dot{\vartheta}_k^{m,j} - \dot{J}_k^{m,j} v \tag{3.6}$$

Replacing the actual joint acceleration vector $\dot{v}$ with the desired joint acceleration vector $\dot{v}_{\mathrm{d}}$ and the spatial acceleration $\dot{\vartheta}_k^{m,j}$ by its desired value $\dot{\vartheta}_{k,\mathrm{d}}^{m,j}$, we can write this as a motion task with $J_i = J_k^{m,j}$ and

$$d_i = \dot{\vartheta}_{k,\mathrm{d}}^{m,j} - \dot{J}_k^{m,j} v \tag{3.7}$$

Note that spatial acceleration tasks can be expressed between any two of the robot's rigid bodies, not just between a body and the world.

Point acceleration tasks may be derived in similar fashion. In this case $J_i \in \mathbb{R}^{3\times n_v}$ is a Jacobian that maps the joint velocity vector to the Cartesian velocity of a body-fixed point and d_i is the desired linear acceleration of this point minus a velocity-dependent bias term, similar to the right hand side of (3.7).

To express a momentum rate task, we note that the relationship (2.9) can be differentiated to find

$$\dot{h} = A(q)\,\dot{v} + \dot{A}(q)\,v.$$

where we recall that $A(q)$ is the centroidal momentum matrix. Hence, to achieve a desired momentum rate $\dot{h}_{\mathrm{d}}$, we can use a motion task with $J_i = A(q)$ and $d_i = \dot{h}_{\mathrm{d}} - \dot{A}(q)\,v$.

If only certain components of motion should be constrained, each of these basic motion tasks can be premultiplied with a selection matrix S, i.e., $SJ_i\dot{v}_{\mathrm{d}} = Sd_i$. For example, to only constrain the angular acceleration between two bodies, one can use a spatial acceleration task in conjunction with the selection matrix $S = \begin{pmatrix} I_3 & 0_{3\times 3} \end{pmatrix}$.

Typical usage

We typically use motion tasks to track joint space or task space trajectories, in which case the desired joint, spatial, or point acceleration contained in d_i is determined using PD control with an added trajectory-based feed forward acceleration term.

As an example, consider tracking an $SE(3)$ trajectory specifying the reference configuration of body k with respect to body j. The reference configuration can be represented by a homogeneous transform

$$T_{k_r}^{j}(t) = \begin{pmatrix} R_{k,\mathrm{r}}^{j}(t) & p_{k,\mathrm{r}}^{j}(t) \\ 0 & 1 \end{pmatrix}. \tag{3.8}$$

The corresponding reference twist and spatial acceleration trajectories are written in body frame k as $\vartheta_{k,\mathrm{r}}^{k,j}(t)$ and $\dot{\vartheta}_{k,\mathrm{r}}^{k,j}(t)$ respectively. In this case, we use $d_i = \dot{\vartheta}_{k,\mathrm{d}}^{m,j} - \dot{J}_k^{m,j} v$ from (3.7)[2] with $m = k$ and with the desired spatial acceleration $\dot{\vartheta}_{k,\mathrm{d}}^{m,j}$ computed using

[2]The bias term $\dot{J}_k^{m,j} v$ is computed based on measured, not desired velocities.

a double-geodesic PD control law [84], i.e.:

$$\dot{\vartheta}^{k,j}_{k,\mathrm{d}} = \begin{pmatrix} K_{P,\omega} \log_{SO(3)} \left(R^k_{k,\mathrm{r}} \right) \\ K_{P,v} p^k_{k,\mathrm{r}} \end{pmatrix} + K_D \vartheta^{k,k}_{k,\mathrm{r}} + \dot{\vartheta}^{k,j}_{k,\mathrm{r}}. \tag{3.9}$$

Here, $R^k_{k,\mathrm{r}}$ and $p^k_{k,\mathrm{r}}$ are the rotation matrix and translation corresponding to the error transform $T^k_{k,\mathrm{r}} = T^k_j T^j_{k,\mathrm{r}}$, $\vartheta^{k,k}_{k,\mathrm{r}} = \vartheta^{k,j}_{k,\mathrm{r}} - \vartheta^{k,j}_k$ is the error twist, and $K_{P,\omega}$, $K_{P,v}$, and K_D are positive definite gain matrices. Explicit time dependence was suppressed in the notation. In our experience, adding the feed forward reference acceleration term allows the use of lower PD gains and improves tracking performance significantly for dynamic motions as long as smooth trajectories are used. We typically use low-order polynomials as reference trajectories. Slerp interpolation with a low-order polynomial interpolation function is used for orientations.

Load-bearing bodies are typically constrained to have zero spatial acceleration with respect to the world. Alternatively, it is possible to constrain the motion of a load-bearing body such that it simulates damping in Cartesian space, as in Kuindersma et al. [78].

3.3.2 Contact wrenches

This section concerns the right hand side of the momentum rate of change equation, (3.1): the contact wrenches $w_{\mathrm{c},i}$.

Unilateral ground contacts are modeled as point contacts subject to static Coulomb friction. Flat, polygonal contact surfaces can be modeled approximately using such point contacts at the vertices.

For point contact i with normal n_i, the Coulomb friction constraint on the contact force $f_{\mathrm{c},i}$ can be written as (see section 2.3):

$$\| f_{\mathrm{c},i} - (n_i \cdot f_{\mathrm{c},i})\, n_i \| \leq \mu_i n_i \cdot f_{\mathrm{c},i} \tag{3.10}$$

where μ_i is the coefficient of static friction. The second-order cone constraint (3.10) cannot be incorporated in a quadratic program. To allow the use of high-performance

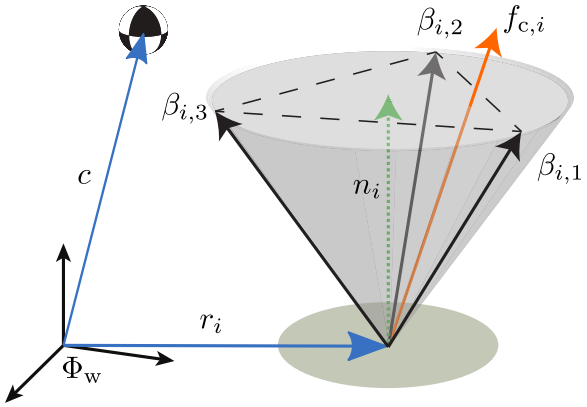


Figure 3-2: A unilateral contact at position r_i in centroidal frame with normal n_i. The friction cone is approximated by basis vectors $\beta_{i,j}$.

QP solvers, we use a standard conservative polyhedral approximation to the friction cone. The approximated friction cone is parameterized using a set of m extreme rays of the original second-order cone as basis vectors [85] (see Fig. 3-2). That is, we replace (3.10) with

$$f_{c,i} = \sum_{j=1}^{m} \rho_{i,j} \beta_{i,j} \tag{3.11}$$
$$\rho_{i,j} \geq 0 \quad \forall j$$

where the $\beta_{i,j} \in \mathbb{R}^3$ are the unit-length basis vectors and $\rho_{i,j}$ are basis vector multipliers (force intensities). For the results presented in this thesis, we use four basis vectors and set $\mu_i = 0.8$ for all i. We may also specify finite upper limits $\rho_{\max,i,j}$ on the multipliers as a rough limit on the total contact force. An upper limit of zero may also be used to for potential contact points that are not actively in contact with the environment. Compared to simply eliminating the decision variables, this has the advantage that the same QP structure is retained, which allows for easier use of warm-starting facilities of QP solvers.

The contact wrench $w_{c,i}$ expressed in the centroidal frame is related to the contact force $f_{c,i}$ by

$$w_{c,i} = \begin{pmatrix} \widehat{r_i - c} \\ I \end{pmatrix} f_{c,i} \tag{3.12}$$

where r_i is the location of contact point i, c is the center of mass location, and $\hat{x} \in \mathbb{R}^{3\times3}$ is the skew symmetric matrix such that $\hat{x}y = x \times y$ for all y. Combining (3.11) and (3.12) with the upper limits, we obtain

$$\begin{aligned} w_{\mathrm{c},i} &= Q_i \rho_i \\ 0 &\leq \rho_i \leq \rho_{\max,i} \end{aligned} \tag{3.13}$$

where

$$Q_i = \begin{pmatrix} \widehat{r_i - c}\beta_{i,1} & \cdots & \widehat{r_i - c}\beta_{i,m} \\ \beta_{i,1} & \cdots & \beta_{i,m} \end{pmatrix} \tag{3.14}$$

is a grasp map [20], $\rho_i = \begin{pmatrix} \rho_{i,1} & \cdots & \rho_{i,m} \end{pmatrix}^T$, and $\rho_{\max,i}$ follows the same pattern.

In practice, we use a surface normal n_i that is fixed in the frame of the contacting robot body. We prefer the extreme ray parameterization from Pollard and Reitsma [85] to the alternative parameterization in Stewart and Trinkle [86] because the extreme ray parameterization requires one fewer decision variable per contact point.

3.3.3 QP formulation

To reconcile the motion tasks (3.2) with the available contacts (3.13) while satisfying the momentum rate balance (3.1) (with desired joint accelerations $\dot{v}_\mathrm{d}$ substituted for $\dot{v}$), we formulate the following QP:

$$\begin{aligned} \underset{\dot{v}_\mathrm{d},\rho}{\text{minimize}} \quad & \sum_{i\in I_\mathrm{o}} \|J_i\dot{v}_d - d_i\|^2_{C_i} + \|\rho\|^2_{C_\rho} + \|\dot{v}_\mathrm{d}\|^2_{C_{\dot{v}}} \\ \text{subject to} \quad & \sum_{i\in I_\mathrm{c}} \|J_i\dot{v}_d - d_i\|^2_{C_i} \\ & A\dot{v}_\mathrm{d} + \dot{A}v = w_g + \sum_i Q_i\rho_i \\ & 0 \leq \rho \leq \rho_{\max} \end{aligned}$$

where the indices of the motion tasks have been split up into a subset I_o, to be incorporated into the objective function, and its complement, I_c, to be enforced as

a hard constraint. Here, $\|x\|_C^2 = x^T C x$. The matrices $\{C_i\}_{i \in I_o}$, C_ρ, and $C_{\dot{v}}$ are cost function weighting matrices determined by the high level controller. In particular, C_ρ and $C_{\dot{v}}$ are used for regularization, and $C_{\dot{v}}$ may be used to implement a damped least squares approach [87] within this framework. Example values of these weighting matrices are given for the walking behavior presented in section 3.4.2.

This quadratic program is solved at every controller time step for the desired joint acceleration vector $\dot{v}_\mathrm{d}$ and the basis vector multiplier vector ρ. For the results in this thesis, we used the OSQP solver [72]. Given the multipliers ρ, contact wrenches at each contact point are computed as $w_{\mathrm{c},i} = Q_i \rho_i$.

3.3.4 Inverse dynamics

The contact wrenches $w_{\mathrm{c},i}$ and desired joint accelerations $\dot{v}_\mathrm{d}$ are used as the input to a recursive Newton-Euler inverse dynamics algorithm [88, 21] to solve (2.2), resulting in desired joint torques τ_d for all joints, as well as a residual wrench that should be exerted across a 6-degree-of-freedom floating joint that connects one of the bodies (in this work, the pelvis) to the world. Because the desired joint accelerations are compatible with the total contact wrench, this residual wrench is identically zero. Using a recursive Newton-Euler algorithm obviates the need to explicitly compute the mass matrix.

3.4 Application to humanoid walking

This section discusses an example application of the presented control framework. The framework is used to map planning results from the LIP to a simulation model of the Boston Dynamics Atlas robot, resulting in a basic walking behavior.

Atlas has 30 actuated degrees of freedom: 6 in each leg, 3 in the back, 7 in each arm, and one in the neck.

Section 3.4.1 discusses the footstep plan. Section 3.4.2 presents the motion task setup used for both the LIP-based walking behavior here, as well as throughout this thesis. Section 3.4.3 discusses the LIP-specific generation of the linear momentum

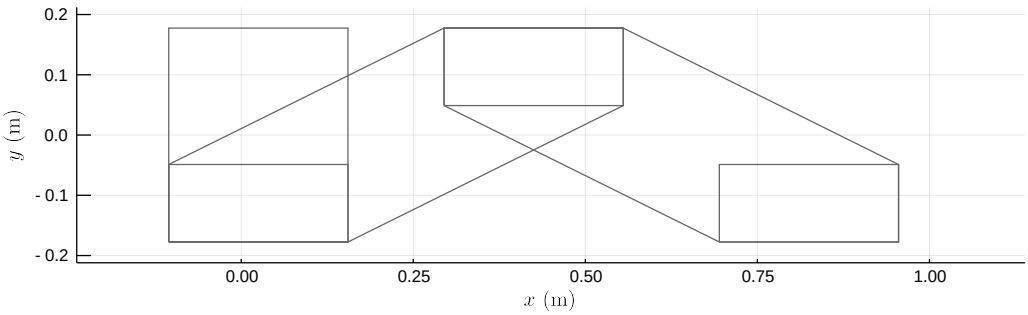


Figure 3-3: Overhead view of the support polygons for a two-step footstep plan. The footstep plan used for subsequent simulation results is similar, but consists of seven steps.

rate reference.

3.4.1 Footstep plan

We use the term *footstep plan* to refer to a timed sequence of desired landing poses for the feet. While locomotion over rough terrain with sparse footholds requires special attention as to where and when to step, generating a workable footstep plan for straight-line walking on flat ground is trivial. For the purposes of this example application, we manually generated such a plan, consisting of equally spaced steps. A step length of 0.4 m was used, with 0.75 s allocated for each single support phase, and 0.55 s for each double support phase. See Fig. 3-3 for an overhead view of the footstep plan.

The footstep plan dictates when foot-fixed contact points become active, and provides endpoints for swing foot trajectory generation, as discussed in the next section.

3.4.2 Motion tasks and regularization

See table 3.1 for a list of motion tasks that were used for both the LIP-based walking example presented here, as well as the results for other centroidal models in this thesis. The types and selection matrices refer to terminology introduced in section 3.3.1. For this application, we opted to only use 'soft' motion tasks, appearing as

Task	Type	Count	Selection matrix S	Weight C_i
Linear momentum rate control	Momentum rate	1	$\begin{bmatrix} 0_{3\times3} & I_3 \end{bmatrix}$	1
Foot pose control w.r.t. world	Spatial acceleration	2	I_6	10
Pelvis orientation control	Spatial acceleration	1	$\begin{bmatrix} I_3 & 0_{3\times3} \end{bmatrix}$	10
Upper body posture control	Joint acceleration	17	I	10

Table 3.1: List of motion tasks.

objective terms in the QP.

The desired value for the linear momentum rate task will be discussed in section 3.4.3.

Desired spatial accelerations for the foot motion tasks are composed of a feed-forward term and a PD feed-back term, as in (3.9). During a stance phase, the feed-forward term is zero, and only the derivative gains for the angular component of the feed-back term are non-zero. This is to dampen out unwanted of the feet while still being compliant to the environment. For a swing phase, we generate a simple polynomial trajectory for the origin of a reference point on the sole of the foot. The trajectory is constructed given:

- the swing duration, from the footstep plan;
- the initial position of the sole reference point;
- the final desired position of the sole reference point, from the footstep plan;
- a desired height at the midpoint of the trajectory;
- a final desired vertical foot velocity.

Based on these data, a cubic trajectory is constructed for the component of the trajectory in the horizontal plane, and a quadratic is used for the vertical component. These trajectories are a function of an interpolation parameter θ, ranging from 0 to 1, which itself is a quintic polynomial of time with zero initial and final first and second derivatives. This results in a smooth interpolation with gradual acceleration and deceleration. The desired angular acceleration is computed based on slerp interpolation

between the initial actual and final desired orientation, with a quintic polynomial interpolant.

The desired angular acceleration for the pelvis orientation task is computed similarly. For the pelvis, the final desired orientation matches the orientation of the upcoming desired footstep in the footstep plan.

The right-hand sides for the joint-space tasks for the upper body simply stem from PD control towards fixed, nominal joint angles.

The weighting matrices $C_{\dot{v}}$ and C_ρ, which respectively regularize joint accelerations and contact force basis wrench multipliers, were set to $C_{\dot{v}} = 0.05 \cdot I$ and $C_\rho = 0.001 \cdot I$. These values were tuned rather roughly by increasing them until linear momentum rate of change tracking started to deteriorate in simulation.

3.4.3 Desired Linear momentum rate

Given the footstep plan, the overall motion of the robot is planned and controlled based on instantaneous capture point (ICP) dynamics, (2.16), briefly introduced in section (2.5). This part of the walking behavior is specific to the use of the LIP, and may be replaced if another centroidal model is used.

In earlier work, we used the simple ICP trajectory generation procedure of [7] (see Fig. 2-6). Here, we instead formulate a quadratic program to simultaneously find a piecewise-polynomial reference CoP trajectory $p_{\mathrm{r}}(t)$ and an associated desired ICP trajectory $\xi_{\mathrm{r}}(t)$, with constraints stemming from:

- the (linear, low-dimensional) ICP dynamics (2.16);
- initial and final ICP positions;
- initial and final CoP positions;
- C^1 continuity of the CoP trajectory;
- the fact that the CoP must at all times be in the currently active support polygon, as dictated by the footstep plan.

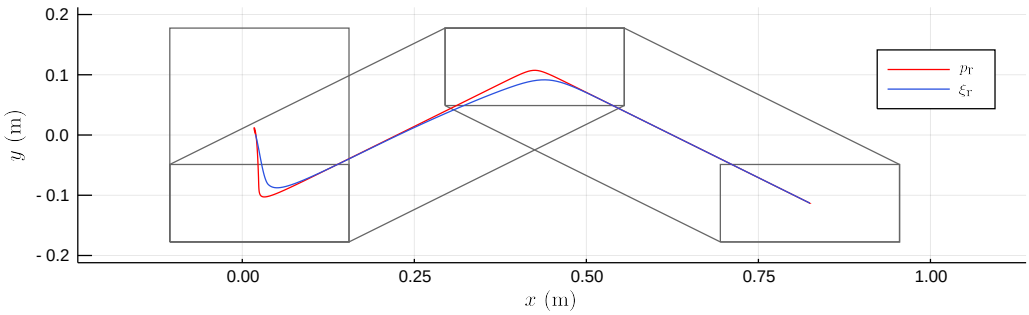


Figure 3-4: Overhead view of the output of the joint CoP and ICP trajectory generator: CoP reference p_r and ICP reference (ξ_r).

These constraints are implemented by parameterizing the CoP trajectory $p(t)$ as a piecewise Bézier curve and exploiting the Bézier convex hull property to turn constraints on the trajectory $p(t)$ into constraints on the control points of the pieces. See section 5.4.3 for a more in-depth discussion of a similar application of the Bézier convex hull property. See Fig. 3-4 for the output (p_r, ξ_r) of this trajectory generation approach.

To track the resulting ξ_r trajectory, we use (2.16) as the basis for an ICP control law aimed at tracking these desired values:

$$p_d = \xi - \frac{1}{\omega_0}\dot{\xi}_r + k_{ic}(\xi - \xi_r) \tag{3.15}$$

where p_d is the desired location of the CoP and $k_\xi > 0$ is a proportional feedback gain (we use $k_\xi = 3$). We find that the feedforward term involving $\dot{\xi}_d$ greatly improves tracking performance compared to the control law presented in [42].

Given a desired CoP obtained from (3.15), the horizontal component of the desired linear momentum rate of change is computed based on (2.12):

$$P\dot{l}_d = Pm\ddot{c} = P\frac{mg_z}{z_0}(c - p_d) \tag{3.16}$$

where $P = \left(\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right)$. This serves as the horizontal linear momentum rate of change for the control framework. The desired vertical linear momentum rate of change is provided by a simple PD controller with a constant reference CoM height, using critically damped gains with a proportional gain of 10.

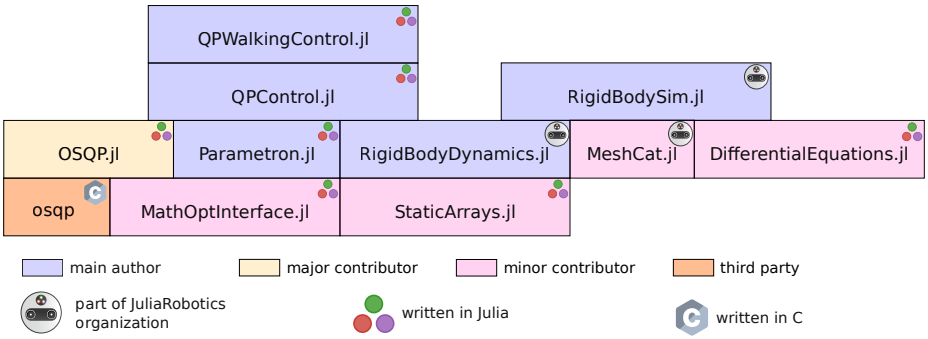


Figure 3-5: Main components of the software stack for the presented controller, including simulation. The stacking structure visualizes each package's main dependencies.

3.5 Implementation

Except for the quadratic program solver (OSQP, [72]), the implementation of the controller used in this thesis was done using a pure Julia software stack. See Fig. 3-5 for an overview of the software stack, and see [89] for a more in-depth description. The entire stack is freely available and published under permissive software licenses. To highlight a few of these packages:

- RigidBodyDynamics.jl[3] implements various dynamics and kinematics algorithms.
- RigidBodySim.jl[4] is used for simulation and visualization.
- Parametron.jl[5] enables efficient formulation and solution of instances of a parameterized family of optimization problems in a solver-independent way.
- QPControl.jl[6] implements the parts of the controller framework that are independent of robot morphology (section 3.3).
- QPWalkingControl.jl[7] implements the humanoid-specific parts of the controller on top of QPControl.jl (section 3.4).

[3] `https://github.com/JuliaRobotics/RigidBodyDynamics.jl`
[4] `https://github.com/JuliaRobotics/RigidBodySim.jl`
[5] `https://github.com/tkoolen/Parametron.jl`
[6] `https://github.com/tkoolen/QPControl.jl`
[7] `https://github.com/tkoolen/QPWalkingControl.jl`

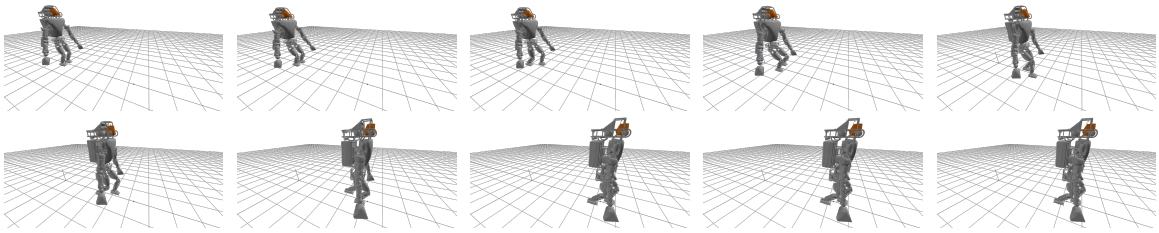


Figure 3-6: Atlas walking on flat ground.

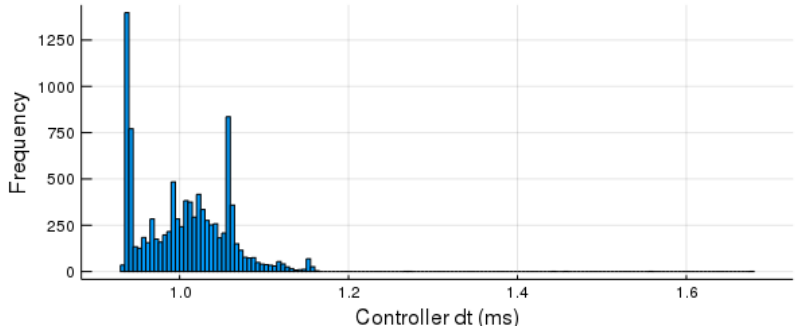


Figure 3-7: Histogram of controller run times. There is one sample at 1.67 ms.

The controller is run at a simulated 500 Hz. For the simulation results presented in this thesis, there are no external disturbances acting on the robot and state estimation is perfect.

3.6 Results

See Fig. 3-6 for results of an 18-second simulation of Atlas walking in a straight line on flat ground using the momentum-based control framework and the walking behavior detailed in section 3.4.

Fig. 3-7 shows a histogram of controller runtime durations during this simulation, obtained on a desktop machine with an Intel Core i7-6950X CPU @ 3.00GHz running Linux. No special precautions were taken to limit jitter, apart from using the performance CPU power governor.

3.7 Discussion

The presented approach can be interpreted in various ways. It can be seen as a way to perform partial feedback linearization for underactuated robots that are in contact with their environment. Indeed, the feedback control laws all act in joint acceleration space, under the nominal dynamics $\dot{v} = \dot{v}_{\mathrm{d}}$. This allows feedback gains to generally be chosen more-or-less independent of any specific robot's inertial parameters, as the joint accelerations and expected contact wrenches are transformed into joint torques during the model-based inverse dynamics step. Alternatively, the approach can be seen as a type of degenerate model-predictive control approach, using a horizon of just a single time step.

The controller implementation presented here is missing some of the robustness improvements that allowed it to work on the physical Atlas robot [57, 90]. These techniques include augmenting the ICP control law (3.15) with an integral term, and integrating the desired joint accelerations $\dot{v}_{\mathrm{d}}$ to obtain a desired velocity signal that can be tracked using high-rate joint-local controllers, augmented by the feed-forward torque component τ_{d}. Despite these missing features, Fig. 3-7 shows that the implementation is already fast enough to be run on a physical robot while comfortably hitting the 500 Hz deadlines. Jitter may be further reduced through the use of a realtime kernel patch.

In contrast to *e.g.* [78], an important advantage of momentum-based control approaches like the presented control framework is that the high-rate tracking component of the controller does not make any fundamental assumptions regarding the motion of the robot. This will allow us to switch away from the LIP as the source of the linear momentum reference trajectory in the following chapters.

3.8 Conclusion

This chapter presented a QP-based control framework that may be used to track a desired centroidal momentum or center of mass trajectory, while simultaneously sat-

isfying secondary motion tasks and taking limitations due to static Coulomb friction into account. A walking behavior based on LIP dynamics was presented as an example application of the control framework. Subsequent chapters will investigate the use of other centroidal models to find a centroidal momentum reference.

Chapter 4

2D balance control using non-planar CoM motion

4.1 Introduction

While the centroidal dynamics, (2.6), are low-dimensional and control-affine in the contact wrenches, the constraints on these contact wrenches are nontrivial: contact is unilateral, and forces must remain within friction cones. For this reason, the centroidal dynamics are often further simplified by adding artificial constraints on the external wrenches, or equivalently, on the motion of the robot. Most commonly, angular momentum about the CoM is assumed to be constant (typically zero), and CoM height is assumed to be an affine (typically constant) function of horizontal CoM position. These assumptions define the Linear Inverted Pendulum (LIP) [34], which has long been a fixture in the design of balancing and walking controllers for legged robots. The LIP naturally leads to control strategies involving stepping and regulation of the center of pressure (CoP).

Although the LIP dynamics are easy to work with, the CoM height assumption in particular is overly constraining. This becomes apparent when the robot is *e.g.* required to dynamically step up onto a platform. Moreover, varying CoM height in an appropriate manner can be used as an additional mechanism to achieve balance.

Recently, efforts have been made to finally move away from the CoM height as-

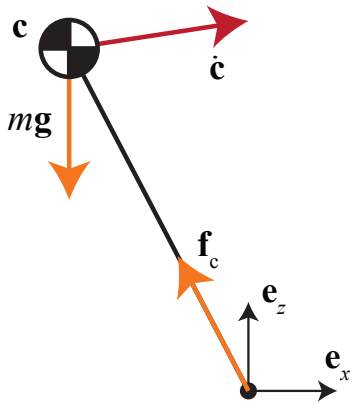


Figure 4-1: Variable-height inverted pendulum model.

sumption. In [91], the dynamics of the divergent component of motion, also known as the instantaneous capture point, are extended to 3D and used to find a control law. In [92], numerical techniques are used to find quadratic height trajectories that solve the balancing problem for a variable-height inverted pendulum model. An important earlier work derived conditions that characterize when a given CoM height trajectory leads to balance [50], however without paying much attention to the constraint of unilateral contact.

This chapter studies a 2D variable-height inverted pendulum model (see Fig. 4-1) and provides three main contributions. First, we derive an intuitively appealing outer approximation to the set of initial CoM positions and velocities that allow balance to be achieved, in the sense of convergence to a fixed point of the dynamics. Second, we design two *closed-form* balance control laws based on [50] (in contrast to the numerical techniques used in [92]). Third, we derive the exact regions of attraction of these control laws in closed form using quantifier elimination, explicitly taking unilateral contact into account. These regions of attraction are also inner approximations to the set of CoM states from which balance can be achieved. We show that the region of attraction for one of the controllers matches the outer approximation, and hence achieves balance from any state from which balance can possibly be achieved.

The remainder of the chapter is structured as follows. Section 4.2 derives the dynamics of the variable-height model under consideration. Section 4.3 derives necessary conditions for balance, *i.e.*, the outer approximation. We then summarize the

results of [50] (section 4.4), which form the basis of the control laws (section 4.5) and the derivation of their regions of attraction (section 4.6). Section 4.7 discusses the results, and section 4.8 concludes the chapter.

Code and additional figures accompanying the chapter can be found at `https://github.com/tkoolen/VariableHeightInvertedPendulum`.

4.2 Variable-height inverted pendulum

The equations of motion of the 2D variable-height inverted pendulum (see Fig. 4-1) are

$$m\ddot{c} = mg + f_{\mathrm{c}} \tag{4.1}$$

where $c = (c_x, c_z)$ is the CoM position relative to the fixed point foot, m is the robot's total mass, $g = (0, -g_z)$ is the gravitational acceleration vector, and $f_{\mathrm{c}} = \sum_i f_{\mathrm{c},i}$ is the total contact force (see section 2.3). We assume that angular momentum about the CoM remains constant, implying that the line of action of the ground reaction force must pass through the CoM. This allows the ground reaction force to be parameterized as

$$f_{\mathrm{c}} = mcu \tag{4.2}$$

where u is a scalar control input. Combining (4.2) and (4.1), we find

$$\ddot{c} = g + cu \tag{4.3}$$

as the dynamics of the variable-height inverted pendulum. We assume unilateral contact (pulling on the ground is impossible), so we require that

$$u \geq 0. \tag{4.4}$$

By inspection, fixed points of the dynamics (4.3) must satisfy $\dot{c}_x = \dot{c}_z = 0$, $c_x = 0$, and $zu = g_z$, so $c_z > 0$. We are interested in achieving 'balance', in the sense of

asymptotic convergence to a fixed point of the dynamics[1] at a specified desired final height $c_z = c_{z,\mathrm{f}} > 0$.

Consider a state like the one depicted in Fig. 4-1, and suppose the initial horizontal velocity, $\dot{c}_x$, is 'too large' in some sense. To achieve balance, a large value of u can be used to decrease $\dot{c}_x$ quickly. However, this also has the effect of increasing $\dot{c}_z$, and over time, c_z, as a byproduct. The challenge is to control both horizontal and vertical CoM position with a single control input by exploiting the state-dependent variation of the effect of u in the nonlinear dynamics (4.3).

4.3 Necessary conditions for balance

In this section we investigate conditions that must be satisfied in any case if balance is to be achieved.

Let the state of the variable-height inverted pendulum be $x = (c_x, c_z, \dot{c}_x, \dot{c}_z)$. Starting from x, convergence to the fixed point requires that the horizontal CoM position c_x and velocity $\dot{c}_x$ satisfy

$$c_x \dot{c}_x < 0 \tag{4.5}$$

This is because $\dot{c}_x$ would otherwise increase without bound, since (4.3) and (4.4) imply that $\mathrm{sign}\,(\ddot{c}_x) = \mathrm{sign}\,(c_x)$.

Another necessary condition for balance stems from the extreme control policy of choosing $u = 0$ for all time. This policy results in a ballistic CoM trajectory. We will show that it is impossible to reach a fixed point of the dynamics from state x if the c_z-intercept of the ballistic trajectory starting from x is nonpositive, making it impossible to achieve balance.

The ballistic trajectory starting from state x is

$$c_{\mathrm{bal}}\,(x,t) = \begin{bmatrix} c_{x,\mathrm{bal}}\,(x,t) \\ c_{z,\mathrm{bal}}\,(x,t) \end{bmatrix} = \begin{bmatrix} c_x + \dot{c}_x t \\ c_z + \dot{c}_z t - \frac{1}{2} g_z t^2 \end{bmatrix}. \tag{4.6}$$

[1]Note that we are not interested in achieving asymptotic *stability* of a fixed point, which is impossible in the variable-height inverted pendulum.

Let T be the time at which $c_{x,\mathrm{bal}}(x,T)=0$, $T=-\frac{c_x}{\dot{c}_x}$, and let

$$c_{z,\mathrm{crit}}(x)=c_{z,\mathrm{bal}}(x,T)=c_z-\frac{\dot{c}_z c_x}{\dot{c}_x}-\frac{g_z x^2}{2\dot{c}_x^2}$$

be the c_z-intercept of the ballistic trajectory.

Lemma 1 *Suppose $c_{z,crit}(x)\leq 0$ and assume (4.5), so that $T\geq 0$. Then no state $c_{x,f}=(c_{x,f},c_{z,f},\dot{c}_{xf},\dot{c}_{zf})$ for which $c_{x,f}=0$ and $c_{z,f}>0$ is reachable from x given the dynamics (4.3) and input limit (4.4).*

Proof 1 *The time derivative of $c_{z,crit}(x)$ along trajectories of (4.3) is*

$$\begin{aligned}\frac{d}{dt}c_{z,crit}(x)&=\frac{\partial c_{z,crit}(x)}{\partial c}\dot{c}+\frac{\partial c_{z,crit}(x)}{\partial \dot{c}}(g_z+cu)\\&=u\frac{c_x}{\dot{c}_x}\frac{1}{\dot{c}_x^2}\left(g_z x^2+\dot{c}_x(c_x\dot{c}_z-\dot{c}_x c_z)\right).\end{aligned}$$

The condition $c_{z,crit}(x)\leq 0$ can be rearranged to find

$$\dot{c}_x(c_x\dot{c}_z-\dot{c}_x c_z)\geq -\frac{g_z x^2}{2}.$$

Together with $u\geq 0$ according to (4.3), $\frac{c_x}{\dot{c}_x}<0$ according to (4.5), and $g_z>0$, we infer that

$$\frac{d}{dt}c_{z,crit}(x)\leq 0$$

so we conclude that $c_{z,crit}$ cannot increase along trajectories of (4.3), and if $c_{z,crit}(c_{x,f})>c_{z,crit}(x)$, then $c_{x,f}$ must not be reachable from x. Noting that $c_{z,crit}(c_{x,f})=c_{z,f}$ and $c_{z,f}>c_{z,crit}(x)$ completes the proof.

Since any fixed point of the dynamics must satisfy $c_{x,\mathrm{f}}=0$ and $c_{z,\mathrm{f}}>0$, Lemma 1 provides the condition

$$c_{z,\mathrm{crit}}(x)>0 \tag{4.7}$$

as a useful necessary condition for balance, describing an outer approximation of the set of states that allow balance. In the following sections, we will derive a feedback

control law with a region of attraction described exactly by (4.7), assuming only unilateral contact and no kinematic constraints or actuation limits. This will show that the outer approximation is tight.

4.4 Approach and summary of previous results

Our general approach to control is to design and enforce a virtual constraint [93]. Virtual constraints were first applied to the variable-height inverted pendulum in [50]. This section summarizes their results, providing no new contributions.

4.4.1 Height trajectory as a virtual constraint

Consider the time-invariant virtual holonomic constraint

$$c_z = f(c_x) \tag{4.8}$$

where f describes a desired CoM height trajectory.

We will assume that the initial value and slope of the CoM height trajectory match the initial state $x_0 = (c_{x,0}, c_{z,0}, \dot{c}_{x,0}, \dot{c}_{z,0})$, as depicted in Fig. 4-2:

$$f(c_{x,0}) = c_{z,0} \tag{4.9a}$$

$$f'(c_{x,0}) = \frac{\dot{c}_{z,0}}{\dot{c}_{x,0}}. \tag{4.9b}$$

Differentiating (4.8) twice with respect to time gives

$$\ddot{c}_z = f'(c_x)\,\ddot{c}_x + f''(c_x)\,\dot{c}_x^2.$$

Substituting this into (4.3), we can solve for the input u required to enforce the virtual constraint:

$$u = \frac{g_z + f''(c_x)\,\dot{c}_x^2}{\bar{f}(c_x)} \tag{4.10}$$

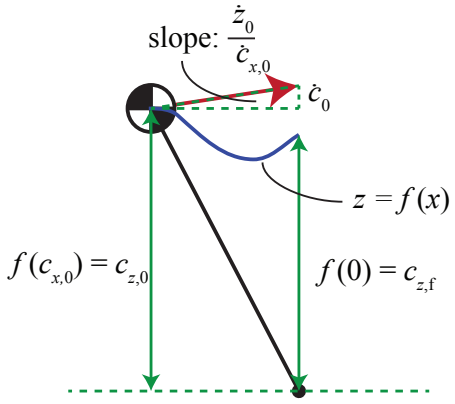


Figure 4-2: Kinematic constraints placed on CoM height trajectory f.

where

$$\bar{f}\left(c_x\right) = f\left(c_x\right) - f'\left(c_x\right) c_x. \tag{4.11}$$

The dynamics of the unconstrained degree of freedom c_x are simply

$$\ddot{c}_x = ux. \tag{4.12}$$

Example 1 *For the LIP, f is an affine function, fully specified by the continuity constraints (4.9):*

$$f\left(c_x\right) = c_{z,0} + \frac{\dot{c}_{z,0}}{\dot{c}_{x,0}} c_x.$$

The required input simplifies to a constant,

$$u = \frac{g_z}{c_{z,0}}$$

showing that as long as $c_{z,0} > 0$, the condition $u \geq 0$ is always satisfied (no pulling on the ground). Equation (4.12) becomes

$$\ddot{c}_x = \frac{g_z}{c_{z,0}} c_x, \tag{4.13}$$

the familiar LIP dynamics. ▲

By not artificially constraining f to be an affine function, balance can be achieved from more initial states than what is allowed by the LIP dynamics. We now investigate

how the requirement of achieving balance constrains f.

4.4.2 Orbital energy

Orbital energy can be used to decide which states converge to the fixed point while tracking a given height trajectory f.[2] We will first introduce orbital energy for the LIP, and then generalize to arbitrary f.

LIP

For the LIP, orbital energy [34] is a well-known conserved quantity, defined as:

$$E_{\text{LIP}}\left(c_x, \dot{c}_x\right) = \frac{1}{2}\dot{c}_x^2 - \frac{g_z}{2z_0}c_x^2. \tag{4.14}$$

The fact that LIP orbital energy is conserved can be shown by taking the time derivative of (4.14) and plugging in the dynamics (4.13). To achieve balance, conservation of orbital energy implies that the initial orbital energy must be the same as the orbital energy at the fixed point,

$$E_{\text{LIP}}\left(c_{x,0}, \dot{c}_{x,0}\right) = E_{\text{LIP}}\left(0, 0\right) = 0$$

If (4.5) is also satisfied at x_0, this results in the familiar condition

$$c_{x,0} + \sqrt{\frac{c_{z,0}}{g_z}}\dot{c}_{x,0} = 0. \tag{4.15}$$

The left hand side is known as the instantaneous capture point [36, 1], extrapolated center of mass [51], or divergent component of motion [49].

[2] Note that orbital energy is *not* the same as the total energy of the system.

Variable-height model

Perhaps a less known fact is that a conserved orbital energy exists for *any* C^2 height trajectory f. As derived in [50], the orbital energy associated with f is

$$E_f\left(c_x, \dot{c}_x\right) = \frac{1}{2}\dot{c}_x^2 \bar{f}^2\left(c_x\right) + g_z c_x^2 f\left(c_x\right) - 3g_z \int_0^{c_x} f\left(\xi\right)\xi d\xi \tag{4.16}$$

with $\bar{f}\left(c_x\right)$ as defined in (4.11). Analogous to the LIP, the fact that orbital energy is conserved can be (tediously) verified by taking the time derivative and plugging in the dynamics (4.12). Again, balance requires that $c_{x,0}$ and $\dot{c}_{x,0}$ have opposite sign and that

$$E_f\left(c_{x,0}, \dot{c}_{x,0}\right) = E_f\left(0, 0\right) = 0.$$

With the continuity constraints from (4.9), this requirement simplifies to

$$3g_z \int_0^{c_{x,0}} f\left(\xi\right)\xi d\xi = k \tag{4.17}$$

with

$$k = \frac{1}{2}\left(\dot{c}_{x,0}c_{z,0} - \dot{c}_{z,0}c_{x,0}\right)^2 + g_z x_0^2 c_{z,0}.$$

In general, the integral on the left hand side of (4.17) may not be computable in closed form, but if we restrict the class of height trajectories f to polynomials,

$$f\left(c_x\right) = \sum_{i=0}^{n} \alpha_i c_x^i,$$

then the integral is readily evaluated and (4.17) can be written as a linear constraint on the coefficients α_i:

$$3g_z \sum_{i=0}^{n} \frac{1}{i+2}\alpha_i c_{x,0}^{i+2} = k. \tag{4.18}$$

This constraint will be used to find a feedback control law in the following section.

4.5 Control laws

This section presents two feedback control laws based on virtual constraints and orbital energy.

We will take height trajectory f to be a cubic polynomial ($n = 3$), and impose four linear constraints that uniquely determine its coefficients:

1) the final desired height: $f(0) = c_{z,\mathrm{f}}$;

2–3) the continuity constraints (4.9);

4) the orbital energy constraint (4.18).

These constraints can be written concisely as

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & c_{x,0} & c_{x,0}^2 & c_{x,0}^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ \frac{3}{2}g_z x_0^2 & g_z x_0^3 & \frac{3}{4}g_z x_0^4 & \frac{3}{5}g_z x_0^5 \end{bmatrix}}_{A} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} c_{z,\mathrm{f}} \\ c_{z,0} \\ \frac{\dot{c}_{z,0}}{\dot{c}_{x,0}} \\ k \end{bmatrix}.$$

The determinant of matrix A is $\frac{g_z x_0^7}{10}$, showing that a solution exists as long as $c_{x,0} \neq 0$.

We omit the solution for the coefficients α_i here, since the expressions are somewhat long and uninformative. It is important to note however that the α_i will depend *rationally* on x_0 (as well as $c_{z,\mathrm{f}}$ and g_z). Substituting the α_i back into f shows that f is also a rational function of c_x and x_0. Furthermore, note that the term $\dot{c}_x^2$ in (4.10) can be solved for given f and c_x using (4.16), since $E_f(c_x, \dot{c}_x) = 0$ by construction of f. This observation allows us to even find u in closed form, as a rational function of only c_x and x_0 after substitution into (4.10):

$$u = U(c_x, x_0) = \frac{p(c_x, x_0)}{q(c_x, x_0)} \tag{4.19}$$

where $p(c_x, x_0)$ and $q(c_x, x_0)$ are polynomials of respective total degrees 20 and 21 in the variables (c_x, x_0).

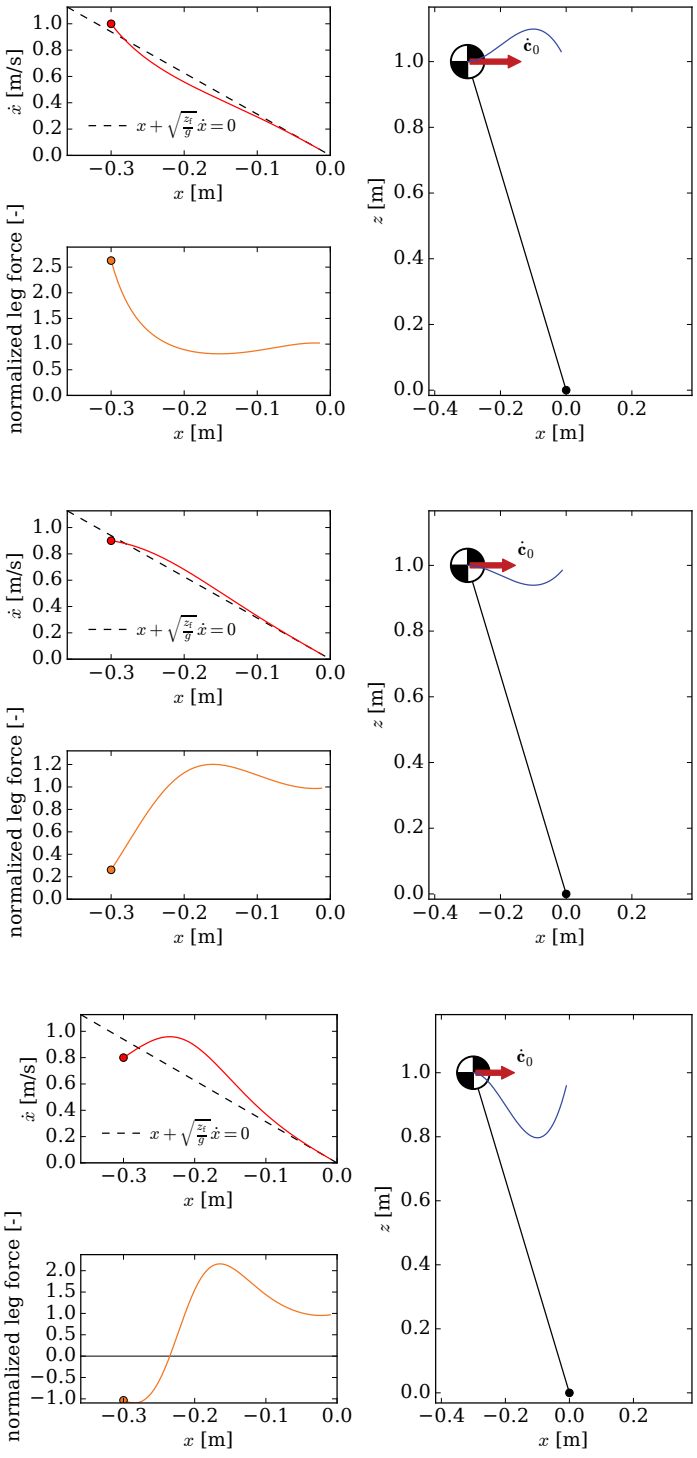


Figure 4-3: Simulation results for orbital energy controller (4.20) with initial conditions $c_{x,0} = -0.3$ [m], $c_{z,0} = 1$ [m], $\dot{c}_{z,0} = 0$ [m/s] and three values of $\dot{c}_{x,0}$: 1.0 [m/s] (top), 0.9 [m/s] (middle), and 0.8 [m/s] (bottom). The normalized leg force is computed as $\frac{f_{\text{gr}}}{mg_z} \cdot \frac{c}{\|c\|} \equiv \frac{1}{g_z} u \, \|c\|$. For this plot, $g_z = 9.8$ [m/s^2], $c_{z,\text{f}} = 1$ [m]. For the third (slowest) initial condition, the simulation was performed as if pulling on the ground were possible.

To arrive at a feedback controller, our strategy is to essentially 'continually resolve' for the CoM height trajectory f by substituting the current state x for x_0 in (4.19). This substitution also greatly simplifies the expression. The resulting control law is:

$$u = U(c_x, x) = -7a^2 + \frac{3c_{z,\mathrm{f}}a^3 - g_z a}{b} - \frac{10a^3 b}{g_z}. \tag{4.20}$$

with

$$a = \frac{\dot{c}_x}{c_x} \qquad\qquad b = \dot{c}_z - az. \tag{4.21}$$

We will refer to this control law as the 'orbital energy controller'. The orbital energy controller has the property that it keeps the CoM height on the cubic height trajectory f in the absence of external disturbances to the dynamics (4.3). This is because the control law ensures that $\dot{x}$ is tangent to the constraint manifold described by $c_z = f(c_x)$ since it stems from (4.10), and because f is uniquely specified by the initial conditions.

Note the singularities at $c_x = 0$ and $b = 0$. The singularity at $c_x = 0$ is due to the fundamental lack of effect of the control input u on $\ddot{c}_x$ when $c_x = 0$. The singularity at $b = 0$ occurs when the CoM velocity vector points from the CoM toward the point foot. This implies that the necessary condition for balance derived earlier, (4.7), is not satisfied, so states for which $b = 0$ are not of interest. The root cause of this singularity is the fact that the input u that enforces constraint (4.8) is not uniquely defined in this case.

See Fig. 4-3 for the results of simulations with the orbital energy controller, starting from three example initial conditions. Note that the orbital energy controller will attempt to pull on the ground if the initial velocity is very low, as shown by the negative normalized ground reaction forces. To address this, we will also consider the following 'clipped controller':

$$u = \max(U(c_x, x), 0). \tag{4.22}$$

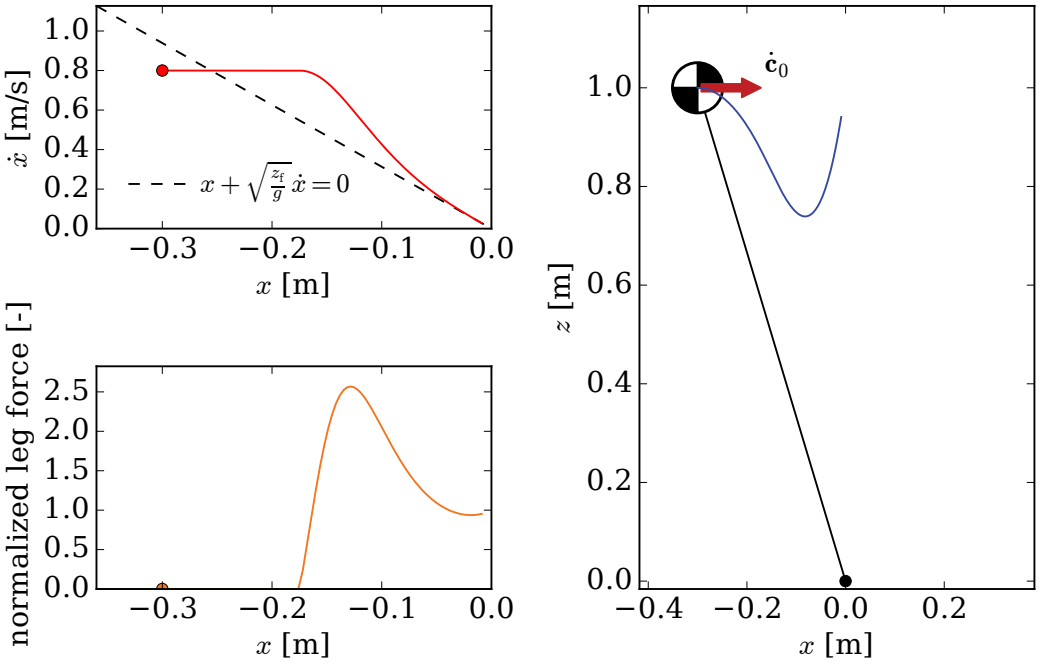


Figure 4-4: Simulation results for clipped controller (4.22) with initial conditions $c_{x,0} = -0.3$ [m], $c_{z,0} = 1$ [m], $\dot{c}_{x,0} = 0.8$ [m/s], and $\dot{c}_{z,0} = 0$ [m/s]. For this plot, $g_z = 9.8$ [m/s^2], $c_{z,\mathrm{f}} = 1$ [m].

Fig. 4-4 shows that this clipped controller successfully achieves balance from the same state that resulted in pulling on the ground with controller (4.20).

In addition to the limitation of unilateral contact, there are two other issues that require more attention compared to the LIP:

1. the height trajectory may not be kinematically feasible due to robot geometry and joint limits;

2. actuation limits may be violated.

Barring unilaterality of contact, kinematics, and actuation limits, it would be possible to achieve balance from any state. In this chapter, we choose not to address kinematics or actuation limits. Instead, we focus on the fundamental implications of unilateral contact.

4.6 Region of attraction

We now derive the regions of attraction of the orbital energy controller (4.20) (in Section 4.6.1) and the clipped controller (4.22) (in Section 4.6.2). These regions of attraction are also inner approximations of the set of initial states from which balance can be achieved. The inner approximation for the clipped control law will turn out to be the same as the outer approximation derived in Section 4.3.

4.6.1 Orbital energy controller

For the orbital energy controller (4.20), we will consider passing through a state for which the controller outputs $u < 0$ at any time to be a failure. Observe that if (4.5) holds, then by construction of f, c_x will converge to 0, so c_x is between $c_{x,0}$ and 0 for all time. Hence, requiring that $u \geq 0$ for all time is the same as requiring (4.5) and

$$U\left(\delta c_{x,0}, x_0\right) \geq 0 \quad \forall \delta \in [0,1] \tag{4.23}$$

The task is now to find an explicit description of the set of initial conditions that

satisfy (4.23). Condition (4.23) can be written equivalently as

$$p\left(\delta c_{x,0}, x_0\right) q\left(\delta c_{x,0}, x_0\right) \geq 0 \quad \forall \delta \in [0,1]\,. \tag{4.24}$$

with p and q as defined by (4.19). The conditions (4.5) and (4.24) together form a *first-order formula* over the reals in the variables $x_0, \delta, g_z, c_{z,\mathrm{f}}$ [94]. For completeness, we should add that $g_z > 0$ and $c_{z,\mathrm{f}} > 0$.

In this context, a first-order formula in variables $y_1, \ldots, y_n$ is an expression written by combining a set of polynomial equations and inequalities in $y_1, \ldots, y_n$ using the logical conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$) operators, while some or all of the variables are quantified over by universal and/or existential quantifiers (*e.g.*, $\forall y_1$, $\exists y_2$). Variables that are not quantified over are called free. For any first-order formula over the reals, there is an equivalent *quantifier-free* formula, *i.e.*, a formula without any universal or existential quantifiers, by the famous Tarski-Seidenberg theorem [95]. The process of finding an equivalent quantifier-free formula is known as quantifier elimination.

Example 2 *[94] Quantifier elimination can be applied to the first-order formula*

$$a \neq 0 \wedge \left(\exists x \text{ such that } ax^2 + bx + c = 0\right)$$

in variables a, b, c, x (with a, b, c free) to find the familiar equivalent quantifier-free formula $b^2 - 4ac \geq 0$. ▲

Cylindrical Algebraic Decomposition (CAD) methods can be used to solve quantifier elimination problems [94]. The worst case running time of modern CAD-based algorithms is polynomial in the number of polynomials in the input formula and their degree, but exponential in the number of free variables [94]. Implementations include QEPCAD B [96] and Mathematica's `CylindricalDecomposition` function [97].

We used Mathematica's implementation to find a quantifier-free formula that is equivalent to the conjunction of (4.24), (4.5), $g_z > 0$ and $c_{z,\mathrm{f}} > 0$. It should be noted that applying the CAD algorithm directly took too long, but using the variable

substitutions

$$a_0 = \frac{\dot{c}_{x,0}}{c_{x,0}} \qquad\qquad b_0 = \dot{c}_{z,0} - a_0 c_{z,0}$$

analogous to (4.21), we were able to reduce the number of free variables in (4.23) from six (x_0, g_z, and $c_{z,\mathrm{f}}$) to four (a_0, b_0, g_z, and $c_{z,\mathrm{f}}$), which made the problem tractable. After conversion to a first-order formula (analogous to the step from (4.23) to (4.24)), the CAD algorithm was able to solve the problem in less than two seconds. The result (after simplification) is that initial states must satisfy

$$a_0 < 0 \wedge 7g_z + 20a_0 b_0 + \sqrt{9g_z^2 + 120a_0^2 g_z z_{\mathrm{f}}} \leq 0. \tag{4.25}$$

See Fig. 4-5 for a 3D slice of this region at $\dot{c}_{z,0} = 0$, in terms of $c_{x,0}$, $\dot{c}_{x,0}$, and $c_{z,0}$. See Fig. 4-6 for a 2D slice at $\dot{c}_{z,0} = 0$ and $c_{z,0} = c_{z,\mathrm{f}}$, as well as a comparison to the region for the LIP with fixed point foot, a line defined by the instantaneous capture point, and the necessary condition (4.7).

For $\dot{c}_{z,0} = 0$ and fixed $c_{z,0}$, Figs. 4-5 and 4-6 show that balance can be achieved from a double cone in the $(c_{x,0}, \dot{c}_{x,0})$ plane, with a nappe in each quadrant where $c_{x,0}$ and $\dot{c}_{x,0}$ have opposite sign. The double cone geometry is due to the fact that in (4.25), $c_{x,0}$ and $\dot{c}_{x,0}$ only appear as the ratio $a_0 = \frac{\dot{c}_{x,0}}{c_{x,0}}$. Informally speaking, the 'size' of the double cone increases as $c_{z,0}$ increases, as can be expected from intuition. Increasing $\dot{c}_{z,0}$ also grows the double cone (not shown in figures). If $\dot{c}_{z,0} > 0$, there exist (unrealistic) states with $c_{z,0} < 0$ for which balance can be achieved.

Figs. 4-5 and 4-6 suggest (for $\dot{c}_z = 0$) that if balance can be achieved from $(c_{x,0}, c_{z,0}, \dot{c}_{x,0}, \dot{c}_{z,0})$, then balance can also be achieved from $(c_{x,0}, c_{z,0}, c\dot{c}_{x,0}, \dot{c}_{z,0})$, for any $c \geq 1$. In other words, dilating the initial horizontal velocity can never compromise the ability to achieve balance. Indeed, we were able to prove this dilation property for any $\dot{c}_{z,0}$ using Mathematica functions, including the same CAD techniques described earlier, as long as $c_{z,0} > 0$.[3] This implies that for fixed $c_{x,0}$, $\dot{c}_{z,0}$, and $c_{z,0} > 0$, the

[3]For the unrealistic case of $c_{z,0} \leq 0$, a counterexample was found with $a_0 = -1$, $c_{z,0} = -\frac{9}{16}$, $\dot{c}_{z,0} = \frac{13}{8}$, $c = 2$, $g_z = 1$, and $c_{z,\mathrm{f}} = 1$.

constraint $u \geq 0$ and the restriction to cubic CoM height trajectories only impose a *lower* limit on the initial velocity at which the CoM approaches $c_x = 0$, and no upper limit. An upper limit could come from the robot's kinematics or actuation limits, not investigated in this chapter. Indeed, for high initial velocities, CoM height can become unrealistically high.

The double cone in Fig. 4-6 can be compared to the line defined by the instantaneous capture point for the LIP, (4.15). The most interesting comparison is between the LIP line and the line associated with the lower limit on horizontal velocity for the variable-height model, for the case $c_{z,0} = c_{z,\mathrm{f}}$. The slope of this line can be found by maximizing a_0 subject to (4.25). For $c_{z,0} = c_{z,\mathrm{f}}$ and $\dot{c}_{z,0} = 0$, the optimal value can be found in closed form:

$$a_0 = \frac{\dot{c}_{x,0}}{c_{x,0}} \leq -\sqrt{\frac{5+\sqrt{15}}{10}}\sqrt{\frac{g_z}{c_{z,0}}} \approx -0.94\sqrt{\frac{g_z}{c_{z,0}}}. \tag{4.26}$$

We can compare this to the LIP by rewriting (4.15) as

$$\frac{\dot{c}_{x,0}}{c_{x,0}} = -\sqrt{\frac{g_z}{c_{z,0}}}$$

which shows that for a given value of $c_{x,0}$, the orbital energy controller can achieve balance from states that have an initial horizontal velocity up to about 6% slower compared to the LIP.

We can also compare this lower velocity limit to the necessary condition derived from the ballistic trajectory in Section 4.3. With $\dot{c}_{z,0} = 0$, condition (4.7) evaluated at $x = x_0$ can be rearranged to find

$$\frac{\dot{c}_{x,0}}{c_{x,0}} < -\frac{1}{\sqrt{2}}\sqrt{\frac{g_z}{c_{z,0}}} \approx -0.71\sqrt{\frac{g_z}{c_{z,0}}} \tag{4.27}$$

implying that for fixed $c_{x,0}$, it is *impossible* to achieve balance from initial horizontal velocities that are more than about 39% lower than the balancing velocity for the LIP.

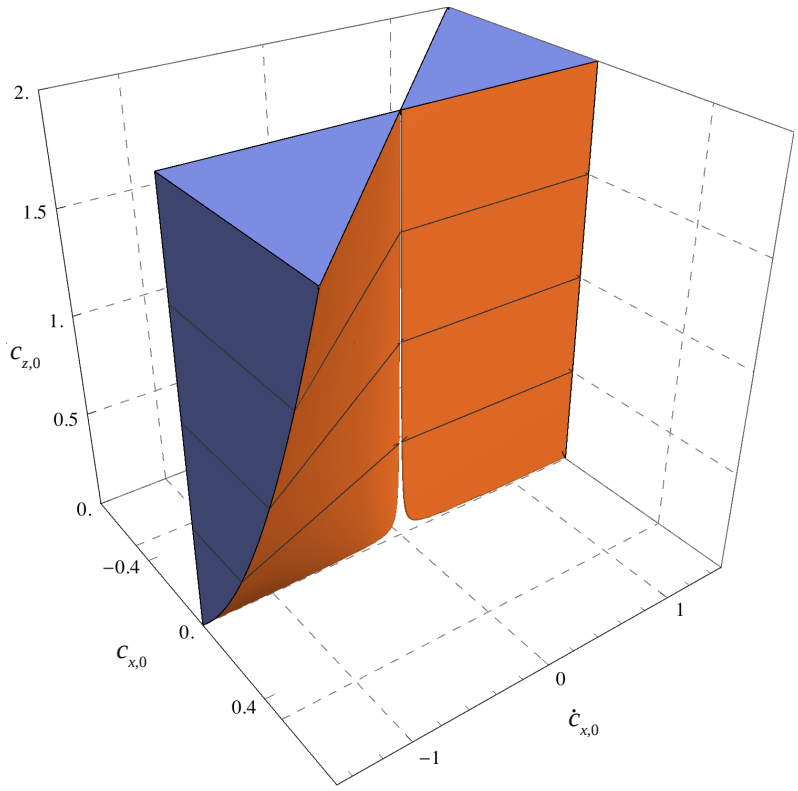


Figure 4-5: Slice at $\dot{c}_{z,0} = 0$ of the set of states from which balance can be achieved. Note that the apparent separation between the regions on opposite sides of the $\dot{c}_{x,0}$-axis is merely a plotting artifact. For this plot, $g_z = 9.8$ [m/s^2], $c_{z,\mathrm{f}} = 1$ [m]. The full region extends outside the borders of the plot, to infinity, along the blue sections.

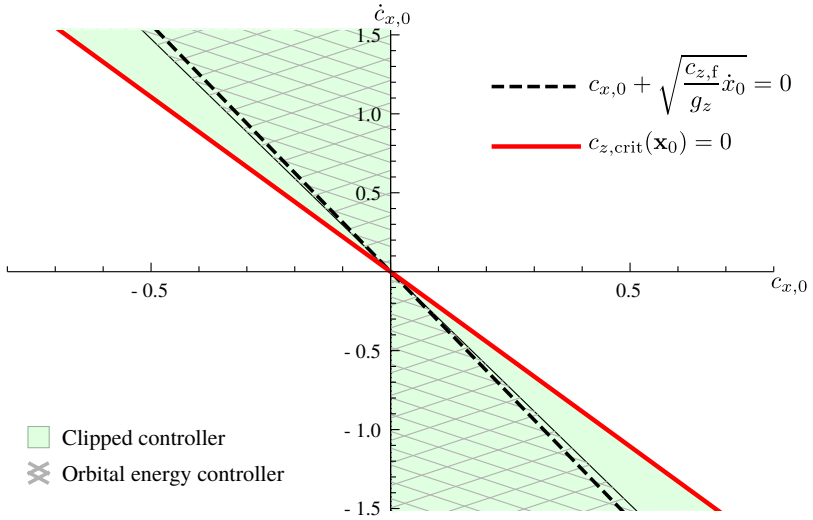


Figure 4-6: Set of states from which balance can be achieved; comparison between LIP (region defined by (4.15), the instantaneous capture point), orbital energy controller (region defined by (4.25)), and clipped controller (also corresponding to the necessary condition $c_{z,\mathrm{crit}}(x) > 0$). Slice at $\dot{c}_{z,0} = 0$ and $c_{z,0} = c_{z,\mathrm{f}}$ for the variable-height inverted pendulum. For this plot, $g_z = 9.8$ [m/s^2], $c_{z,\mathrm{f}} = 1$ [m].

4.6.2 Clipped controller

We now use the results for the orbital energy controller to derive the region of attraction of the clipped controller.

Note that if the orbital energy controller produces $u < 0$, then the output of the clipped controller is $u = 0$, meaning that the CoM will follow the ballistic trajectory (4.6). We will show that for any state satisfying necessary condition (4.7), following the ballistic trajectory will eventually bring the state in the region of attraction of the orbital energy controller.

For states satisfying (4.5), following the ballistic trajectory means that $\dot{c}_x$ remains constant and $|c_x|$ decreases. This implies that the ratio $a = \frac{\dot{c}_x}{c_x}$ approaches $-\infty$ as c_x approaches 0. We will thus examine what happens to the conditions (4.25), describing the region of attraction of the orbital energy controller, as a_0 approaches $-\infty$. The first condition, $a_0 < 0$, will certainly be satisfied. The second condition will be satisfied as long as $c_{z,0} > 0$:

$$\lim_{a_0 \to -\infty} 7g_z + 20a_0\dot{c}_{z,0} - 20a_0^2 c_{z,0} + \sqrt{9g_z^2 + 120a_0^2 g_z z_\mathrm{f}} = -\infty c_{z,0} \leq 0$$

since the quadratic term dominates. Here, we have substituted the definition $b_0 = \dot{c}_{z,0} - a_0 c_{z,0}$ into (4.25).

We now note that if necessary condition (4.7) is satisfied, then following the ballistic trajectory will make $c_z > 0$ in some open interval around $c_x = 0$, so $c_{z,0} > 0$ will indeed be satisfied as $a_0 \to -\infty$. We thus conclude that the region of attraction of (4.22) matches the outer approximation given by (4.7): if the ballistic trajectory starting from the current state has a c_z-intercept greater than zero, then the clipped controller (4.22) will make the state converge to the fixed point at $c_z = c_{z,\mathrm{f}}$.

4.7 Discussion

A limitation of the presented control approach is that, similar to typical LIP-based approaches, kinematic limits and actuator limits are not taken into account. This

poses more of a problem for the presented model than for the LIP. Indeed, using control laws (4.20) or (4.22), ground reaction forces and CoM heights may become very large for high initial horizontal velocities, *i.e.* with a large value of $-\frac{\dot{c}_{x,0}}{c_{x,0}}$. For the clipped controller, similar issues occur at very low initial horizontal velocities, which result in entering the region of attraction of the orbital energy controller at a very large value of $-\frac{\dot{c}_{x,0}}{c_{x,0}}$, in turn resulting in extreme control actions. More work is needed to study the implications of kinematics and actuation limits on balance achievement using CoM height control.

We based our orbital energy control law on a cubic CoM height trajectory. All of the techniques used in this chapter still apply with higher order polynomials. However, the limiting factor in our analysis is the computational cost of quantifier elimination. For example, with a quartic CoM height trajectory and the additional constraint that the second derivative of the trajectory be zero at $c_x = c_{x,0}$, the CAD algorithm ran for several hours before we decided to terminate it. This is interesting, because the main determinant of worst-case computational cost, the number of free variables, remains the same[4]. It could be the case that the cubic trajectory has special properties that result in a drastically easier quantifier elimination problem. It is also interesting that the clipped version of the controller corresponding to the cubic trajectory, (4.22), is already able to achieve balance from any state from which balance can possibly be achieved.

The presented model and approach do not explicitly take friction cone constraints into account. However, the angle between the ground reaction force vector f_{gr} and the vertical axis is the same as the angle between the CoM position, c, and the vertical axis. It can be shown that if balance is achievable from initial state x_0 according to (4.25) then the greatest angle between c and the vertical axis that occurs while using either controller is at x_0.

With these caveats in mind, the analysis presented in this chapter can still provide insight into the relative importance of CoM height control and other balancing mech-

[4]The change of variables (4.21) still results in the same reduction in the number of variables for this quartic polynomial.

anisms. Since the clipped orbital energy controller (4.22) achieves balance from any state that satisfies necessary condition (4.7), (4.27) applies: the controller achieves balance from initial horizontal CoM velocities that are up to 39% lower than the horizontal velocity that results in balance for the LIP, a significant difference. In the presence of kinematic constraints and actuation limits, this provides a useful upper bound on the effectiveness of CoM height control. An interesting property of CoM height control is that its effectiveness, quantified in terms of a lower limit on horizontal CoM velocity, increases with horizontal displacement between the CoM and the point foot, as shown by (4.27). This is in contrast to stepping, CoP control, and control of centroidal angular momentum, for which effectiveness does not depend on horizontal CoM position at a fixed horizontal CoM velocity [1]. For $c_{x,0} = -0.4$ [m], $c_{z,0} = c_{z,\mathrm{f}} = 1$ [m], $g_z = 9.8$ [m/s^2], and $\dot{c}_{z,0} = 0$ [m/s], CoM height control can be used to achieve balance from the same lower limit on $\dot{c}_{x,0}$ as could be achieved with a 2.3 [cm] change in CoP for the orbital energy controller, and an 11.7 [cm] change for the clipped controller (executing a ballistic trajectory).

Assessing to what degree CoM height control is used for balance in humans is an interesting topic. Long jump athletes that land on their feet can be observed to keep their CoM height low during the landing phase to avoid falling backwards. Humans seem to increase their CoM height when required to stop abruptly without taking additional steps. It would be interesting to know to what degree humans vary CoM height (with respect to a nominal trajectory) in order to regain balance during more periodic motions, such as walking and running.

We note that extending our results to a 3D model is not trivial, contrary to the LIP. Typically, virtual constraint approaches shine when the degree of underactuation[5] is one. A possible direction of future research is to include *e.g.* lateral center of pressure control as an additional input in a 3D model, so as to maintain one degree of underactuation. Another possible approach is to use decoupled lateral and sagittal plane controllers [98]. In general, combining CoM height control with other stabilizing mechanisms is a topic of future research.

[5]The number of degrees of freedom minus the number of inputs.

We employed quantifier elimination techniques. While these techniques can be extremely powerful, they do not scale well. In addition, the end results are not always as clean as condition (4.25). Nevertheless, there may be other problems in legged locomotion that could benefit from the application of these techniques.

4.8 Conclusion

This chapter investigated the use of CoM height variation to achieve balance, subject to unilateral contact. For a 2D variable-height inverted pendulum model, we derived an outer approximation of the set of states from which balance is achievable. We presented a controller based on orbital energy, and derived its region of attraction. We also studied a clipped version of this controller, and showed that it achieves balance from any state for which balance is physically possible to achieve.

Chapter 5

Multi-contact centroidal trajectory optimization as a mixed-integer nonlinear program

5.1 Introduction

The previous chapter presented an approach that breaks away from the linear inverted pendulum's limiting assumption of planar center of mass (CoM) motion, and results in a control law with trivial computational cost and interesting theoretical guarantees. However, as alluded to in section 4.7, the approach has some severe limitations when it comes to using the results to achieve real robot locomotion in a rich contact environment. The online control law was designed based on a 2D centroidal model without contact changes, kinematic constraints, contact force constraints, or even variation of the center of pressure (CoP). In contrast, this chapter presents an offline planning approach based on a 3D multi-contact centroidal model that allows for (approximate) kinematic constraints as well as contact force constraints, including both Coulomb friction and upper limits on magnitude.

The main goal of the presented planning approach is to provide reference trajectories to a whole-body tracking controller [80, 99, 57, 100], allowing a high-degree-of-

freedom (high-DoF) humanoid robot to navigate its environment. In particular, this work targets the momentum-based control approach presented in chapter 3, which requires a reference trajectory for the robot's total linear momentum (or its CoM), as well as a footstep plan, *i.e.*, a timed sequence of contacts, including reference positions for the robot's extremities when contact is first established.

The problem of optimizing over locomotion plans that allow a high-DoF humanoid robot to traverse complex terrain is hard for several reasons, chief among which are the following:

- Trajectory optimization problems based on detailed models of high-DoF robots translate into nonlinear optimization problems with many decision variables and constraints.

- Even centroidal models are still subject to bilinear constraints, stemming from joint optimization over contact positions and contact forces and the desire to limit centroidal angular momentum rate (see section 2.3).

- Modeling and optimizing over contact changes requires special care due to abrupt changes in dynamics (discussed in more detail in section 5.3.1).

- The value of locomotion plans increases dramatically if the plans can be found quickly. User-interactive rates are desirable, but ideally, good plans would be generated at sufficiently high rates to allow embedding in a model-predictive control (MPC) approach.

Several recent studies have focused on this challenging problem. On the fast-but-limited end of the spectrum, a simple extension of the LIP was employed in [101] to plan locomotion over terrain consisting of moderate-size rolling hills based on the divergent component of motion and a so-called enhanced centroidal moment pivot (essentially a slightly modified CoP/ZMP). However, this approach cannot guarantee that the contact forces associated with the planned trajectory are actually achievable, and doesn't properly handle the case of multiple non-coplanar contacts. Bretl and Lall point out some of these issues in the simpler context of static equilibrium [102].

Furthermore, contact sequence timing and footstep placement are prespecified, which vastly simplifies the problem but neglects an important part of it. Such cascaded approaches are numerous in the literature.

Moving towards slower but more capable approaches, [103] propose a planner that finds a centroidal momentum trajectory on moderately rough terrain given a contact sequence by reasoning about the set of achievable contact wrenches (contact wrench cone, or CWC) and using a convex outer approximation of the rate of change of centroidal angular momentum to avoid the bilinearities described in section 2.3. [104] also propose a centroidal planning algorithm given a contact sequence based on properties of the CWC, but using a dual formulation. Convex outer approximations of the bilinear angular momentum rate constraints are also used in [105]. While these approaches can handle the multi-contact case properly, a possible issue is that they allow constraints on angular momentum rate to be violated, sometimes to a significant degree. Moreover, these approaches still rely on a prespecified contact sequence. Recent approaches can separately find reasonable contact sequences for locomotion over rough terrain using heuristics and optimization [106, 107], but for truly dynamic locomotion over such terrain, it would be preferable to explicitly include the robot's dynamics in the contact planning problem, at least at the centroidal level.

At the other extreme of the slow-versus-capable spectrum, off-line nonlinear optimization techniques have been employed in *e.g.* [108, 109] based on the full dynamics of the robot. Intermediate solutions using a detailed kinematic model but a centroidal dynamics model were used in [32, 110] and showed impressive results in simulation, but these approaches still have a rather high computational cost. Remarkably, approaches based on nonlinear optimization have recently been extended into the MPC domain at impressively large scale. Such methods include differential dynamic programming (DDP) and its variants, particularly the iterative Linear Quadratic Regulator approach (iLQR) [111, 112, 113, 114]. However, nonlinear MPC at the scale of a humanoid robot has not yet been demonstrated, and will require considerable engineering effort [115, 116, 117]. Additionally, there are likely to be significant issues with local minima, and convergence to a solution, especially within the time budget

needed for online control, is not guaranteed.

A recent line of research has proposed avoiding the drawbacks of the cascaded approach and the convergence problems associated with gradient-based optimization by formulating trajectory optimization problems involving contact as mixed-integer programs. These approaches explicitly optimize over contact mode sequences, while also taking constraints stemming from the continuous-time dynamics into account.

Using the mixed-integer programming formalism has several advantages. First, integer decision variables (specifically, binary variables) are a natural fit to model the on-off behavior of contact between the robot and its environment. Second, typical solvers do not require a starting point to arrive at a solution (though a good starting point may improve solver performance). Third, standard mixed-integer solvers maintain both upper and lower bounds on the optimal objective function value during the solution process, and can thereby provide a certificate of global optimality when these bounds match. While mixed-integer optimization is NP-complete in the number of integer variables, specialized solvers have been developed that typically do much better in practice than the theoretical worst case.

In [118], mixed-integer optimization was applied to plan footsteps over rough terrain. An early mixed-integer-based trajectory optimization approach is [119], where mixed-integer quadratic programming (MIQP) was applied to a trajectory optimization problem stemming from a linear centroidal model of a bipedal robot, and solved approximately in an MPC setting. In [31], the author applies trajectory optimization based on MIQP to a 2D model of the Boston Dynamics LittleDog robot, jumping between staggered platforms. The model includes Coulomb friction constraints, and the approach uses an off-the-shelf MIQP solver. In [120], similar ideas are applied to a 3D LittleDog model, while employing a contact wrench cone margin as a robustness metric. In [121], MIQP trajectory optimization is applied to a planar multi-link humanoid model. This approach, called LVIS (Learning from Value function IntervalS), partially solves a large number of these trajectory optimization problems off-line, and uses the bounds on the optimal objective value provided by the solver to approximate the value function of the optimal control problem over a region of the state space.

The on-line control problem then reduces to a simpler, one-step mixed-integer MPC problem with the objective of minimizing the approximated value function value one time step in the future.

While these approaches produce encouraging results, they share a common drawback, rooted in the use of mixed-integer quadratic programming: the underlying dynamics model must be piecewise affine in order to fit into the MIQP framework. This means that the bilinear dependence of angular momentum rate of change on contact positions and contact forces (see section 2.3) cannot be modeled natively, which may result in non-physical trajectory optimization results.

MIQP's lacking expressivity in the face of bilinear constraints can be sidestepped to a certain degree through the use of piecewise McCormick envelopes [31] (discussed in more detail in section 5.3.5). However, piecewise McCormick envelopes require a number of integer variables, used to select between the pieces, that is at least logarithmic in the number of pieces (see e.g. [122]), and a fairly high number of pieces is required to achieve sufficiently low violation of the original bilinear constraints over a large region. This is problematic, as the number of integer variables is strongly correlated with solve time. The use of piecewise McCormick envelopes in planning approaches for humanoid robots is particularly problematic: for humanoids, the base of support is typically small compared to that of the quadruped robots used in the experiments of [31] and [120], reducing the margin of error for angular momentum constraint violation.

In this work, we 'bite the bullet' by switching away from the use of MIQP solvers and associated techniques used to fit nonlinear problems into the MIQP framework.[1] Instead, we embrace the nonlinearity and adopt the use of dedicated mixed-integer nonlinear programming (MINLP) tools, allowing us to enforce angular momentum rate constraints up to a prespecified, arbitrarily tight tolerance.

Dedicated MINLP solvers, such as SCIP [123], BARON [124], and COUENNE

[1]We note that [31] also explored using an MINLP formulation, and reported computational performance many times worse than the MIQP relaxation. However, these results were just for one possible problem formulation and were obtained using the COUENNE solver, which is known to be less performant than *e.g.* SCIP and BARON.

[125], internally use specialized methods for dealing with bilinear constraints, including the McCormick envelope approach used in [31], but employ them in an adaptive fashion. This leads to a better tradeoff between accuracy and computation time. In addition, solvers such as SCIP employ an array of primal heuristics to quickly find good feasible points, even in the absence of a good starting point. These primal heuristics subsume and extend many approaches used to solve bilinear optimization problems in the controls literature, such as bilinear alternations and relaxed complementarity reformulations (MPEC, mathematical program with equilibrium constraints) [126].

Another problem with the state of the art in mixed-integer trajectory optimization is the use of a direct transcription in conjunction with basic Euler integration and fixed time steps. Since the number of integer variables is proportional to the number of time steps, and solve times in practice heavily depend on the number of integer variables, relatively few time steps and large time intervals are commonly employed. For example, [121] used a mere 10 knot points with 50 ms time intervals for the simplified 2D humanoid model. At this point, integration accuracy can become a real issue. In addition, the formulations used in [31, 121] allow every end effector to make or break contact with the environment at every time step. It seems unlikely that the optimal mode sequence involves switching contacts at every 50 ms (or less, if we want to improve integration accuracy), and so it seems wasteful and unnatural to use so many integer variables, searching over the associated astronomical numbers of possible mode sequences.

In addition to the shift from MIQP to MINLP, we further propose to address some of these issues in previous approaches with the following contributions:

- Using a piecewise polynomial representation of the CoM trajectory and contact forces, which removes any concerns related to integration accuracy, and means that longer time steps can be used.

- In particular, using a piecewise *Bézier* representation (Bernstein polynomial basis) for the polynomial pieces, as this allows constraints on the Bézier control

points to imply constraints on the entire Bézier piece due to the Bézier convex hull property (see section 5.4.3), avoiding issues with constraint violation between knot points.

- Associating a single integer contact variable with an entire piece of the piecewise polynomial (corresponding to a longer time interval), thus requiring fewer integer variables. This reduces the tension between integration accuracy and number of integer variables.

- Reverting to a centroidal model, as opposed to the multi-link model used in [121], to make a 3D version feasible. Limitations due to kinematics and actuation are modeled approximately, by placing constraints on relative positions of contacting bodies and the CoM and on contact force magnitudes respectively.

- Using variable time steps, which will result in additional bilinearities (as state variables and inputs are multiplied by the time steps), but appears to reduce the time needed to find a feasible solution in practice.

We present example results of the proposed approach, including dynamic simulations of the Atlas robot under the controller from chapter 3.

It should be noted that the goal of this work is not to outperform existing approaches in terms of computation time, but rather to explore in a new direction and present our experience with a set of tools that is relatively unfamiliar to the robotics community. Furthermore, we note that approaches like LVIS [121] could be used to obtain an online policy by using offline trajectory optimization to sample from the optimal policy. This is in line with the recent popularity of machine learning methods in general, in that a large amount of offline computation may be used to arrive at a result that allows for fast online evaluation. This alleviates concerns regarding the cost of offline computation.

The remainder of this chapter is structured as follows. Section 5.2 describes the trajectory optimization problem we wish to solve at a high level. Section 5.3 gives some background on mixed-integer programming. Section 5.4 details the reformulation of the trajectory optimization problem as a mixed-integer nonlinear program.

Section 5.5 discusses how the results of the trajectory optimization are used to generate inputs for the whole-body controller. Section 5.6 describes implementation details. Section 5.7 presents results for a few example problems. Section 5.8 discusses these results, and we conclude the chapter in section 5.9.

5.2 Problem statement

This section establishes notation and provides a high-level conceptual description of the planning problem we wish to solve. This problem statement is meant to be independent of specific solution techniques, and does not require an understanding of mixed-integer programming. However, some of the modeling choices made in this section are indeed guided by the desire to effectively reformulate the problem as an MINLP, as will be done in section 5.4.

Throughout this chapter, we will use $[\![n]\!]$ to denote the set of integers $\{1, 2, \ldots, n\}$.

5.2.1 High-level description and desired output

We consider a robot with n_c *contact bodies, i.e.*, links that may be used to exert forces on the contact environment.[2] The contact environment consists of n_e planar *environment regions* at various positions and orientations, fixed in world frame Φ_w. The initial position and velocity of the CoM are specified, and the initial angular momentum about the CoM is zero. See Fig. 5-1 for an example scenario.

At a high level, the goal is to find a plan that may be used as an input to a momentum-based tracking controller [57], resulting in locomotion over the terrain. The plan should consist of:

- A timed contact sequence, *i.e.*, the assignment of contact bodies to available environment regions as a function of time. Timing may either be prespecified or left free.

[2]Contact bodies are sometimes referred to as end effectors in related approaches [31, 120], and are typically the extremities of a humanoid robot.

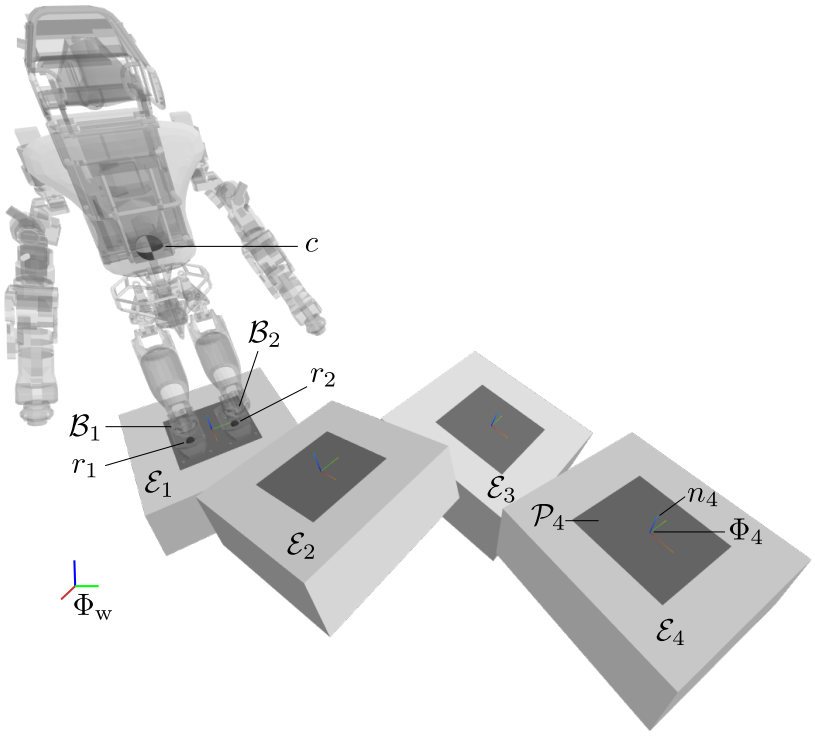


Figure 5-1: Example scenario with four environment regions and two contact bodies. The light grey regions represent the actual contact geometry used for dynamic simulation. The dark grey regions are the polyhedra $\{\mathcal{P}_i\}_{i\in[\![n_{\mathrm{e}}]\!]}$, used for the trajectory optimization. The latter are essentially configuration space regions for the contact reference points, obtained by shrinking the top surfaces of the former.

- Contact locations: the 3D position of a reference point on each contact body when it first comes into contact with an environment region.
- Trajectories for the center of mass, contact forces, and centers of pressure for each contact body. Contact force trajectories should be within static friction cones (no slip).

The following sections will describe various aspects of the planning problem in more detail.

5.2.2 Environment regions

The contact environment consists of a finite set of world-fixed planar environment regions, $\{\mathcal{E}_k\}_{k\in[\![n_e]\!]}$, where region k is described by:

- $T_k^{\mathrm{w}} \in SE(3)$, a rigid transformation from region-local coordinates to world coordinates, which defines the region-local reference frame Φ_k with respect to Φ_{w}, such that the unit vector in the z-direction of Φ_k is the region's contact normal, n_k.
- $\tilde{\mathcal{P}}_k \subset \mathbb{R}^2$, a bounded polyhedron that describes the extents of the environment region as a subset of the x-y plane of frame Φ_k.
- $\mu_k \geq 0$, the coefficient of friction.

The contact normal n_k and friction coefficient μ_k together parameterize the region's friction cone, $\mathcal{F}_k$, defined as

$$\mathcal{F}_k = \{f \mid \|f - (n_k \cdot f)\, n_k\| \leq \mu_k n_k \cdot f\}.$$

We denote set of contact positions *in world coordinates* associated with $\mathcal{E}_k$ (a polyhedron in $\mathbb{R}^3$) as

$$\mathcal{P}_k = \left\{T_k^{\mathrm{w}}\left((x, y, 0)\right) \mid (x, y) \in \tilde{\mathcal{P}}_k\right\},$$

where $T_k^{\mathrm{w}}(\cdot)$ denotes the application of the rigid transformation from Φ_k to Φ_{w}.

5.2.3 Contact bodies

The robot's contact bodies (*e.g.*, hands and feet) are denoted $\{\mathcal{C}_j\}_{j\in[\![n_c]\!]}$. Contact body $\mathcal{C}_j$ is has a planar contact region $\mathcal{S}_j$ (for a foot, the sole), which acts as the interface between the body and any environment region with which it may be in contact.

The configuration of $\mathcal{C}_j$ is summarized by the position of a single body-fixed contact reference point, $r_j \in \mathcal{S}_j$. The orientation of the contact body is assumed to conform to the environment region to which it is assigned. Whenever body $\mathcal{C}_j$ is not in contact with any environment region (*i.e.*, during swing phases), we consider the world-frame location of r_j to be immaterial for the purposes of the proposed planning approach. The exact trajectory of a contact body during swing is considered an implementation detail that will be filled in as part of the on-line tracking controller.[3] However, if body $\mathcal{C}_j$ is in contact with region $\mathcal{E}_k$, we require that

$$r_j \in \mathcal{P}_k.$$

Contact body $\mathcal{C}_j$ has an associated center of pressure at position p_j. Similar to r_j, p_j is only of consequence when $\mathcal{C}_j$ is in contact with one of the environment regions. When body $\mathcal{C}_j$ is in contact with environment region $\mathcal{E}_k$, CoP p_j must lie on the x-y plane of frame Φ_k. Furthermore, an upper bound is specified on the distance between the contact reference point r_j and the CoP p_j, denoted $\overline{d}_{p,j}$. This maximum distance roughly models the extents of the body's contact surface, and should be chosen such that the disk

$$\left\{p_j \mid \|p_j - r_j\| \leq \overline{d}_{p,j}\right\} \cap \operatorname{aff} \mathcal{S}_j \tag{5.1}$$

is fully contained in the planar contact region $\mathcal{S}_j$ of contact body $\mathcal{C}_j$, where $\operatorname{aff} \mathcal{S}_j$ denotes the affine hull of $\mathcal{S}_j$. This ensures that any CoP position that satisfies

[3] We note that it would also be possible to formulate a problem with an explicit partitioning of the free space as well as the environment [31], which may enable better collision avoidance. However, we have chosen not to do so here to simplify the problem.

$\|p_j - r_j\| \leq \overline{d}_{p,j}$ is actually achievable in the full robot model,[4] while avoiding the need for increased problem complexity associated with modeling foot orientation.

The contact force $f_{\mathrm{c},j}$ exerted upon body $\mathcal{C}_j$ acts at CoP p_j, and must be within the friction cone $\mathcal{F}_k$ whenever $\mathcal{C}_j$ is in contact with region $\mathcal{E}_k$. Contact forces are additionally limited by an upper bound on normal force,

$$n_k \cdot f_{\mathrm{c},j} \leq \overline{f}_{\mathrm{c}}.$$

When $\mathcal{C}_j$ is not in contact with any environment region, we require that $f_{\mathrm{c},j} = 0$.

5.2.4 Dynamics

When the center of mass, c, is expressed in an inertial frame of reference such as Φ_{w}, its evolution is described by (2.11), *i.e.*:

$$\ddot{c} = mg + \sum_{j=1}^{n_{\mathrm{c}}} f_{\mathrm{c},j} \tag{5.2}$$

where m is the robot's total mass and g is the gravitational acceleration vector. For simplicity, we do not consider impact events in this problem statement. The resulting lack of impulsive forces implies that the CoM trajectory is C^1-continuous.

To disallow the use of the robot's rotational inertia as a flywheel that stores angular momentum, as motivated in section 2.3, we require the rate of change of angular momentum about the robot's center of mass, $\dot{k}$, to remain zero at all times:

$$\dot{k} = \sum_{j=1}^{n_{\mathrm{c}}} (p_j - c) \times f_{\mathrm{c},j} = 0,$$

or equivalently,

$$\sum_{j=1}^{n_{\mathrm{c}}} p_j \times f_{\mathrm{c},j} = c \times \sum_{j=1}^{n_{\mathrm{c}}} f_{\mathrm{c},j} \tag{5.3}$$

[4]The environment region polyhedra, $\left\{\tilde{\mathcal{P}}_k\right\}_{k \in [\![n_{\mathrm{e}}]\!]}$, should also be chosen so that if a contact reference point r_j is in $\mathcal{P}_k$, the body's planar contact region $\mathcal{S}_j$ can fully fit inside the actual contact geometry in any orientation. That is, the actual contact geometry should be shrunk by $\sup_{j \in [\![n_{\mathrm{c}}]\!]} \sup_{x \in \mathcal{S}_j} \|x - r_j\|$.

As noted in section 2.4, this constraint is fundamentally bilinear in the contact forces and their application points relative to the CoM (the CoPs, p_j), making it an important source of both computational complexity and interesting behavior.

5.2.5 Contact sequence constraints

We wish to find a timed contact sequence, *i.e.*, an assignment of contact bodies to available environment regions as a function of time. Valid contact sequences should obey the following basic rules:

- Each contact body can be assigned to at most one environment region at any given time.
- Region reassignment requires an intermediate swing phase: if a contact body is assigned to a given environment region, it must be unassigned for a nonzero amount of time before it can be reassigned to a different environment region.
- Depending on the application, it may be desirable to disallow jumping phases, where none of the contact points are assigned to a region.

Finally, if the contact sequence timing is left free, we additionally specify:

- Minimum and maximum swing duration: if a body goes out of contact at time t, it can only be reassigned to an environment region after $t + \underline{\Delta t}$, and it must be reassigned to a region by $t + \overline{\Delta t}$.
- Minimum stance time: if a body comes into contact with the environment at time t, it must remain assigned to the region until at least $t + \underline{\Delta t}$.

5.2.6 Approximate kinematic constraints

Centroidal trajectory planning approaches achieve reduced problem complexity by summarizing the configuration of the full robot using only the positions of the center of mass and contact force application points. A major drawback of these approaches, in general, is that kinematic constraints are hard to incorporate.

While methods have been proposed that do include the full configuration of the robot while planning a centroidal trajectory [32], we choose not to do so, to reduce problem complexity. Instead, we resort to merely increasing the likelihood that the plan can be executed on the full robot by specifying kinematic constraints on the relative positions of the CoM and the contact reference points.

Ideally, these constraints would be strictly conservative, in the sense that for any allowable configuration of the CoM and contact reference points, there exists a matching configuration of the full robot. Providing such a guarantee is a nontrivial task, however, requiring a complex kinematic reachability analysis that may not be feasible at the scale of a full humanoid robot.[5] Consequently, we opt to employ the following simple heuristics:

- Maximum CoM-to-contact distance: $\|c - r_j\| \leq \overline{d}_{c,j}$ for $j \in [\![n_{\mathrm{c}}]\!]$.
- Minimum CoM-to-contact distance: $\|c - r_j\| \geq \underline{d}_{c,j}$ for $j \in [\![n_{\mathrm{c}}]\!]$.
- Minimum inter-contact distance: $\|r_a - r_b\| \geq \underline{d}_{r,a,b}$ for $a, b \in [\![n_{\mathrm{c}}]\!]$ such that $a \neq b$.
- No cross-over: for select pairs of contact bodies $(\mathcal{C}_a, \mathcal{C}_b)$ (notably, the feet), we require that $e \cdot (r_a - r_b) \geq \underline{d}_{e,a,b}$ for some vector $e \in \mathbb{R}^3$, fixed in world frame Φ_{w}.

Note that the second and third requirements will result in nonconvex constraints. The last requirement implicitly relies on the assumption that body orientation will not change much when the trajectory is tracked on the full robot. This is certainly a limiting assumption, and will be further discussed in section 5.8.

[5] In [30], the authors propose the use of sampling and function approximation to find approximate probability distributions describing the feasibility of CoM positions described in a contact body's coordinate frame. However, this approach cannot be used to formally guarantee kinematic feasibility, and requires making the assumption that the probability of feasibility of the CoM relative to *all* of the contact bodies permits can be factorized into independent feasibility probabilities relative to each of the contact bodies.

5.2.7 Initial and final conditions

The initial position and velocity of the CoM are specified as c_0 and $\dot{c}_0$, respectively. The initial position of the contact reference point for body $\mathcal{C}_j$ is denoted $r_{j,0}$.

Final conditions may be application-specific, but for all of the examples presented in section 5.7, we require that the robot is in static equilibrium at the final time t_f, implying that $\dot{c}\left(t_\mathrm{f}\right) = 0$ and $\ddot{c}\left(t_\mathrm{f}\right) = 0$. Furthermore, we specify a desired final contact situation: at time t_f, each body $\mathcal{C}_j$ is specified to either be in contact with a given environment region $\mathcal{E}_m$, or with none of the environment regions.

5.2.8 Optimization objective

In this study, we focus on finding a feasible point, rather than minimizing an objective function. Feasibility is of course a prerequisite for optimality, and solving a mixed-integer feasibility problem reformulation is already an NP-complete problem. Computational results will also show that finding a feasible point is indeed difficult in practice.

5.3 Mixed-integer programming preliminaries

This section provides a primer on mixed-integer programming formulations and solution techniques. Section 5.3.1 demonstrates by example how mixed-integer formulations can be used to model problems involving rigid contact, and gives a brief comparison to other common formulation styles. Section 5.3.2 gives an overview of mixed-integer convex programming. Section 5.3.3 discusses ways to reformulate constraints involving unions of disjoint sets as mixed-integer convex programs. Such constraints arise naturally from our description of the contact environment as a set of disjoint planar regions. Section 5.3.4 discusses (piecewise) McCormick envelopes, which may be used to approximate bilinear constraints. Finally, section 5.3.5 reviews extensions of mixed-integer convex programming techniques to mixed-integer nonconvex problems.

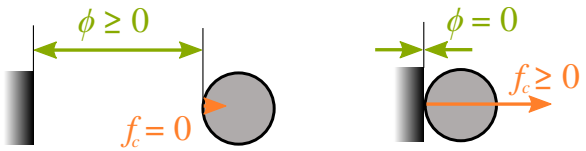


Figure 5-2: 1D contact situation with contact separation ϕ and contact force f_c.

5.3.1 Motivating example

Consider the 1D contact situation shown in Fig. 5-2, involving a moving rigid ball as the contact body and a fixed wall as the contact environment. In this scenario, the contact force $f_c \in \mathbb{R}$ can only be nonzero when the separation $\phi \in \mathbb{R}$ is zero, and vice versa. In a trajectory optimization problem involving contact, both f_c and ϕ may appear as decision variables.

A popular modeling approach for this rigid contact scenario in the context of optimization is to use a complementarity formulation,

$$0 \leq \phi \perp f_c \geq 0$$

which (in the scalar case) is shorthand for

$$\phi \geq 0$$
$$f_c \geq 0$$
$$\phi f_c = 0$$

Complementarity-based formulations have been used successfully in the context of dynamic simulation [127, 128] as well as trajectory optimization [112, 109]. However, without special care, complementarity-based formulations will not, in general, satisfy standard constraint qualifications such as LICQ (linear independence constraint qualifications) [126, 129], violating assumptions of some standard nonlinear program solvers. However, specialized algorithms can be used to avoid this problem to a certain extent.

An alternative approach is to approximate the rigid contact using a soft contact

model. One advantage of this approach is that it resolves paradoxes stemming from static indeterminacy in the presence of multiple contacts. In the context of gradient-based optimization, soft contact models have also been proposed as a solution to the problem of vanishing gradients of contact forces with respect to configuration-related decision variables, by allowing (nonphysical) small contact forces even when separation is nonzero [130, 131]. However, numerical problems can arise from the fact that small displacements cause large contact forces when contact stiffness is high.

If we suppose that f_c and ϕ are upper-bounded by $\overline{f}_\mathrm{c}$ and $\overline{\phi}$ respectively, then a third option is to model the contact situation as:

$$\begin{aligned} &0 \leq \phi \leq (1 - z)\,\overline{\phi} \\ &0 \leq f_\mathrm{c} \leq z\overline{f}_\mathrm{c} \\ &z \in \{0, 1\} \end{aligned}$$

where z is an auxiliary integer (specifically, binary) variable that takes the value 1 when the contact is active. This type of formulation, involving both continuous variables (f_c and ϕ) and integer variables (z) fits within the framework of mixed-integer optimization. The integrality constraint $z \in \{0, 1\}$ is clearly nonconvex, since the midpoint $z = \frac{1}{2}$ between the feasible assignments $z = 0$ and $z = 1$ is infeasible. This third option has not been studied in the context of contact-aware planning for robots until quite recently.

In general, a mixed-integer program can be written in standard form as

$$\begin{aligned} \text{minimize} \quad & f\,(x) \\ \text{subject to} \quad & g_j\,(x) \leq 0, \quad \forall j \\ & x_i \in \mathbb{Z}, \quad \forall i \in I \subseteq [\![\dim x]\!] \end{aligned} \tag{5.4}$$

In a multi-contact trajectory optimization setting, the number of binary variables needed to transcribe the trajectory optimization problem, $|I|$, is at least linear in the number of contact bodies and the number of pieces in the time-discretized trajec-

tory. Furthermore, the complexity of the contact environment is also a factor. We wish to support contact environments consisting of disjoint planar regions, each with their own normal and coefficient of friction. To select between the regions, typical approaches used so far have employed a number of binary variables that is linear in the number of planar regions [31, 120].[6] As a result, there are $2^{n_c n_p n_e}$ possible assignments of the binary variables,[7] with n_c the number of contact bodies, n_p the number of trajectory pieces, and n_e the number of environment regions. For 2 contact bodies, 12 trajectory pieces, and 4 environment regions, that amounts to over $7.9 \cdot 10^{28}$ possible assignments.

Despite this astronomical number of options, relatively effective solution techniques exist for mixed-integer programs, which can in some cases even provide guarantees of global optimality in a reasonable amount of time. The following sections briefly introduce these techniques in the special case of mixed-integer convex programming.

5.3.2 Mixed-integer convex programming

A foundational concept in mixed-integer programming is the continuous relaxation: the optimization problem obtained by dropping all integrality constraints from a mixed-integer program.[8] If the continuous relaxation of a mixed-integer problem is a convex program, the original problem is a *mixed-integer convex program* (MICP).[9] Referring to (5.4), this corresponds to the case where f and the g_j are convex functions.[10] Subclasses include mixed-integer linear and quadratic programs (MILP and MIQP, respectively), similarly referring to the nature of the continuous relaxation. This section provides a very basic overview of popular solution techniques for MICPs.

[6] However, methods exist that require only a logarithmic number of variables (see section 5.3.2).

[7] In this naive analysis, we are disregarding the reduction in number of assignments due to the use of special ordered set constraints.

[8] For binary variables, $z \in \{0, 1\}$ becomes $0 \leq z \leq 1$.

[9] We will use the abbreviation MICP to refer to either mixed-integer convex program or mixed-integer convex programming, depending on context. This also goes for the terms MILP, MIQP, MINLP, and MINCP.

[10] Typically, smoothness of f and the g_j is also required to allow effective solution of continuous relaxations in practice.

For more comprehensive tutorials, surveys, and literature reviews, see *e.g.* [132, 133].

MICP solvers exploit the fact that the continuous relaxation of an MICP is efficiently solvable to global optimality, being a convex optimization. Indeed, in the (generally, unlikely) case that the solution to the continuous relaxation of an MICP happens to also satisfy the integrality constraints, we have found the global optimum of the MICP, since we optimized over a strictly larger search space. Moreover, if the continuous relaxation is infeasible, then the MICP is also infeasible. If we are not so lucky to find ourselves in one of these cases, typical solution techniques rely on solving a sequence of iteratively refined continuous relaxations.

A basic example of such iterative refinement is branching on integer variables. Suppose that $x_i^* \notin \mathbb{Z}$ for some $i \in I$ in the solution x^* to the continuous relaxation of the MICP. Then x_i may be selected for branching, in which case two subproblems are constructed from the parent MICP by adding either $x_i \leq \lfloor x_i^* \rfloor$ or $x_i \geq \lceil x_i^* \rceil$ as constraints, where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ denote rounding down and up to the nearest integer, respectively. These subproblems have two critical properties:

1. they divide the search space, and

2. they both exclude the so-called *fractional solution* x^*.

Continuous relaxations of these subproblems can then be solved (ideally, in parallel), each resulting in new solutions. This approach may be applied recursively, resulting in a search tree with mixed-integer programs and their associated continuous relaxations at the nodes. If the continuous relaxation at one of the nodes is infeasible, the branch rooted at that node may be disregarded, since expanding the node further can only shrink the feasible set. If, at any node, $x_i^* \notin \mathbb{Z}$ for multiple $i \in I$, heuristic *branching rules* are used to select an x_i for branching.[11] One objective in the design of these rules is to quickly find good *integer feasible* points (*i.e.*, points that are feasible in the original MICP), so that an incomplete optimization can still yield valuable results.

[11] Many different branching rules have been proposed, and the perhaps intuitive choice of branching on the variable whose value is farthest away from an integer (maximal fractional branching) turns out to be one of the worst, performing about as well as random selection [134].

Solving an MICP to global optimality using this basic branching algorithm amounts to brute-force enumeration of all integer variable assignments. The branch and bound algorithm (B&B) can be used to improve on this situation by exploiting the following observations:

1. the objective value $f(x^*)$ for a solution x^* to the continuous relaxation at any node is a lower bound on the objective value of the MICP at that node, since the feasible set of the relaxation contains the feasible set of the MICP at the node.

2. if the solution x^* at any node satisfies the integrality constraints, $f(x^*)$ is an upper bound on the objective value of the original (root) MICP.

By keeping track of the best upper and lower bound on the objective value during the solution process, B&B can stop expanding nodes for which the lower bound is higher than the best known upper bound. This is because further branching can only result in subproblems with smaller feasible sets, with optimal objective values that are at least as high as the lower bound established for their parent problem. Furthermore, B&B can certify global optimality of the best feasible point found so far (referred to as the *incumbent*) when the gap between the upper and lower bound reduces to zero. Optimality up to a desired tolerance may be certified in similar fashion.

Branch and cut algorithms (B&C), further improve on B&B by introducing the use of cutting planes to refine the continuous relaxations.[12] In the particular case of MILP, it is well known that the optimal value at any feasible and bounded continuous relaxation (an LP) is attained at an extreme point of the feasible set of the relaxation. If this extreme point, x^*, is a fractional solution, then a linear inequality is guaranteed to exist which separates x^* from the convex hull of the feasible set of the original MILP (proof by separating hyperplane theorem [135]). Such an inequality is also referred to as a *cutting plane*, or as a *valid inequality* for the feasible set of the original MILP with respect to x^* [132]. Many different methods have been proposed for generating

[12]We note that the term 'branch and bound' is often used in a more generic sense, referring to 'branch and cut' as well.

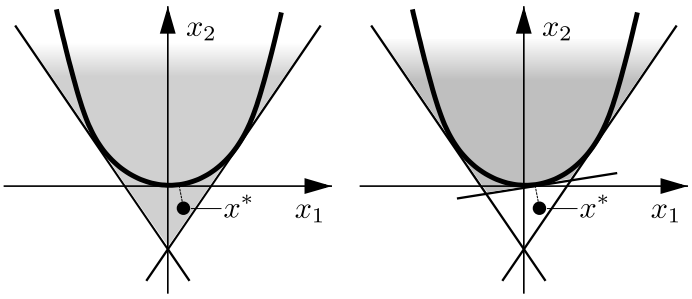


Figure 5-3: Polyhedral outer approximations for the convex constraint $x_2 \geq x_1^2$. **(a)** Point x^* is within the polyhedral outer approximation, but violates the original constraint. **(b)** Refined polyhedral outer approximation that renders the point x^* infeasible through the addition of a valid inequality (separating hyperplane), obtained by linearizing the constraint function at x^*. Adapted from [8].

cutting planes, starting with the work of Gomory [136]. Cutting plane generation can have a great impact on solver performance. B&C is especially valuable in the case of mixed-binary programs, since in this case cutting planes can be generated that are simultaneously valid for all nodes in the search tree [137, 138], while for general integer programs the cutting planes are only valid for descendants of the node at which they were generated.

Beyond MILP, a popular approach for solving general MICPs uses iteratively refined polyhedral outer approximations of the feasible set [139]. Similar to cutting plane methods, such approaches generate valid inequalities for the feasible set of the MICP with respect to the solution x^* of a continuous relaxation. In the case of polyhedral outer approximation, such inequalities can be found by linearizing the (convex) constraint functions, g_j, at x^* (see Fig. 5-3). A nonlinear (but convex) objective function f can be handled by introducing a slack variable, t, along with the constraint

$$f(x) \leq t,$$

and replacing the objective to minimize with t. The new constraint function $f(x) - t$ can then be handled using standard polyhedral outer approximation.

Other important components of MICP solvers include presolving methods and pri-

mal heuristics. Presolving methods are aimed at reducing problem size and tightening the formulation by adding valid constraints at the root note of the search tree. A trivial example is the elimination of variables whose upper bounds equal their lower bounds, but many more sophisticated techniques are in use. Primal heuristics are aimed at quickly finding a good incumbent, whose associated upper bound on the objective value can be used to quickly prune the search tree. Primal heuristics include diving methods [140]. At certain nodes, these methods select subsets of integer variables (*e.g.*, those that take near-integer values), and fix them to integer values (*e.g.,* by rounding). The associated continuous relaxation is then solved, and this process is applied recursively, reminiscent of a depth-first search [133].

Solvers that can handle (subclasses of) MICP include open-source implementations such as Cbc [141] (MILP), GLPK [142] (MILP), and Bonmin [143] (capable of more general MICP formulations). Commercial solver implementations are generally significantly more performant [144],[13] and include Gurobi, IBM CPLEX, FICO XPRESS, and MOSEK [145], which all support various MICP subclasses, including, but not limited to, MILP and MIQP.

5.3.3 Mixed-integer reformulations of disjunctive constraints

Mixed-integer convex programming may be used to solve problems with constraints of the form

$$x \in \bigcup_i \mathcal{S}_i,$$

where the $\mathcal{S}_i$ are convex sets and the union is finite. This application is especially interesting in light of our description of the contact environment as a finite union of environment regions. While each $\mathcal{S}_i$ is convex, such *disjunctive constraints* are in general nonconvex. Optimization problems involving disjunctive constraints are referred to as disjunctive programs [146]. Such programs may be reformulated as MICPs in various ways, allowing them to be solved using the techniques described in

[13] See also `http://plato.asu.edu/bench.html` for typically up-to-date benchmarks periodically performed by Dr. Hans Mittelmann.

the previous section.

As an important special case, consider the polyhedral disjunctive constraint

$$x \in \bigcup_i \mathcal{P}_i, \tag{5.5}$$

where each $\mathcal{P}_i$ is a bounded polyhedron. By the Minkowski-Weyl theorem, each $\mathcal{P}_i$ may be described either as the convex hull of a finite number of vertices (the V-representation) or as the intersection of a finite number of halfspaces (the H-representation). Each of these representations gives rise to various mixed-integer reformulations, each with different properties. We refer to [147] for a survey of such reformulations, and will only highlight one reformulation here, based on the H-representation.

In the H-representation, each $\mathcal{P}_i$ is described as

$$\mathcal{P}_i = \{x \mid A_i x \leq b_i\}.$$

Given this representation, one traditional way to implement constraint (5.5) in a mixed-integer setting is to use a *big-M* formulation,[14]

$$\begin{aligned} A_i x &\leq b_i + M_i (1 - z_i) \\ \sum_i z_i &= 1 \\ z_i &\in \{0, 1\} \quad \forall i \end{aligned} \tag{5.6}$$

where we introduced auxiliary binary indicator variables, z_i, along with constants M_i (either scalars or vectors of appropriate size). Note that only one z_i can be nonzero. If $z_i = 1$, the corresponding inequality constraint reduces to $A_i x \leq b_i$. For inequality constraints associated with zero-valued z_i variables, we have

$$A_i x \leq b_i + M_i \tag{5.7}$$

[14]Note that this formulation is rather ad-hoc; it is just meant to show the style of big-M formulations.

As long as each M_i is chosen large enough to ensure that constraint (5.7) is never active for any $x \in \bigcup_i \mathcal{P}_i$, we have succeeded in constructing a mixed-integer reformulation of the disjunctive constraint (5.5).

5.3.4 McCormick envelopes

While MICP reformulations of disjunctive polyhedral constraints can be used to model the problem of assigning contact bodies to regions and enabling or disabling associated constraints, fundamentally nonlinear constraints such as (5.3) are not MICP-representable, motivating the use of mixed-integer nonconvex programming (MINCP). However, before we delve into MINCP techniques, we will first discuss a commonly used approach to approximate bilinear constraints in an MICP formulation. Consider a constraint of the form

$$w = uv$$

where $w \in \mathbb{R}$, $u \in \mathbb{R}$, and $v \in \mathbb{R}$ are all decision variables. If u and v are bounded, $\underline{u} \leq u \leq \overline{u}$, and $\underline{v} \leq v \leq \overline{v}$, the tuple (u, v, w) is confined to a bounded set S,

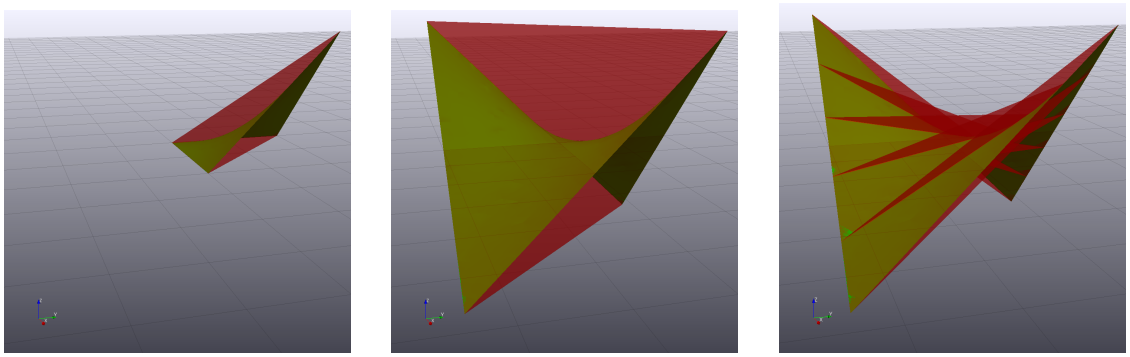
$$S = \{(u, v, w) \mid \underline{u} \leq u \leq \overline{u}, \underline{v} \leq v \leq \overline{v}, w = uv\}.$$

To approximate the constraint $(u, v, w) \in S$ in an MINCP, a commonly used approach is to simply use a convex hull relaxation,

$$(u, v, w) \in \operatorname{conv} S.$$

The convex hull is classically referred to as a McCormick envelope [148], and is visualized in Fig. (5-4a). It is a polyhedron, representable as a set of linear constraints in an MICP.

If the upper and lower bounds on u and v are tight, the maximal violation of the original bilinear constraint $w = uv$ may be acceptable. But with larger bounds, the quality of the approximation decreases drastically, as shown in Fig. 5-4b. To



(a) (b) (c)

Figure 5-4: (Piecewise) McCormick envelopes. **(a)** A single McCormick envelope (in red), used to relax the bilinear constraint $w = uv$ (green surface) in the region $0 \leq u \leq 1$, $0 \leq v \leq 1$. **(b)** Using McCormick envelopes with larger bounds on u and v, $-1 \leq u \leq 1, -1 \leq v \leq 1$, results in a bad approximation. **(c)** A piecewise McCormick formulation can be used to obtain a tighter relaxation, but at the cost of additional binary variables used to select between the pieces.

mitigate this problem, the intervals $[\underline{u}, \overline{u}]$ and/or $[\underline{v}, \overline{v}]$ may be subdivided, after which McCormick envelopes with smaller constraint violation may be constructed for these subregions of the (u, v) space (see Fig. 5-4c). However, containment of (u, v, w) in at least one of the McCormick envelope pieces is a disjunctive polyhedral constraint of the form (5.5). Consequently, auxiliary mixed-integer variables are needed for the associated MICP reformulation, resulting in artificial combinatorics and a tradeoff between computation time and constraint violation.

5.3.5 Mixed-integer nonconvex programming

This section gives a very brief overview of mixed-integer nonconvex programming (MINCP), which corresponds to the case that the f or g_j in (5.4) are nonconvex.

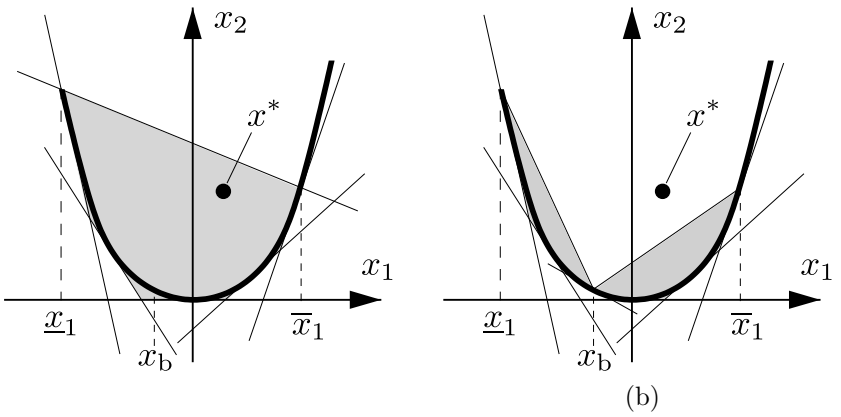
MINCPs are necessarily mixed-integer nonlinear programs (MINLPs), though the converse is not true. We note that in the literature, the term MINCP is used as often as MINLP, which is often contrasted with MICP subclasses with linear constraints such as MILP and MIQP. Although MINLPs with nonlinear but convex f and g_j are indeed a step up in difficulty, especially compared to MILP, the biggest gap in efficacy of solution techniques comes with the switch to nonconvex f and g_j. Not only

that, MINCP instances may be undecidable [149], though that problem is avoided if the decision variables are constrained to a bounded polyhedral set. Still, some easier problems may be solved to provably global optimality in reasonable time using existing MINLP solvers. For an excellent survey of applications, models, and solution methods for MINLP (including both nonlinear MICP and MINCP), see [8].

Solution methods for MINCP are often founded in those for MICP. In particular, tree search techniques like B&B and B&C have extensions to MINCP. The main additional challenge with MINCP is that even the continuous relaxation at a node in the search tree is a hard optimization problem, with potentially a disconnected feasible set and multiple local optima. One consequence of this is that guaranteed lower bounds on the objective function are not as easily obtainable.

The polyhedral outer approximation approach described in section 5.3.2, used for general MICPs, may be extended to the MINCP case. The key remains to ensure that any polyhedral relaxation solution x^* that violates the original constraints is infeasible after refinement of the relaxation. However, when constraint functions are nonconvex, it is not always possible to exclude a point x^* that is feasible in a given polyhedral outer approximation by adding a linear inequality, as was the case in Fig. 5-3. Fig. 5-5 demonstrates this issue for the nonconvex equality constraint $x_2 = x_1^2$ with x_1 in the interval $[\underline{x}_1, \overline{x}_1]$. In this case, x^* can be excluded from a subsequent, refined polyhedral approximation by subdividing the interval into two subintervals. The subproblems for each subinterval become branches in the search tree, and can be handled as usual. This approach, called spatial branch & bound (sBB), is considered the best-known method for solving MINCPs [8]. It effectively extends the notion of branching on integer variables, familiar from MICP, to continuous variables. In the case of bilinear constraints, sBB can be thought of as producing a piecewise McCormick formulation, but adaptively refined as part of the tree search process.

MINCP solvers often require a symbolic problem formulation: merely providing black-box functions for constraint evaluation is not enough. Based on the symbolic description, such solvers build an expression tree with variables or constants at the leaf nodes and basic operations ($+$, sin, $\cdot$, etc.) at the non-leaf nodes. An auxiliary variable



(b)

Figure 5-5: Example polyhedral outer approximation for the nonconvex equality constraint $x_2 = x_1^2$, with bounds on x_1. **(a)** Point x^* is feasible in the polyhedral outer approximation, but lies above the original constraint function. **(b)** Refined polyhedral outer approximation that renders the point x^* infeasible through spatial branching on x_1, with one branch with one branch for $x_1 \leq x_\mathrm{b}$ and one branch with $x_1 \geq x_\mathrm{b}$. Various heuristics exist for the selection of the branching point x_b. Adapted from [8].

is introduced for the output of each non-leaf node, effectively factorizing the constraint function into simpler constraints, involving only these basic operations. Solvers have a library of suitable outer approximations for the finite set of operations they support, which are typically more effective than basic linearization cuts. Moreover, the white-box problem formulation allows advanced solvers to determine local convexity of a subgraph of the expression tree for a constraint function when it is restricted to a suitably small subinterval, in which case specialized methods may be applied [150].[15]

Note that sBB heavily relies on the availability of known bounds on the decision variables. The most straightforward way to ensure that variables are suitably bounded is to explicitly specify (perhaps redundant) bounds in the problem description. However, typical solvers also implement bounds tightening techniques, which are used to infer tighter variable bounds on the basis of the objective and constraint functions. A very basic example is the use of interval propagation to infer that $x_2 \in [0, 2]$ given $x_1 \in [0, 1]$ and the constraint $x_2 = 2x_1$. Bounds tightening techniques are another reason that solvers often require symbolic problem descriptions. Bounds tightening

[15] As an example, consider the nonconvex constraint $x_2 \geq \sin(x_1)$. If, in a particular subproblem, x_1 is restricted to the interval $[\pi, 2\pi]$, then the constraint function restricted to the interval is convex.

techniques generally trade off between time and accuracy, as optimizing for the tightest possible variable bounds on x amounts to solving $2 \dim x$ problems of similar complexity to the original MINCP.

Similar to the MICP case, modern MINCP solvers rely heavily on primal heuristics for performance, so as to quickly find a good incumbent. See [133] for a recent dissertation on such techniques, applied mostly to the SCIP solver. Here, we discuss a few techniques that apply to MINCP, though we note that most originate from MICP. Examples include feasibility pump techniques, large neighborhood search strategies, and MPEC (mathematical program with equilibrium constraints) relaxations.

Feasibility pump techniques alternate between solving a continuous relaxation (a nonlinear program) to find a point that is likely fractional and heuristically rounding the integer variables to find an integer-feasible point that may violate the continuous constraints. The hope is that this alternating sequence converges to a feasible point of the original MINCP.

Large neighborhood search techniques heuristically determine a region around a reference point in the search space (for example, a bounded polyhedral region around the incumbent, or around the solution of an LP relaxation), and fix a subset of the integer variables. The goal is to arrive at a sub-MIP that is easier to solve, also because tighter variable bounds can be established in the neighborhood, leading to fewer possible integer assignments and tighter outer relaxations.

MPEC-based heuristics apply only to mixed-binary programs [126].[16] A binary constraint such as $z \in \{0, 1\}$ is rewritten as $z\,(1 - z) = 0$, a complementarity constraint. To avoid the constraint qualification issues mentioned in section 5.3.1, the complementarity constraint is relaxed to $z\,(1 - z) \leq \theta$, where θ is a scalar parameter, whose value may initially be set to, *e.g.*, $\frac{1}{4}$, and is iteratively reduced while feasible solutions are found. The relaxations are solved using a gradient-based nonlinear program solver.

MINCP-capable solvers include free/open-source implementations such as SCIP

[16] This description is based on `https://scip.zib.de/doc/html/heur__mpec_8h.php`, which documents the SCIP implementation of this heuristic.

[123] and COUENNE [125], where we note that development of SCIP is more active than that of COUENNE. Proprietary solvers include BARON [124], ANTIGONE [151], LindoGlobal [152], and Artelys Knitro.[17] These solvers can often be configured to use various subsolvers for continuous relaxations (NLP solvers such as Ipopt [153] and filterSQP [154]) and for MICP sub-problems (see section 5.3.2 for examples).

5.4 MINLP reformulation of the planning problem

This section describes how we reformulate the problem posed in section 5.2 as a finite-dimensional mixed-integer nonlinear program using the techniques described in section 5.3. Our general strategy is to limit the number of nonconvex constraints and the number of variables involved with unavoidable nonconvex constraints as much as possible, and to exploit the standard tools for working with disjoint sets from the mixed-integer literature. Furthermore, we restrict our usage of nonconvex constraints to bilinear constraints, as handling of such constraints is most mature and ubiquitous among MINCP-capable solvers. We will employ a Bézier curve parameterization for the continuous trajectories.

Section 5.4.1 lists the main decision variables and establishes an indexing convention. Section 5.4.2 discusses contact sequence timing constraints. Section 5.4.3 addresses the parameterization of continuous trajectories, such as that of the center of mass. Section 5.4.4 details the transcription of constraints stemming from the dynamics. Section 5.4.5 derives constraints on the binary variables that enforce the contact sequence requirements. Section 5.4.6 handles constraints on the contact forces, and section 5.4.7 addresses constraints on the CoPs and contact reference points. Finally, section 5.4.8 discusses additional, redundant variable bounds.

[17] We note that support for non-convex quadratic constraints is planned to be added to Gurobi 9.0, but this version is not available at the time of writing.

5.4.1 Decision variables

This section is mainly meant for future reference, and we refer to subsequent sections for precise interpretations of the indexing convention and decision variables.

Since sets of decision variables are defined on several axes, indexing can become unwieldy. We establish the following indexing convention:

- $j \in [\![n_{\mathrm{c}}]\!]$: contact body index.
- $k \in [\![n_{\mathrm{e}}]\!]$: environment region index.
- $i \in [\![n_{\mathrm{p}}]\!]$: time piece index for piecewise trajectories.

Defining $\mathcal{B}^{n,m}$ as the set of Bézier curves with n control points in $\mathbb{R}^m$ (having nm scalar decision variables) and using d to denote the degree of the CoM trajectory, the main decision variables are:

- $\Delta t[i] \in \mathbb{R}$: trajectory piece durations (if variable timing is used).
- $z_{j,k}[i] \in \{0, 1\}$: binary contact indicators.
- $c[i] \in \mathcal{B}^{d,3}$: center of mass trajectory pieces.
- $f_{\mathrm{c},j}[i] \in \mathcal{B}^{d-2,3}$: pieces of the total contact force exerted upon body $\mathcal{C}_j$, expressed in world frame Φ_{w}.
- $\tilde{f}_{\mathrm{c},j,k}[i] \in \mathcal{B}^{d-2,3}$: pieces of the individual contact forces exerted upon body $\mathcal{C}_j$ from region $\mathcal{E}_k$, expressed in local frame Φ_k.
- $r_j[i] \in \mathbb{R}^3$: contact reference points in world frame Φ_{w}.
- $\tilde{r}_{j,k}[i] \in \mathbb{R}^2$: region-local contact reference points in the x-y plane of frame Φ_k.
- $p_j[i] \in \mathcal{B}^{d,3}$: pieces of CoP trajectories in world frame Φ_{w}.
- $\tilde{p}_{j,k}[i] \in \mathcal{B}^{d,2}$: pieces of region-specific CoP trajectories, defined in the x-y plane of frame Φ_k.

We note that certain auxiliary variables have been omitted, and will be introduced later.

5.4.2 Timing

Contact sequence timing may either be prespecified or left free. In either case, we break up the trajectory into n_p pieces, and refer to the duration of trajectory piece i as $\Delta t[i]$. If timing is prespecified, the $\Delta t[i]$ are simply data. If timing is left free, the $\Delta t[i]$ are decision variables, and we add the constraints

$$\underline{\Delta t}[i] \leq \Delta t[i] \leq \overline{\Delta t}[i] \quad \forall i \in [\![n_\mathrm{p}]\!]$$

in partial fulfillment of the timing requirements specified in section 5.2.5.

For future use, we also introduce auxiliary variables $\Delta t_\mathrm{sq}[i]$ and constraints

$$\Delta t_\mathrm{sq}[i] = \Delta t[i]^2 \quad \forall i \in [\![n_\mathrm{p}]\!] .$$

Note that these constraints are bilinear in the decision variables when timing is left free. Again, if timing is prespecified, the $\Delta t_\mathrm{sq}[i]$ are just data.

5.4.3 Parameterization of continuous trajectories

We opt to represent the trajectories of the center of mass, $c(t)$, contact forces, $f_{\mathrm{c},j}(t)$, and CoPs, $p_j(t)$, as composite (piecewise) Bézier curves defined on the interval $[0, t_\mathrm{f}]$, where $t_\mathrm{f} = \sum_{i=1}^{n_\mathrm{p}} \Delta t[i]$. That is, for $x \in \{c, f_{\mathrm{c},j}, p_j\}$,

$$x(t) = \Big\{ x[i](\theta[i](t)) \quad \text{if } t[i-1] \leq t \leq t[i]$$

where the $\{x[i]\}_{i \in [\![n_\mathrm{p}]\!]}$ are the individual Bézier curves for each piece, $t[0] = 0$, $t[i] = t[i-1] + \Delta t[i]$, $t[n_p] = t_\mathrm{f}$, and

$$\theta[i](t) = \frac{t - t[i-1]}{\Delta t[i]}, \quad \Delta t[i] = t[i] - t[i-1]$$

serves as a phase (interpolation) variable for piece i. For a piecewise-Bézier curve in $\mathbb{R}^3$ of order n, each Bézier piece is defined by n 3-dimensional control points, which will serve as decision variables.

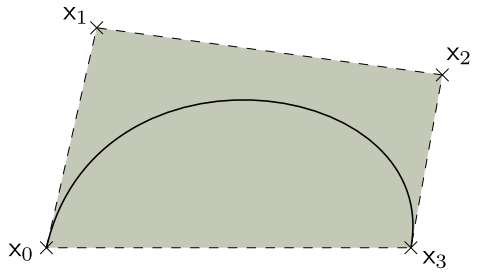


Figure 5-6: The Bézier curve convex hull property: for Bézier curve $x(t)$ with control points $\{\mathsf{x}_1, \mathsf{x}_2, \dots\}$, the curve $\{x(\theta) \mid 0 \leq \theta \leq 1\}$ lies within the convex hull of the control points. Adapted from `https://commons.wikimedia.org/wiki/File:Bezier_curve.svg`.

The main advantage of this parameterization is that it allows us to exploit the convex-hull property of Bézier curves (see Fig. 5-6): the entire curve for piece i, $\{x_i(\theta) \mid 0 \leq \theta \leq 1\}$, is contained in the convex hull of the control points for piece i. This means that we can (conservatively) translate infinite-dimensional constraints on trajectories into low-dimensional constraints on the control points. As a special case, the convex-hull property allows us to easily establish valid upper and lower bounds on the decision variables, which often result in better solver performance, as noted in section 5.3.

For the experiments presented in section 5.7, we use piecewise cubic Bézier curves to represent $c(t)$ (*i.e.*, $d = 4$). To match the degree of $\ddot{c}(t)$ according to the CoM dynamics, (5.2), we use piecewise linear curves for the $f_{\mathrm{c},j}(t)$. Guided by the degree of the right-hand side of the nonlinear angular momentum rate constraint, (5.3), we choose the degree of the $p_j(t)$'s to be the same as that of $c(t)$. The break points for each of the continuous trajectories are aligned.

To ensure continuity of the CoM trajectory at the break points, we require that for $i \in [\![n_{\mathrm{p}} - 1]\!]$

$$c[i+1](0) = c[i](1)$$

which simply means that the last control point of $c[i](\cdot)$ should be constrained to be equal to the first control point of $c[i+1](\cdot)$. To additionally ensure differentiability at the break points, we use the chain rule to find

$$\dot{c}[i](t) = \frac{dc[i](\theta[i](t))}{dt} = \frac{\partial c[i](\theta[i](t))}{\partial \theta[i]} \frac{d\theta[i]}{dt} = \frac{c'[i](\theta[i](t))}{\Delta t[i]},$$

where the prime denotes a derivative with respect to a phase variable. We therefore require that for $i \in [\![n_\mathrm{p} - 1]\!]$,

$$\frac{c'[i+1](0)}{\Delta t[i+1]} = \frac{c'[i](1)}{\Delta t[i]}$$

or equivalently

$$c'[i+1](0)\,\Delta t[i] = c'[i](1)\,\Delta t[i+1].$$

Note that these are linear constraints on the control points of the $c[i]$ if the contact sequence timing is prespecified, since the derivative operator simply applies a linear transformation to the control points. However, with variable timing, these constraints become bilinear in the decision variables.

The initial and final conditions on the CoM trajectory from section 5.2.7 also translate into constraints on the control points of the Bézier curves.

5.4.4 Dynamics constraints

To enforce the CoM dynamics (5.2), we differentiate the CoM trajectory pieces once more to find

$$\ddot{c}[i](t) = \frac{c''[i](\theta[i](t))}{\Delta t[i]^2}.$$

We now introduce a slack piecewise Bézier trajectory for the total contact force, $f_\mathrm{c}(t)$, and subject it to the constraint

$$f_\mathrm{c}[i](t) = \sum_{j=1}^{n_\mathrm{c}} f_{\mathrm{c},j}[i](t).$$

This allows (5.2) to be written as

$$c''[i](\theta[i]) = (mg + f_\mathrm{c}[i](\theta[i]))\,\Delta t_\mathrm{sq}[i] \quad \forall \theta[i] \in [0,1], i \in [\![n_\mathrm{p}]\!]$$

This implies that the control points of the Bézier curve pieces on either side of the equation must match.[18] Again, this is a linear constraint on the control points of c and f_c when the timing is fixed, which becomes bilinear in the control points of f_c and the Δt_sq with variable timing. In the latter case, the introduction of the slack trajectory $f_\mathrm{c}(t)$ reduces the number of variables involved in the bilinear constraint.

To handle the inherently nonlinear angular momentum rate constraint, (5.3), we first note that the slack trajectory $f_\mathrm{c}(t)$ can also be used to replace $\sum_{j=1}^{n_\mathrm{c}} f_{\mathrm{c},j}(t)$ in this constraint:

$$\sum_{j=1}^{n_\mathrm{c}} p_j(t) \times f_{\mathrm{c},j}(t) = c(t) \times f_\mathrm{c}(t) \quad \forall t \in [0, t_\mathrm{f}]$$

We then substitute the piecewise Bézier parameterization of the trajectories, resulting in

$$\sum_{j=1}^{n_\mathrm{c}} p_j[i](\theta) \times f_{\mathrm{c},j}[i](\theta) = c[i](\theta) \times f_\mathrm{c}[i](\theta) \quad \forall \theta[i] \in [0,1], i \in [\![n_\mathrm{p}]\!]$$

In practice, we compute the required cross products of Bézier curve pieces symbolically in terms of the control point decision variables by converting each piece to the monomial basis (an invertible linear transformation of the control points), after which a standard polynomial multiplication routine can be used. Note again that this constraint is nonlinear even with fixed contact sequence timing.

Without loss of generality, we normalize the contact forces by setting the total mass m of the robot in the reformulated problem equal to one.

[18] One way to see this is as follows. Each constraint can be rewritten as $x(\theta[i]) = 0$, where x is a Bézier curve. A Bézier curve is simply a polynomial in the Bernstein basis, restricted to the interval $[0, 1]$. Its coefficients (control points) are just an invertible linear transformation away from those of a polynomial of the same degree that is expressed in the monomial basis, and in the monomial basis it is clear that all of the coefficients need to be zero for the polynomial to be identically zero.

5.4.5 Region assignment constraints

To encode the rules that govern allowable contact sequences, as described in section 5.2.5, we introduce binary contact indicator variables

$$z_{j,k}[i] \in \{0, 1\},$$

where $z_{j,k}[i] = 1$ if and only if contact body $\mathcal{C}_j$ is in contact with environment region $\mathcal{E}_k$ during trajectory piece i. Note that if body $\mathcal{C}_j$ is not assigned to any region during piece i, *i.e.*,

$$z_{j,k}[i] = 0 \quad \forall k \in [\![n_{\mathrm{e}}]\!]$$

or equivalently

$$\sum_{k=1}^{n_{\mathrm{e}}} z_{j,k}[i] = 0$$

then body $\mathcal{C}_j$ is in a swing phase.

To ensure that each contact body is assigned to at most one environment region at any given time, we add the constraints

$$\sum_{k=1}^{n_{\mathrm{e}}} z_{j,k}[i] \leq 1 \quad \forall i \in [\![n_{\mathrm{p}}]\!], j \in [\![n_{\mathrm{c}}]\!]. \tag{5.8}$$

We note that these constraints take the specific form of type-1 special ordered set (SOS1) constraints, which most mixed-integer solvers handle using specialized routines that improve search efficiency in branch and bound algorithms, compared to generic handling of linear constraints on binary variables [155].

Next, the requirement that environment region reassignments be interleaved with swing phases must be met. To do so, we first introduce notation for the differences between the contact indicator variables for consecutive trajectory pieces:

$$\Delta z_{j,k}[i] = z_{j,k}[i+1] - z_{j,k}[i], \quad i \in [\![n_{\mathrm{p}} - 1]\!].$$

Considering each contact body $\mathcal{C}_j$ individually, we inspect the sum of absolute differ-

ences over all environment regions,

$$\sum_{k=1}^{n_\mathrm{e}} |\Delta z_{j,k}[i]| \,.$$

We find that there are three cases:[19]

1. $\sum_{k=1}^{n_\mathrm{e}} |\Delta z_{j,k}[i]| = 0$: body $\mathcal{C}_j$ is assigned to the same environment region during piece i and piece $i+1$;

2. $\sum_{k=1}^{n_\mathrm{e}} |\Delta z_{j,k}[i]| = 1$: body $\mathcal{C}_j$ is assigned to an environment region during piece i and is unassigned during piece $i+1$, or vice versa;

3. $\sum_{k=1}^{n_\mathrm{e}} |\Delta z_{j,k}[i]| = 2$: body $\mathcal{C}_j$ is assigned to different environment regions during piece i and piece $i+1$.

The third case should be disallowed, so we require that

$$\sum_{k=1}^{n_\mathrm{e}} |\Delta z_{j,k}[i]| \leq 1, \quad \forall i \in [\![n_\mathrm{p} - 1]\!] \,, j \in [\![n_\mathrm{c}]\!] \,. \tag{5.9}$$

These are (convex) ℓ_1-norm constraints, which can be incorporated using the standard procedure of introducing continuous slack variables

$$0 \leq s_{j,k}[i] \leq 1 \quad \forall i \in [\![n_\mathrm{p} - 1]\!] \,, j \in [\![n_\mathrm{c}]\!] \,, k \in [\![n_\mathrm{e}]\!]$$

along with the constraints

$$\begin{aligned} s_{j,k}[i] &\geq +\Delta z_{i,j,k} \\ s_{j,k}[i] &\geq -\Delta z_{i,j,k} \\ \sum_{k=1}^{n_\mathrm{e}} s_{j,k}[i] &\leq 1. \end{aligned}$$

[19] Due to the binary nature of the contact indicator variables and the SOS1 constraint (5.8), $\sum_{m=1}^{n_e} |\Delta z_{i,j,m}| > 2$ is impossible.

If jumping is not allowed, we can simply add the constraints

$$\sum_{j=1}^{n_\mathrm{c}} \sum_{k=1}^{n_\mathrm{e}} z_{j,k}[i] \geq 1, \quad \forall i \in [\![n_\mathrm{p}]\!] ,$$

which ensure that at least one body is assigned to a region during every trajectory piece.

Finally, to ensure that the duration of each swing phase is upper-bounded by $\overline{\Delta t}$, we rely on the timing constraints for each piece as described in section 5.4.2, but we must also disallow consecutive swing phases.[20] This is easily accomplished by adding the constraints

$$\sum_{k=1}^{n_\mathrm{e}} z_{j,k}[i] + \sum_{k=1}^{n_\mathrm{e}} z_{j,k}[i+1] \geq 1, \quad \forall i \in [\![n_\mathrm{p} - 1]\!] , j \in [\![n_\mathrm{c}]\!] .$$

Initial and final conditions on the contact situation simply result in fixings of the $z_{j,k}[1]$ and $z_{j,k}[n_\mathrm{p}]$.

5.4.6 Contact force constraints

Constraints on the contact forces (see section 5.2.3) depend on whether a given contact body is assigned to the region, as well as the environment region description. To enforce these constraints, we first split up the contact force Bézier pieces into region-specific components, expressed in region-local coordinates. The two are related by the Bézier curve constraints

$$f_{\mathrm{c},j}[i] = \sum_{k=1}^{n_\mathrm{e}} R_k^\mathrm{w} \tilde{f}_{\mathrm{c},j,k}[i] \quad \forall j \in [\![n_\mathrm{c}]\!] , i \in [\![n_\mathrm{p}]\!]$$

where $R_k^\mathrm{w} \in SO(3)$ is the rotation component of the rigid transformation T_k^w. In region-local coordinates, we then constrain the control points of the $\tilde{f}_{\mathrm{c},j,k}[i]$ to lie within the friction cone $\mathcal{F}_k$, exploiting the Bézier curve convex hull property. These are second-order cone constraints, for which more advanced solvers have special sup-

[20] Note that consecutive stance phases (assigned to the same region) should still be allowed.

port. Similar to the construction used in the motivating example of section 5.3.1, we add constraints

$$0 \leq \left(\tilde{f}_{c,j,k}[i]\right)_z \leq z_{j,k}[i]\overline{f}_c \quad \forall j \in [\![n_c]\!], k \in [\![n_e]\!], i \in [\![n_p]\!]$$

to meet the requirement that forces can only be exerted when there is contact and to upper-bound contact force normal components. Here, $(x)_z$ extracts the z-component of vector x. Again, these Bézier curve constraints translate into constraints on control points.

5.4.7 CoP and contact reference point constraints

Similar to the treatment of the contact forces, we have decision variables for the region-local coordinates of the CoPs p_j and contact reference points r_j. For all $j \in [\![n_c]\!], k \in [\![n_e]\!], i \in [\![n_p]\!]$,

$$\tilde{r}_{j,k}[i] \in \mathbb{R}^2$$

refers to the x-y coordinates of body j's contact reference point in local frame Φ_k during piece i. These local coordinates are only valid if body j is in contact with region k. In the local coordinates, the polyhedral region constraints from section 5.2.3 should be satisfied:

$$\tilde{r}_{j,k}[i] \in \tilde{\mathcal{P}}_k \tag{5.10}$$

We then introduce additional auxiliary decision variables $\hat{r}_{j,k}[i]$, representing the difference between the contact reference points in world coordinates and the transformed contact reference points in local coordinates:

$$\hat{r}_{j,k}[i] = r_j[i] - T_k^w \left(\begin{pmatrix} \tilde{r}_{j,k}[i] \\ 0 \end{pmatrix} \right)$$

after which we use a big-M formulation to ensure equality if body j is in contact with region k:

$$-M_r (1 - z) \leq \hat{r}_{j,k}[i] \leq M_r (1 - z)$$

where M_r is a suitably large constant.

The per-contact CoP's p_j are handled in very similar fashion. However, since each $p_j[i]$ is a Bézier curve, we apply the constraints to the 2-dimensional control points of $\tilde{p}_{j,k}[i]$. Furthermore, we omit the equivalent of (5.10), and instead add the quadratic constraints

$$\|p_j[i] - r_j[i]\|^2 \leq \overline{d}_{p,j}^2$$

to fulfill the CoP distance requirement, (5.1). This polynomial constraint is again conservatively approximated using constraints on the control points of the $p_j[i]$. We opt to implement these constraints in world coordinates because this avoids having the number of these quadratic constraints scale with the number of contact regions.

The approximate kinematic constraints from section 5.2.6 translate directly into constraints on the control points of the CoM and the contact reference points.

5.4.8 Variable bounds

As noted in section 5.3.5, the search space should be suitably bounded so as to avoid formulating an undecidable problem. Moreover, section 5.3.5 notes the importance of having tight variable bounds in an MINCP setting, so that the initial outer approximations of nonconvex constraint functions are not unnecessarily conservative.

While we rely on the solvers' bounds tightening capabilities to an extent, adding explicit bounds on at least a subset of the decision variables as a seed is required. Some solvers, such as BARON, will return an error code if a problem has insufficient variable bounds. Furthermore, variable bounds that may seem redundant, given *e.g.* the initial conditions and approximate kinematic constraints, can drastically change solver performance and also reduce the performance gap between different solvers due to differences in bounds tightening techniques.

Aside from the bounds discussed in previous sections, the main 'redundant', manually added variable bounds we use are on:

- squared piece time durations: $\underline{\Delta t}[i]^2 \leq \Delta t_{\text{sq}}[i] \leq \overline{\Delta t}[i]^2$;

- contact reference points in world frame ($r_j[i]$): bounding box around environment regions $\{\mathcal{P}_k\}_{k\in[\![n_e]\!]}$ (computed from V-representations);
- CoM control points in world frame ($c[i]$): obtained by offsetting the contact reference points bounding box;
- contact reference points in local frames ($\tilde{r}_{j,k}[i]$): obtained from bounding boxes around the $\tilde{\mathcal{P}}_k$ polyhedra.
- CoP control points in world frame and local frames ($p_j[i]$, $\tilde{p}_{j,k}[i]$): obtained by offsetting bounding boxes for $r_j[i]$ and $\tilde{r}_{j,k}[i]$;
- tangential components of contact forces in local frames ($\tilde{f}_{c,j,k}[i]$): from friction cone and upper limit on contact normal force, $\overline{f}_c$.

5.4.9 MIQP relaxation

Our computational results include a comparison between solving the MINCP reformulation of the planning problem directly using a dedicated MINCP solver, and solving an MIQP relaxation of this problem using a dedicated MIQP solver, after approximating the nonconvex quadratic constraints using piecewise McCormick envelopes (see section 5.3.4). The polyhedral disjunctive constraints are reformulated as mixed-integer/linear constraints using a 1-dimensional version of the method presented in [156], which uses a number of binary variables and constraints that is logarithmic in the number of McCormick envelope pieces.

5.5 Whole-body control

Using the results of the trajectory optimization as inputs to the whole-body controller from chapter 3 is fairly straightforward. We use the motion task setup and weights described in section 3.4.2.

The piecewise-Bézier CoM trajectory is used unaltered as the reference for a PD controller on CoM position, which in turn outputs a desired value for the linear mo-

mentum rate of change motion task, thus replacing the ICP-based linear momentum reference generation from section 3.4.3. Trajectories for the contact forces and CoPs are not used for online tracking. The trajectories found by the MINLP solver simply provide a certificate that the center of mass trajectory is indeed feasible given the contact sequence (in similar vein to *e.g.* [104]). This also makes it unnecessary to add constraints or objective terms to the trajectory optimization aimed at smoothening individual contact forces.

Turning the sequence of contact reference points into a footstep plan that includes the orientations of the contact bodies at touchdown is slightly less trivial. The sequence of desired touchdown orientations of a given contact body is computed recursively, starting from the body's actual orientation at $t = 0$. Each subsequent touchdown orientation is computed by finding the closest orientation to the previous orientation, subject to the constraint that the z-axis be aligned with the normal n_k of the environment region $\mathcal{E}_k$ with which the body is coming into contact. Here, great-circle distance is used as the metric to minimize, and we use the well-known analytic solution for this problem.

Given the footstep plan, the foot trajectory generation proceeds as described in section 3.4.2.

5.6 Implementation

The implementation was done in Julia. The main code base is available at `https://github.com/tkoolen/CentroidalTrajOpt.jl`.

Section 5.6.1 describes the various solvers that were used. Section 5.6.2 describes software used for problem formulation.

5.6.1 Solvers

See table 5.1 for a list of used solvers. MINLP solvers use one or more subsolvers for both mixed-integer linear relaxations or nonlinear continuous relaxations / reformulations, which are also listed. BARON ships with Ipopt, FilterSD and FilterSQP

Table 5.1: List of main solvers and subsolvers.

Solver	Version	Julia wrapper	Role
BARON	18.8.24	BARON.jl[21]	MINLP solver
SCIP	6.0.2	SCIP.jl[22]	MINLP solver
CPLEX	12.8.0	Not used.	MILP subsolver for SCIP, BARON
Ipopt	3.12.12	Ipopt.jl[23]	NLP subsolver for SCIP, complementarity comparison
WORHP	1.13	N/A	NLP subsolver for SCIP

as nonlinear subsolvers, but subsolver versions are unknown. Ipopt was built with ASL and METIS, with HSL for linear solver implementations, and with Intel MKL as the BLAS/LAPACK implementation. SCIP was built with this version of Ipopt as well as the WORHP NLP solver, in addition to using CPLEX as the (mixed-integer) linear program solver.

Minimal parameter tuning was performed for these solvers. Default settings were used for Ipopt. The following parameter settings were used for BARON:

- `AllowFilterSD`: 1
- `AllowFilterSQP`: 1
- `AllowIpopt`: 1
- `threads`: 10 (on a machine with 10 physical cores)

Parameters for SCIP were as follows:

- Emphasis setting: feasibility
- Presolving hyperparameter: aggressive
- Heuristics: aggressive
- `heuristics/rens/freq`: -1

[21] `https://github.com/joehuchette/BARON.jl`
[22] `https://github.com/SCIP-Interfaces/SCIP.jl`
[23] `https://github.com/JuliaOpt/Ipopt.jl`

For Ipopt, we set `linear_solver` to `ma27` (a fast solver that is part of HSL).

The last of these settings disables the RENS (Relaxation Enforced Neighborhood Search) heuristic, which often takes a long time, but appears to never be successful in finding a feasible point.

5.6.2 Problem formulation

For whole-body control, the software stack described in section 3.5 is used. The software stack for trajectory optimization comprises mainly of the following packages:

- JuMP.jl[24] is used as the optimization modeling language.
- Polyhedra.jl[25] is used for polyhedron representation conversions.
- AxisArrays.jl[26] is used to simplify storing and access of decision variables.
- MultilinearOpt.jl[27] (based on PiecewiseLinearOpt.jl [157]) is used to automatically generate the MIQP relaxation of the MINCP.
- StaticUnivariatePolynomials.jl[28] is used to represent and manipulate polynomials, both in monomial and Bernstein basis.

All of these packages are free and open source. Of these packages, only the last was fully written by the author.

5.7 Results

This section presents results of computational experiments. All results were obtained on a desktop machine with an Intel Core i7-6950X CPU @ 3.00GHz. Section 5.7.1 presents the benchmark scenarios. Section 5.7.2 presents nominal performance results, and section 5.7.3 investigates performance variability in the face of small changes to the problem formulation.

[24] `https://github.com/JuliaOpt/JuMP.jl`
[25] `https://github.com/JuliaPolyhedra/Polyhedra.jl`
[26] `https://github.com/JuliaArrays/AxisArrays.jl`
[27] `https://github.com/joehuchette/MultilinearOpt.jl`
[28] `https://github.com/tkoolen/StaticUnivariatePolynomials.jl`

Table 5.2: Problem parameters.

Parameter	Value	Description
$\underline{\Delta t}$	0.6 s	Min. trajectory piece time.
$\overline{\Delta t}$	1.5 s	Max. trajectory piece time.
$\overline{d}_{p,j}$ $(\forall j)$	0.0626 m	Max. distance from CoP to contact reference point.
$\overline{d}_{c,j}$ $(\forall j)$	1.05 m	Max. distance between CoM and contact reference points.
$\underline{d}_{c,j}$ $(\forall j)$	0.8 m	Min. distance between CoM and contact reference points.
$\underline{d}_{r,1,2}$	0.2919 m	Min. distance between foot contact reference points.

5.7.1 Scenarios

Results are presented for two scenarios. Referring to the scenario of Fig. 5-1, the scenarios are:

1. Only regions $\mathcal{E}_1$ and $\mathcal{E}_2$ are present, and the trajectory consists of $n_\mathrm{p} = 4$ pieces.

2. Regions $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$, and $\mathcal{E}_4$ are all present, and the trajectory consists of $n_\mathrm{p} = 10$ pieces.

In both scenarios, the only contact bodies are the feet. The intial state is the same in both scenarios, and the goal is to end up with both feet in the environment region with highest index. See Table 5.2 for a list of additional problem parameters, shared between the scenarios.

5.7.2 Nominal performance

See Table 5.3 for the time taken by BARON and SCIP to find a feasible point for the trajectory optimization problems for each scenario. See Fig. 5-7 for an example CoM trajectory with scenario 2. Unless stated otherwise, the resulting trajectory was successfully executed in simulation on the full robot using the momentum-based control framework.

We expected that the fixed-timing version would be easier to solve than the version with variable timing, due to the lower number of bilinear constraints. To evaluate this hypothesis, we first found a solution with variable timing, after which a fixed-timing problem was formulated using the values from the variable timing problem.

Table 5.3: Solver performance: time to find a feasible point. Times reported by solvers.

Solver	Scenario	Timing	Time (s)	Notes
BARON	1	Variable	189	
		Fixed	106	
	2	Variable	4453	
		Fixed	2294	
SCIP	1	Variable	0.53	
		Fixed	0.52	
	2	Variable	21.83	
		Fixed	Time limit (1800)	
Ipopt	1	Variable	3.876	
		Fixed	8.575	
	2	Variable	35.228	Ipopt problem restoration phase failed, constraints violated, plan execution failed.
		Fixed	32.30	Ipopt problem restoration phase failed, constraints violated, plan execution failed.

The results are also listed in table 5.3.

Binary constraints of the form $z \in \{0, 1\}$ can be reformulated as complementarity constraints, $z\,(1 - z) = 0$. This allows a gradient-based solver like Ipopt to be used instead of a dedicated MINLP solver. This approach was used to generate the entries for Ipopt in Table 5.3.

SCIP typically finds feasible points using one of the following heuristics:

- `subnlp`, which applies NLP local search to the problem after fixing all integer variables, tends to be successful for smaller problems like scenario 1.
- `alns` (adaptive large neighborhood search [158]), which orchestrates eight popular large neighborhood search (LNS) heuristics (see section 5.3.5), tends to be successful for larger problems like scenario 2.

See Fig. 5-8 for freeze frames of a simulation with the momentum-based controller used to track this CoM trajectory. Foot position tracking errors at touchdown are generally within 2.5 cm. There is occasionally some foot slip.

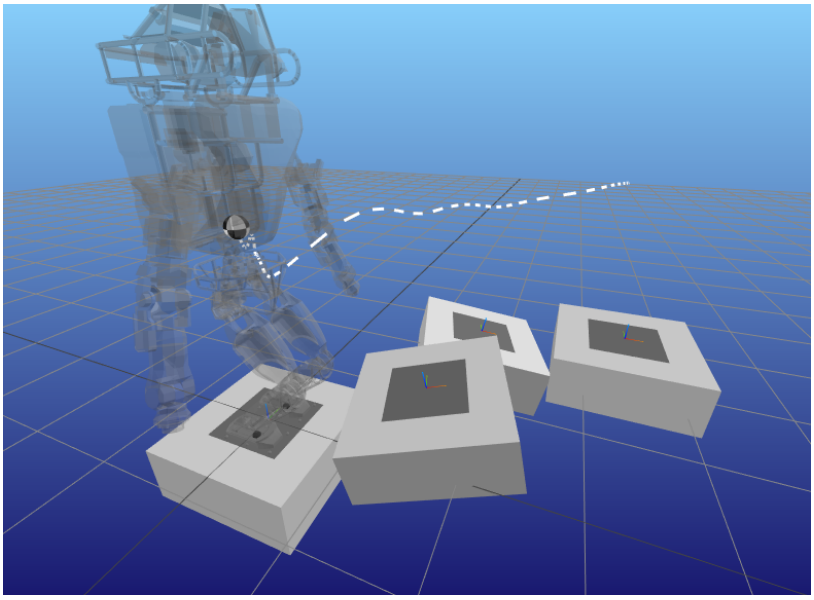


Figure 5-7: Center of mass trajectory for scenario 2, found by SCIP with free timing.

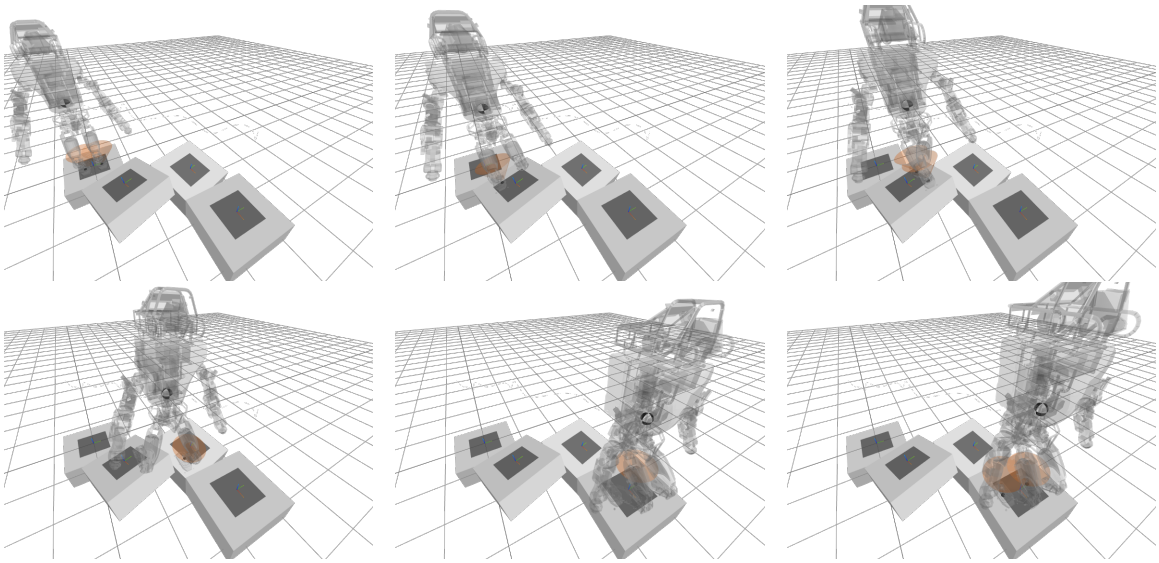


Figure 5-8: Result of tracking the CoM trajectory found by SCIP with free timing for scenario 2.

Table 5.4: Performance variability with SCIP: time to find a feasible point when varying the displacement Δx of the final environment region along the x-axis. Times reported by solver. Solution time was limited to 400 s.

Scenario	Δx (mm)	Time (s)
1	-2	0.57
	-1	0.52
	0	0.53
	1	0.48
	2	0.50
2	-2	21.86
	-1	Time limit
	0	21.83
	1	Time limit
	2	Time limit

5.7.3 Performance variability

Mixed-integer solvers are infamous for having rather unpredictable performance characteristics [159]. We investigated performance variability in finding the first feasible point by varying the position of the origin of the final environment region ($\mathcal{E}_2$ in scenario 1, $\mathcal{E}_4$ in scenario 2) along the x-axis of Φ_{w} within a 5 mm interval centered around the nominal case discussed in section 5.7.2. See Table 5.4 for the resulting solution times with SCIP.

5.8 Discussion

This chapter represents an exploratory study, with formulation and solution techniques that have not yet received much attention from the robotics community.This work should be considered an initial assessment of these techniques, and a starting point for future studies.

We believe that there is much to learn from the mixed-integer optimization community, particularly from primal heuristics devised for mixed-binary nonconvex programs and outer approximation methods. We note that the methods of [105] and [29] can be considered implementations of these outer approximation methods.

The remainder of this section focuses on the following areas: possible extensions

(section 5.8.1), experience with the solvers (section 5.8.2), and issues with whole-body tracking and use on a physical robot (section 5.8.3). Section 5.8.4 provides some future perspectives.

5.8.1 Possible extensions

We note that there are many possible variations of the presented problem statement and MINLP reformulation, corresponding to various tradeoffs between problem complexity and fidelity. As a result of practical experience with MINLP solvers and to simplify exposition, we chose a fairly basic problem formulation.

The presented formulation is most notably missing any notion of orientation of the contact bodies and of the pose of the full robot beyond center of mass position. Future work could explore approximating the pose of the robot with the pose of a lumped rigid body used to approximate the trunk. This would allow for more accurate approximate kinematic constraints, which are the main impediment to application on a physical robot.

The formulation does not include any contact torques normal to the contact surfaces. Experiments with adding a friction-limited normal torque component suggested that this addition results in a much longer time to find a feasible point. Perhaps it would be better to switch from a formulation with time-variant body-fixed CoPs, p_i, as decision variables to a formulation with several body-fixed contact points, more akin to the QP formulation used in the whole-body controller.

Friction coefficients were modeled as being dependent only on the environment region. However, each contact body could be made of a different material, in which case the coefficient of friction depends on the combination of environment region and contact body. Modeling this situation would be straightforward.

In our problem statement, we conflated time duration constraints for the swing phases with those for the stance phases. Supporting a different minimal swing time and minimal stance time could be done using additional indicator constraints.

5.8.2 Solver performance and experiences

The performance of the MINLP solvers used for the computational experiments is perhaps not as good as one might hope. The relative unpredictability in the time to find the first feasible point is especially worrisome. However, we stress that we did not specify any initial point, while other approaches can be quite sensitive to the seed.

Compared to a direct reformulation using complementarity constraints and solution using Ipopt, the benchmarks in the previous section show that a dedicated mixed-integer nonlinear program solver like SCIP can improve performance, especially for larger problems.

As noted, typical MINLP solvers orchestrate various subsolvers, used to solve continuous relaxations and linear outer approximations. Each of the subsolvers may have various parameters that affect performance. In addition, the tree search itself may involve an array of different heuristics, and these heuristics also have associated parameters. Additional frequency and priority parameters govern at which nodes each of these heuristics are applied. Tuning all of these parameters can be a daunting process, with further frustration stemming from performance variability between problem instances. SCIP has over 1600 tunable parameters, while BARON only exposes a fairly limited set of parameters to the user, relying more on pre-tuned, hardcoded defaults. Some guidelines for parameter tuning can be found in [160], and we note the usefulness of SCIP's emphasis meta-parameters. An advantage of SCIP over BARON is that it is possible to query statistics regarding how often each primal heuristic was applied to a particular problem, how much time it required, and whether it was successful in finding a feasible point. A basic parameter tuning approach is to de-prioritize primal heuristics that take a lot of time but are not often successful.

Quickly finding a first feasible point is of great importance, both to provide anytime capability and to effectively prune the search tree when minimizing an objective function. We note that whether or not an objective function is specified may have a large effect on the time required to find the first feasible point. In our experience,

it is preferable to first solve a feasibility problem to find a feasible point, and then reoptimize with the desired objective function.

We expected that fixing the timing of the contact sequence a priori would result in vastly reduced times to find a first feasible point, due to the greatly reduced number of nonconvex constraints, as well as a reduced number of variables involved in such constraints. Instead, we found that fixing the timing to values found by solving a problem with variable timing lead to increased solution times or inability to find a solution within reasonable time limits with SCIP, and reduced, but still long solution times with BARON.

As briefly mentioned in section 5.7.3, performance variability is a frustrating reality of mixed-integer nonlinear program solvers, both when finding an initial feasible point and while improving objective function bounds. This is a result of the heavy reliance on heuristics, in combination with tree search and rounding methods that often rely on floating point number comparisons. Performance may degrade or improve significantly depending on the number of regions and the number of trajectory pieces, as well as solver build options and available subsolvers, where we note that switching to subsolvers that are supposed to be better on paper can sometimes result in worse performance for the benchmarks we tested. More worryingly, very minor changes to the problem formulation can have large effects, especially for large problems. Such changes include the order in which constraints are defined and very minor changes in the locations of environment regions, as demonstrated in section 5.7.3.

We note that SCIP has a modular plugin structure that allows users to define their own constraint types and heuristics. Future work could explore implementing plugins to embed problem-specific knowledge, leading to more efficient pruning and exploration of the search tree. We also briefly explored the use of SCIP's dedicated polyhedral indicator constraint type, to replace manual mixed-integer reformulations of polyhedral disjunctive constraints. This foray did not immediately prove fruitful, but it may be worth exploring SCIP's support for indicator constraints over arbitrary constraint types (superindicator constraints) to model disjunctive second-order cone constraints related to friction cones, and to simplify problem formulation in general.

5.8.3 Whole-body control and application to a physical robot

The approximate nature of the kinematic constraints used for planning is currently the biggest hurdle standing in the way of finding plans that are likely to be executable on a physical robot.

A more subtle problem stems from the fact that the planned trajectory is found on the basis of zero rate of change of angular momentum. We do not actually enforce this as a motion task in the whole-body control framework, because we also desire to keep the orientation of the upper body in check, a conflicting goal. This means that the success of tracking on the full robot, even in simulation, is not guaranteed. Despite the lack of a formal guarantee, we have found that success in simulation is very likely. However, there are occasional issues with rotational foot slip, as a result of a rate of change of vertical angular momentum associated with pelvis orientation tracking combined with fast leg swings. These issues can be mitigated to a certain extent by decreasing pelvis orientation tracking gains. Angular-momentum-aware swing foot trajectory generation could also improve this situation [161].

The proposed approach consumes a description of the local contact environment in the form of a small number of polyhedra. In practice, methods from [162] could be used to extract such a polyhedral environment description, but this integration step is beyond the scope of this thesis.

5.8.4 Applications and future perspectives

There are several possible applications of the presented approach. First, results of an MINLP feasibility problem could be used to warm-start a trajectory optimization involving the full-dimensional dynamics. This may mitigate some of the limitations related to approximate kinematic constraints and lack of orientation information, while potentially providing a significant speedup compared to a cold start.

Second, a library of MINLP trajectory optimization results could be created offline for a variety of scenarios, and used as samples that guide the search for an online policy using function approximation / learning techniques. This approach was

explored in [121] in the context of MIQP, where the value function for an optimal control problem was approximated based on upper and lower bounds on the objective function. However, the slow progress towards establishing nontrivial upper and lower bounds in the case of MINLP with longer time horizons may prove prohibitive.

A compelling alternative is the use of more direct reinforcement learning approaches, which have recently been successfully transferred to the physical robot from the simulation environment in which they were trained [163]. A main advantage of these approaches is that there are fewer restrictions on the problem formulation: the plant system can be considered more of a black box. In addition, the computation is trivial to parallelize on one machine or even distribute across multiple machines. In contrast, state-of-the-art MINLP solvers appear unable to fully exploit even the available threads on a single machine.[29] On the other hand, the approach of [163] has so far only successfully been shown to transfer to a physical quadruped robot, and the reduced margin of error stemming from humanoids robots' reduced base of support may yet prove a significant hurdle.

5.9 Conclusion

This chapter presented a centroidal trajectory optimization approach that relies on the use of dedicated mixed-integer nonlinear program solvers. The approach explicitly takes bilinear constraints associated with whole-body angular momentum into account. The feasibility of the resulting center of mass trajectory is certified by associated contact force trajectories that satisfy friction cone constraints associated with the contact environment, modeled as a set of polyhedral regions. This center of mass trajectory was subsequently as the input to a momentum-based whole-body control framework, and used to achieve locomotion over the terrain in simulation.

[29] We note of course that different problems can still be trivially distributed over available machines, in the context of building a library of MINLP solutions.

Bibliography

[1] T. Koolen, T. De Boer, J. Rebula, A. Goswami, and J. Pratt, "Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models," *The International Journal of Robotics Research*, vol. 31, no. 9, pp. 1094–1113, 2012.

[2] T. McGeer, "Passive dynamic walking," *The International Journal of Robotics Research*, vol. 9, no. 2, pp. 62–82, 1990.

[3] M. J. Coleman and A. Ruina, "An uncontrolled walking toy that cannot stand still," *Physical Review Letters*, vol. 80, no. 16, p. 3658, 1998.

[4] D. G. Hobbelen and M. Wisse, "Limit cycle walking," in *Humanoid Robots, Human-like Machines*, IntechOpen, 2007.

[5] P. Zaytsev, S. J. Hasaneini, and A. Ruina, "Two steps is enough: No need to plan far ahead for walking balance," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6295–6300, IEEE, 2015.

[6] M. Posa, T. Koolen, and R. Tedrake, "Balancing and step recovery capturability via sums-of-squares optimization," in *Robotics: Science and Systems*, pp. 12–16, 2017.

[7] J. Englsberger, T. Koolen, S. Bertrand, J. Pratt, C. Ott, and A. Albu-Schäffer, "Trajectory generation for continuous leg forces during double support and heel-to-toe shift based on divergent component of motion," in *2014 IEEE/RSJ*

International Conference on Intelligent Robots and Systems, pp. 4022–4029, IEEE, 2014.

[8] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131, 2013.

[9] E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski, "The DARPA robotics challenge finals: results and perspectives," *Journal of Field Robotics*, vol. 34, no. 2, pp. 229–240, 2017.

[10] J.-P. Aubin, *Viability theory*. Springer Science & Business Media, 2009.

[11] P.-B. Wieber, "On the stability of walking systems," in *Proceedings of the international workshop on humanoid and human friendly robotics*, 2002.

[12] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[13] P. A. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

[14] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 477–492, Springer, 2004.

[15] M. Korda, D. Henrion, and C. N. Jones, "Convex computation of the maximum controlled invariant set for polynomial control systems," *SIAM Journal on Control and Optimization*, vol. 52, no. 5, pp. 2944–2969, 2014.

[16] A. Majumdar, R. Vasudevan, M. M. Tobenkin, and R. Tedrake, "Convex optimization of nonlinear feedback controllers via occupation measures," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1209–1230, 2014.

[17] J.-P. Aubin, "Viability kernels and capture basins of sets under differential inclusions," *SIAM Journal on Control and Optimization*, vol. 40, no. 3, pp. 853–881, 2001.

[18] C. Mummolo, L. Mangialardi, and J. H. Kim, "Numerical estimation of balanced and falling states for constrained legged systems," *Journal of Nonlinear Science*, vol. 27, no. 4, pp. 1291–1323, 2017.

[19] A. Del Prete, S. Tonneau, and N. Mansard, "Zero step capturability for legged robots in multicontact," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1021–1034, 2018.

[20] R. M. Murray, *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[21] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2008.

[22] V. Duindam, *Port-Based Modeling and Control for Efficient Bipedal Walking Robots*. PhD thesis, University of Twente, 2006.

[23] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous Robots*, pp. 1–16, June 2013.

[24] D. E. Orin and A. Goswami, "Centroidal momentum matrix of a humanoid robot: Structure and properties," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 653–659, IEEE, 2008.

[25] J. E. Pratt, S. Bertrand, and T. Koolen, *Stepping for Balance Maintenance Including Push-Recovery*, pp. 1–48. Dordrecht: Springer Netherlands, 2018.

[26] C. Runge, C. Shupert, F. Horak, and F. Zajac, "Ankle and hip postural strategies defined by joint torques," *Gait & posture*, vol. 10, no. 2, pp. 161–170, 1999.

[27] H. Hirukawa, S. Hattori, S. Kajita, K. Harada, K. Kaneko, F. Kanehiro, M. Morisawa, and S. Nakaoka, "A pattern generator of humanoid robots walk-

ing on a rough terrain," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 2181–2187, IEEE, 2007.

[28] S. Caron, Q.-C. Pham, and Y. Nakamura, "Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5107–5112, IEEE, 2015.

[29] H. Dai and R. Tedrake, "Planning robust walking motion on uneven terrain via convex optimization," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 579–586, IEEE, 2016.

[30] J. Carpentier, R. Budhiraja, and N. Mansard, "Learning feasibility constraints for multi-contact locomotion of legged robots," in *Robotics: Science and Systems*, p. 9p, 2017.

[31] A. K. Valenzuela, *Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain.* PhD thesis, Massachusetts Institute of Technology, 2016.

[32] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 295–302, IEEE, 2014.

[33] G. Nelson, A. Saunders, and R. Playter, "The PETMAN and Atlas robots at Boston Dynamics," *Humanoid Robotics: A Reference*, pp. 169–186, 2019.

[34] S. Kajita and K. Tani, "Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 1405–1411 vol.2, Apr 1991.

[35] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern gener-

ation," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 1, pp. 239–246 vol.1, 2001.

[36] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pp. 200–207, Dec 2006.

[37] S.-H. Lee and A. Goswami, "Reaction mass pendulum (RMP): An explicit model for centroidal angular momentum of humanoid robots," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 4667–4672, IEEE, 2007.

[38] B. Stephens, "Humanoid push recovery," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pp. 589–595, IEEE, 2007.

[39] M. B. Popovic, A. Goswami, and H. Herr, "Ground reference points in legged locomotion: Definitions, biological trajectories and control implications," *The International Journal of Robotics Research*, vol. 24, no. 12, pp. 1013–1032, 2005.

[40] H. Herr and M. Popovic, "Angular momentum in human walking," *Journal of experimental biology*, vol. 211, no. 4, pp. 467–481, 2008.

[41] 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 *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2, pp. 1620–1626, IEEE, 2003.

[42] J. Pratt, T. Koolen, T. De Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus, "Capturability-based analysis and control of legged locomotion, part 2: Application to M2V2, a lower-body humanoid," *The International Journal of Robotics Research*, vol. 31, no. 10, pp. 1117–1133, 2012.

[43] J. Englsberger, C. Ott, M. A. Roa, A. Albu-Schäffer, and G. Hirzinger, "Bipedal walking control based on capture point dynamics," in *2011 IEEE/RSJ Inter-*

national Conference on Intelligent Robots and Systems, pp. 4420–4427, IEEE, IEEE, Sept. 2011.

[44] 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*, pp. 137–142, IEEE, 2006.

[45] R. J. Griffin and A. Leonessa, "Model predictive control for dynamic footstep adjustment using the divergent component of motion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1763–1768, IEEE, 2016.

[46] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the Honda ASIMO humanoid," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, pp. 629–634, IEEE, 2005.

[47] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Online footstep planning for humanoid robots," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 1, pp. 932–937, IEEE, 2003.

[48] D. Witt, "A feasibility study on powered lower-limb prostheses," in *Proceedings of the Institution of Mechanical Engineers, Conference Proceedings*, vol. 183, pp. 18–25, SAGE Publications Sage UK: London, England, 1968.

[49] T. Takenaka, T. Matsumoto, and T. Yoshiike, "Real time motion generation and control for biped robot - 1st report: Walking gait pattern generation," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1084–1091, IEEE, 2009.

[50] J. E. Pratt and S. V. Drakunov, "Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model," in *Robotics*

and Automation, 2007 IEEE International Conference on, pp. 4653–4660, April 2007.

[51] A. L. Hof, M. Gazendam, and W. Sinke, "The condition for dynamic stability," *Journal of Biomechanics*, vol. 38, no. 1, pp. 1–8, 2005.

[52] L. Lanari, S. Hutchinson, and L. Marchionni, "Boundedness issues in planning of locomotion trajectories for biped robots," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 951–958, IEEE, 2014.

[53] J. Pratt and G. Pratt, "Intuitive control of a planar bipedal walking robot," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, vol. 3, pp. 2014–2021, IEEE, 1998.

[54] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Resolved momentum control: Humanoid motion planning based on the linear and angular momentum," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, vol. 2, pp. 1644–1650, IEEE, 2003.

[55] B. Siciliano and J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Proceedings of the 5th International Conference on Advanced Robotics*, vol. 2, pp. 1211–1216, IEEE, 1991.

[56] T. Koolen, J. Smith, G. Thomas, S. Bertrand, J. Carff, N. Mertins, D. Stephen, P. Abeles, J. Englsberger, S. Mccrory, *et al.*, "Summary of Team IHMC's Virtual Robotics Challenge entry," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, (Atlanta, GA), pp. 307–314, IEEE, IEEE, 2013.

[57] T. Koolen, S. Bertrand, G. Thomas, T. De Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt, "Design of a momentum-based control framework and application to the humanoid robot Atlas," *International Journal of Humanoid Robotics*, vol. 13, no. 01, p. 1650007, 2016.

[58] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, pp. 43–53, Feb. 1987.

[59] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 29–43, 2004.

[60] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 04, pp. 505–518, 2005.

[61] L. Sentis, *Synthesis and Control of Whole-Body Behaviors in Humanoid Systems*. Doctor of philosophy, Stanford University, 2007.

[62] M. Ehrgott, *Multicriteria optimization*, vol. 491. Springer Science & Business Media, 2005.

[63] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[64] A. Migdalas, P. M. Pardalos, and P. Värbrand, *Multilevel optimization: algorithms and applications*, vol. 20. Springer Science & Business Media, 2013.

[65] S.-H. Hyon, J. Hale, and G. Cheng, "Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces," vol. 23, pp. 884–898, Oct. 2007.

[66] S.-H. Hyon, R. Osu, and Y. Otaka, "Integration of multi-level postural balancing on humanoid robots," in *Proc. 2009 IEEE Int. Conf. Robot. Automat.*, pp. 1549–1556, IEEE, May 2009.

[67] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Inverse dynamics with optimal distribution of ground reaction forces for legged robots," in *Emerging Trends in*

Mobile Robotics - Proceedings of the 13th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, (Singapore), pp. 580–587, World Scientific Publishing Co. Pte. Ltd., 2010.

[68] 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*, no. 3, (Anchorage, Alaska), pp. 3406–3412, IEEE, 2010.

[69] L. di Gaspero and E. Moyer, "QuadProg++." http://quadprog.sourceforge.net/, 2015.

[70] Gurobi Optimization, Inc., "Gurobi optimizer reference manual." http://www.gurobi.com, 2015.

[71] J. Mattingley and S. Boyd, "CVXGEN: Code generation for convex optimization." http://cvxgen.com/, 2015.

[72] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," in *2018 UKACC 12th International Conference on Control (CONTROL)*, pp. 339–339, IEEE, 2018.

[73] S. Kudoh, T. Komura, and K. Ikeuchi, "The dynamic postural adjustment with the quadratic programming method," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3, pp. 2563–2568, IEEE, 2002.

[74] A. Macchietto, V. Zordan, and C. R. Shelton, "Momentum control for balance," in *ACM Transactions on graphics (TOG)*, vol. 28, p. 80, ACM, 2009.

[75] B. J. Stephens and C. G. Atkeson, "Dynamic balance force control for compliant humanoid robots," in *2010 IEEE/RSJ international conference on intelligent robots and systems*, pp. 1248–1255, IEEE, IEEE, 2010.

[76] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Soueres, and J. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *Robotics, IEEE Transactions on*, vol. 29, no. 2, pp. 346–362, 2013.

[77] L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, and S. Schaal, "Optimal distribution of contact forces with inverse dynamics control," *The International Journal of Robotics Research*, Jan. 2013.

[78] S. Kuindersma, F. Permenter, and R. Tedrake, "An efficiently solvable quadratic program for stabilizing dynamic locomotion," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, (Hong Kong, China), May 2014.

[79] A. Herzog, L. Righetti, F. Grimminger, P. Pastor, and S. Schaal, "Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics," in *Proceedings of the IEEE International Conference on Intelligent Robotics Systems*, 2014.

[80] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based full body control for the DARPA Robotics Challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[81] S.-H. Lee and A. Goswami, "Ground reaction force control at each foot : A momentum-based humanoid balance controller for non-level and non-stationary ground," in *Proc. 2010 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 3157–3162, 2010.

[82] S.-H. Lee and A. Goswami, "A momentum-based balance controller for humanoid robots on non-level and non-stationary ground," *Autonomous Robots*, vol. 33, no. 4, pp. 399–414, 2012.

[83] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. Wiley New Jersey, 2006.

[84] F. Bullo and R. Murray, "Proportional derivative (PD) control on the Euclidean group," tech. rep., California Institute of Technology, 1995.

[85] N. Pollard and P. Reitsma, "Animation of humanlike characters: Dynamic motion filtering with a physically plausible contact model," in *Yale Workshop on Adaptive and Learning Systems*, 2001.

[86] D. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1, pp. 162–169, IEEE, 2000.

[87] S. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," tech. rep., University of California, San Diego, 2004.

[88] J. Y. Luh, M. W. Walker, and R. P. Paul, "On-line computational scheme for mechanical manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, no. 2, pp. 69–76, 1980.

[89] T. Koolen and R. Deits, "Julia for robotics: Simulation and real-time control in a high-level programming language," *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[90] M. Johnson, B. Shrewsbury, S. Bertrand, T. Wu, D. Duran, M. Floyd, P. Abeles, D. Stephen, N. Mertins, A. Lesman, *et al.*, "Team IHMC's lessons learned from the DARPA Robotics Challenge Trials," *Journal of Field Robotics*, vol. 32, no. 2, pp. 192–208, 2015.

[91] J. Englsberger and C. Ott, "Integration of vertical COM motion and angular momentum in an extended capture point tracking controller for bipedal walking," in *IEEE-RAS International Conference on Humanoid Robots*, (Osaka, Japan), 2012.

[92] O. E. Ramos and K. Hauser, "Generalizations of the capture point to nonlinear center of mass paths and uneven terrain," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pp. 851–858, Nov 2015.

[93] A. Shiriaev, J. W. Perram, and C. Canudas-de Wit, "Constructive tool for orbital stabilization of underactuated nonlinear systems: Virtual constraints approach," *Automatic Control, IEEE Transactions on*, vol. 50, no. 8, pp. 1164–1176, 2005.

[94] B. F. Caviness and J. R. Johnson, *Quantifier elimination and cylindrical algebraic decomposition*. Springer Science & Business Media, 1998.

[95] A. Seidenberg, "A new decision method for elementary algebra," *Annals of Mathematics*, pp. 365–374, 1954.

[96] C. W. Brown, "QEPCAD B: a program for computing with semi-algebraic sets using CADs," *ACM SIGSAM Bulletin*, vol. 37, no. 4, pp. 97–108, 2003.

[97] A. Strzeboński, "Solving systems of strict polynomial inequalities," *Journal of Symbolic Computation*, vol. 29, no. 3, pp. 471–480, 2000.

[98] X. Da, O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle, "From 2D design of underactuated bipedal gaits to 3D implementation: Walking with speed tracking," *IEEE Access*, vol. 4, pp. 3469–3478, 2016.

[99] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.

[100] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous Robots*, vol. 40, no. 3, pp. 429–455, 2016.

[101] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[102] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 794–807, 2008.

[103] H. Dai *et al.*, *Robust multi-contact dynamical motion planning using contact wrench set*. PhD thesis, Massachusetts Institute of Technology, 2016.

[104] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3D COM accelerations," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 550–557, IEEE, 2016.

[105] B. Ponton, A. Herzog, S. Schaal, and L. Righetti, "A convex model of humanoid momentum dynamics for multi-contact motion generation," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 842–849, IEEE, 2016.

[106] S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha, and N. Mansard, "An efficient acyclic contact planner for multiped robots," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 586–601, 2018.

[107] S. Tonneau, N. Mansard, C. Park, D. Manocha, F. Multon, and J. Pettré, "A reachability-based planner for sequences of acyclic contacts in cluttered environments," in *The International Journal of Robotics Research*, pp. 287–303, Springer, 2018.

[108] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 43, 2012.

[109] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[110] A. Herzog, S. Schaal, and L. Righetti, "Structured contact force optimization for kino-dynamic motion generation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2703–2710, IEEE, 2016.

[111] E. Todorov and W. Li, "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proceedings of the 2005, American Control Conference, 2005.*, pp. 300–306, IEEE, 2005.

[112] 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*, pp. 4906–4913, IEEE, 2012.

[113] F. Farshidian, E. Jelavic, A. Satapathy, M. Giftthaler, and J. Buchli, "Real-time motion planning of legged robots: A model predictive control approach," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 577–584, IEEE, 2017.

[114] M. Neunert, F. Farshidian, A. W. Winkler, and J. Buchli, "Trajectory optimization through contacts and automatic gait discovery for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1502–1509, 2017.

[115] J. Carpentier and N. Mansard, "Analytical derivatives of rigid body dynamics algorithms," in *Robotics: Science and Systems (RSS 2018)*, 2018.

[116] B. Plancher and S. Kuindersma, "A performance analysis of parallel differential dynamic programming on a GPU," in *International Workshop on the Algorithmic Foundations of Robotics (WAFR), Merida, Mexico*, 2018.

[117] S. Neuman, T. Koolen, J. Drean, J. Miller, and S. Devadas, "Benchmarking and workload analysis of robot dynamics algorithms," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (Accepted)*, 2019.

[118] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 279–286, IEEE, 2014.

[119] A. Ibanez, P. Bidaud, and V. Padois, "Emergence of humanoid walking behaviors from mixed-integer model predictive control," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4014–4021, IEEE, 2014.

[120] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, 2017.

[121] R. Deits, T. Koolen, and R. Tedrake, "LVIS: Learning from value function intervals for contact-aware robot controllers," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7762–7768, IEEE, 2019.

[122] R. Misener, J. P. Thompson, and C. A. Floudas, "APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes," *Computers & Chemical Engineering*, vol. 35, no. 5, pp. 876–892, 2011.

[123] A. Gleixner, M. Bastubbe, L. Eifler, T. Gally, G. Gamrath, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. E. Lübbecke, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, C. Schubert, F. Serrano, Y. Shinano, J. M. Viernickel, M. Walter, F. Wegscheider, J. T. Witt, and J. Witzig, "The SCIP Optimization Suite 6.0," ZIB-Report 18-26, Zuse Institute Berlin, July 2018.

[124] M. R. Kılınç and N. V. Sahinidis, "Exploiting integrality in the global optimization of mixed-integer nonlinear programming problems with baron," *Optimization Methods and Software*, vol. 33, no. 3, pp. 540–562, 2018.

[125] P. Belotti, "COUENNE: a user's manual," tech. rep., Technical report, Lehigh University, 2009.

[126] L. Schewe and M. Schmidt, "Computing feasible points for binary MINLPs with MPECs," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 95–118, 2019.

[127] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[128] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction," *International Journal for Numerical Methods in Engineering*, vol. 39, no. 15, pp. 2673–2691, 1996.

[129] S. Scholtes and M. Stöhr, "How stringent is the linear independence assumption for mathematical programs with complementarity constraints?," *Mathematics of Operations Research*, vol. 26, no. 4, pp. 851–863, 2001.

[130] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, Oct 2012.

[131] M. Neunert, M. Stäuble, M. Giftthaler, C. D. Bellicoso, J. Carius, C. Gehring, M. Hutter, and J. Buchli, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1458–1465, July 2018.

[132] G. Cornuéjols, "Valid inequalities for mixed integer linear programs," *Mathematical Programming*, vol. 112, no. 1, pp. 3–44, 2008.

[133] T. Berthold, *Heuristic algorithms in global MINLP solvers.* Verlag Dr. Hut Munich, 2014.

[134] T. Achterberg, T. Koch, and A. Martin, "Branching rules revisited," *Operations Research Letters*, vol. 33, no. 1, pp. 42–54, 2005.

[135] D. Bertsimas and J. N. Tsitsiklis, *Introduction to linear optimization*, vol. 6. Athena Scientific Belmont, MA, 1997.

[136] R. Gomory, "An algorithm for the mixed integer problem," tech. rep., RAND Corp., Santa Monica, CA, 1960.

[137] E. Balas, S. Ceria, G. Cornuéjols, and N. Natraj, "Gomory cuts revisited," *Operations Research Letters*, vol. 19, no. 1, pp. 1–9, 1996.

[138] R. A. Stubbs and S. Mehrotra, "A branch-and-cut method for 0-1 mixed convex programming," *Mathematical programming*, vol. 86, no. 3, pp. 515–532, 1999.

[139] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter, "An algorithmic framework for convex mixed integer nonlinear programs," *Discrete Optimization*, vol. 5, no. 2, pp. 186 – 204, 2008. In Memory of George B. Dantzig.

[140] E. Danna, E. Rothberg, and C. Le Pape, "Exploring relaxation induced neighborhoods to improve MIP solutions," *Mathematical Programming*, vol. 102, no. 1, pp. 71–90, 2005.

[141] J. Forrest and R. Lougee-Heimer, "Cbc user guide," in *Emerging theory, methods, and applications*, pp. 257–277, INFORMS, 2005.

[142] A. Makhorin, "GNU linear programming kit," *Moscow Aviation Institute, Moscow, Russia*, vol. 38, 2001.

[143] P. Bonami and J. Lee, "BONMIN user's manual," *Numer Math*, vol. 4, pp. 1–32, 2007.

[144] T. Koch, T. Achterberg, E. Andersen, O. Bastert, T. Berthold, R. E. Bixby, E. Danna, G. Gamrath, A. M. Gleixner, S. Heinz, *et al.*, "MIPLIB 2010," *Mathematical Programming Computation*, vol. 3, no. 2, p. 103, 2011.

[145] E. D. Andersen and K. D. Andersen, "The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm," in *High performance optimization*, pp. 197–232, Springer, 2000.

[146] E. Balas, "Disjunctive programming," in *Annals of Discrete Mathematics*, vol. 5, pp. 3–51, Elsevier, 1979.

[147] J. P. Vielma, "Mixed integer linear programming formulation techniques," *Siam Review*, vol. 57, no. 1, pp. 3–57, 2015.

[148] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part i - convex underestimating problems," *Mathematical programming*, vol. 10, no. 1, pp. 147–175, 1976.

[149] R. C. Jeroslow, "There cannot be any algorithm for integer programming with quadratic constraints," *Operations Research*, vol. 21, no. 1, pp. 221–224, 1973.

[150] M. Tawarmalani and N. V. Sahinidis, "A polyhedral branch-and-cut approach to global optimization," *Mathematical Programming*, vol. 103, no. 2, pp. 225–249, 2005.

[151] R. Misener and C. A. Floudas, "ANTIGONE: algorithms for continuous/integer global optimization of nonlinear equations," *Journal of Global Optimization*, vol. 59, no. 2-3, pp. 503–526, 2014.

[152] Y. Lin and L. Schrage, "The global solver in the LINDO API," *Optimization Methods & Software*, vol. 24, no. 4-5, pp. 657–668, 2009.

[153] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.

[154] R. Fletcher and S. Leyffer, "User manual for filterSQP," *Numerical Analysis Report NA/181, Department of Mathematics, University of Dundee, Dundee, Scotland*, vol. 35, 1998.

[155] E. M. L. Beale and J. A. Tomlin, "Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables," *OR*, vol. 69, no. 447-454, p. 99, 1970.

[156] J. P. Vielma and G. L. Nemhauser, "Modeling disjunctive constraints with a logarithmic number of binary variables and constraints," *Mathematical Programming*, vol. 128, no. 1-2, pp. 49–72, 2011.

[157] J. Huchette and J. P. Vielma, "Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools," *arXiv preprint arXiv:1708.00050*, 2017.

[158] G. Hendel, "Adaptive large neighborhood search for mixed integer programming," 2018.

[159] A. Lodi and A. Tramontani, "Performance variability in mixed-integer programming," in *Theory Driven by Influential Applications*, pp. 1–12, INFORMS, 2013.

[160] E. Klotz and A. M. Newman, "Practical guidelines for solving difficult mixed integer linear programs," *Surveys in Operations Research and Management Science*, vol. 18, no. 1-2, pp. 18–32, 2013.

[161] T. Seyde, A. Shrivastava, J. Englsberger, S. Bertrand, J. Pratt, and R. J. Griffin, "Inclusion of angular momentum during planning for capture point based walking," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1791–1798, IEEE, 2018.

[162] J. P. Marion, *Perception methods for continuous humanoid locomotion over uneven terrain*. PhD thesis, Massachusetts Institute of Technology, 2016.

[163] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.