Natural, Efficient Walking for Compliant Humanoid Robots

Robert J. Griffin

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mechanical Engineering

Alexander Leonessa, Chair,
Alan T. Asbeck,
Steve C. Southward,
Craig A. Woolsey,
Pratap Tokekar

September 20, 2017
Blacksburg, Virginia

Keywords: Humanoid robots, Bipedal walking
Copyright 2017, Robert J. Griffin

Natural, Efficient Walking for Compliant Humanoid Robots

Robert J. Griffin

ABSTRACT

Bipedal robots offer a uniquely flexible platform capable of navigating complex, human-centric environments. This makes them ideally suited for a variety of missions, including disaster response and relief, emergency scenarios, or exoskeleton systems for individuals with disabilities. This, however, requires significant advances in humanoid locomotion and control, as they are still slow, unnatural, inefficient, and relatively unstable. The work of this dissertation improves the state of the art with the aim was of increasing the robustness and efficiency of these bipedal walking platforms.

We present a series of control improvements to enable reliable, robust, natural bipedal locomotion that was validated on a variety of bipedal robots using both hardware and simulation experiments.

A huge part of reliable walking involves maximizing the robot's control authority. We first present the development of a model predictive controller to both control the ground reaction forces and perform step adjustment for walking stabilization using a mixed-integer quadratic program. This represents the first model predictive controller to include step rotation in the optimization and leverage the capabilities of the time-varying divergent component of motion for navigating rough terrain. We also analyze the potential capabilities of model predictive controllers for the control of bipedal walking.

As an alternative to standard trajectory optimization-based model predictive controls, we present several optimization-based control schemes that leverage more traditional bipedal walking control approaches by embedding a proportional feedback controller into a quadratic program. This controller is capable of combining multiple feedback mechanisms: ground reaction feedback (the "ankle strategy"), angular momentum (the "hip strategy"), swing foot speed up, and step adjustment. This allows the robot to effectively shift its weight, pitch its torso, and adjust its feet to retain balance, while considering environmental constraints, when available.

To enable the robot to walk with straightened legs, we present a strategy that insures that the dynamic plans are kinematically and dynamically feasible to execute using straight legs. The effects of timing on dynamic plans are typically ignored, resulting in them potentially requiring significantly bending the legs during execution. This algorithm modifies the step timings to insure the plan can be executed without bending the legs beyond certain angle, while leaving the desired footsteps unmodified. To then achieve walking with straight legs we then presented a novel approach for indirectly controlling the center of mass height through the knee angles. This avoids complicated height planning techniques that are both computationally expensive and often not general enough to consider variable terrain by effectively biasing the solution of the whole-body controller towards using straighter legs. To incorporate the toe-off motion that is essential to both natural and straight leg walking, we also present a strategy for toe-off control that allows it to be an emergent behavior of the whole-body controller.

The proposed approach was demonstrated through a series of simulation and experimental results on a variety of platforms. Model predictive control for step adjustment and rough

terrain is illustrated in simulation, while the other step adjustment strategies and straight leg walking approaches are presented recovering from external disturbances and walking over a variety of terrains in hardware experiments. We discuss many of the practical considerations and limitations required when porting simulation-based controller development to hardware platforms. Using the presented approaches, we also demonstrated an important concept: using whole-body control frameworks, not every desired motion need be directly commanded. Many of these motions, such as toe-off, may simply be emergent behaviors that result by attempting to satisfy other objectives, such as desired reaction forces. We also showed that optimization is a very powerful tool for walking control, able to determine both stabilizing inputs and joint torques.

Natural, Efficient Walking for Compliant Humanoid Robots

Robert J. Griffin

GENERAL AUDIENCE ABSTRACT

Bipedal robots offer a uniquely flexible platform capable of navigating the complex, human-centric environment that we live in. This makes them ideally suited for a variety of missions, including disaster response and relief, emergency scenarios, or exoskeleton systems for individuals with disabilities. This, however, requires significant advances in humanoid locomotion and control, as they are still slow, unnatural, inefficient, and relatively unstable. The work of this dissertation aims to increase the robustness and efficiency of these bipedal walking platforms.

To increase the overall stability of the robot while walking, we aimed to develop new control schemes that incorporate more of the same balance strategies used by people. These include the adjustment of ground reaction forces (the "ankle strategy", shifting weight), angular momentum (the "hip strategy", pitching the torso and windmilling the arms), swing foot speed up, and step adjustment. Using these approaches, the robot is able to walk much more stably.

With the ability to use human-like control strategies, the next step is to develop appropriate methods to allow it to walk with straighter legs. Without correct step timing, it may be necessary at times to significantly bend the knees to take the specified step. We develop an approach to adjust the step timing to decrease the required knee bend of the robot. We then present an approach for indirectly controlling the robot height through the knee angles. This avoids traditional complicated height planning techniques that are both computationally hard and not general enough to consider complex terrain. To incorporate the toe-off motion that is essential to both natural and straight leg walking, we also present a new strategy for toe-off that allows it to emerge natural from the controller.

We present the proposed approach through a series of simulation and experimental results on several robots and in several environments. We discuss many of the practical considerations and limitations required when porting simulation-based controller development to hardware platforms. Using the presented approaches, we also demonstrated an important concept: using whole-body control frameworks, not every desired motion need be directly commanded. Many of these motions, such as toe-off, may simply be emergent behaviors that result by attempting to satisfy other objectives, such as desired reaction forces. We also showed that optimization is a very powerful tool for walking control, able to determine both stabilizing inputs and joint torques.

Acknowledgments

First, I would like to thank my wife, Kara. Thank you for putting up with the years of late nights, the weeks of travel, and the months of me being away. You have been more supportive and more understanding than I ever could have expected or deserved. Without your encouragement and love, I would have never made it through. You are the love of my life, and thank you, again, for being undeniably who you are: a proud, caring, woman of God.

Thank you to my parents, Jim and Jan, for their never-wavering support and belief in me. So much of who I am today I owe to you both. Your never-ceasing encouragement throughout the years helped me to believe in myself, and gave me the courage to pursue what I otherwise could not have.

Thanks to my advisor, Dr. Alexander Leonessa, who's provided me with invaluable advice and guidance throughout my graduate career. I appreciate the opportunities you have given me, and your never-hesitating willingness to provide help and input.

Thank you to the rest of my committee, Dr. Alan Asbeck, Dr. Steve Southward, Dr. Craig Woolsey, and Dr. Pratap Tokekar, for all their feedback and advice.

Thank you to Mike Hopkins and Viktor Orkehov, who helped get me started and acted as my mentor during my first months with TREC. Your feedback and direction helped pave the way for the rest of my research. Without your friendship and advice, I might have never escaped.

Thank you to Georg Wiedebach, Sylvain Bertrand, Jerry Pratt and Peter Neuhaus for getting me involved at IHMC, and helping expand the way I looked at problems. Without you all, I'd still be stuck writing in Lua.

Thanks to the entire Virginia Tech DRC and Saffir Team, including James Burton, Graham Cantor-Cooke, Lakshitha Dantanarayana, Graham Day, Oliver Ebeling-Koning, Eric Hahn, Mike Hopkins, Coleman Knabe, Jordan Neal, Jack Newton, Chris Nogales, Viktor Orekhov, John Peterson, Mike Rouleau, John Seminatore, Yoonchang Sung, Jacob Webb, Nick Wittenstein, and Jason Ziglar. You guys are awesome, and together, we did something worth remembering.

Thanks to the IHMC Cybathlon Team, Tyson Cobb, Travis Craig, Mark Daniel, Nick van Dijk, Jeremy Gines, Koen Kramer, Shriya Shah, Olger Siebinga, Jesper Smith, and Peter Neuhaus. We rocked it, guys, and it was unforgettable.

I would also like to thank my extended family and friends for all their support.

Lastly, I would like to thank the Lord for the tremendous blessings and opportunities He has given me, without which none of this would be have been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bipedal robots have been studied for centuries, with the first credited design appearing in the Chinese Liezi text in 250 BC, which described the first bipedal automaton [1]. The ability to build walking robotic platforms is relatively new, however, with the first full-scale humanoid robot, Wabot–1, being built by Waseda Univeristy in 1973 [2]. Since this time, bipedal robotic platforms have come a long way, but have a long way still before they can be utilized in every-day life. Their effectiveness is ultimately governed by the underlying dynamics of walking and the controllers designed around these dynamics. With proper hardware and controls, the potential of platforms such as humanoid robots and exoskeletons, is tremendous. In this work, contributions towards moving these platforms out of laboratory settings and into the world are proposed.

1.1 Motivation for Study

Among the countless number of applications for robotics, disaster response has the potential to be completely transformed through its incorporation. While some attempts such as the Endeavor Robotics' Packbot [3] have attempted to address this, there is still significant room for improvement. The Packbot is an excellent example of the most common form of mobile robot, which typically us aerial or wheeled platforms. These robots, however, are not necessarily the best suited design for disaster response.

To understand typical design requirements for emergency response robots, we can look at two specific examples. In 2011, the Fukushima Daiichi nuclear power plant suffered multiple reactor core meltdowns from a tsunami caused by a high magnitude earthquake. Direct human intervention was impossible due to the resulting radioactive environment. Fieldable, response-ready robots would have not only helped prevent and control the reactor meltdown, but also could have greatly improved the effectiveness of the subsequent cleanup efforts. Similarly, shipboard firefighting could be greatly benefited by disaster response robots. Ship

fires represent one of the most dangerous peace-time hazards for equipment, ships, and personnel. The combination of a confined, metal enclosed space with high amounts of heat and smoke produce a particularly hazardous and difficult to control environment for first-responders. Robots designed for firefighting could augment first responders' abilities through entering dangerous spaces more closely and for longer durations, in addition to helping navigate and analyze smoke-filled environments.

Recognizing these needs, two different government agencies have funded research aimed at developing readily-available emergency response robots. The Office of Naval Research (ONR) started the Shipboard Autonomous Firefighting Robot (SAFFiR) program with Virginia Tech to develop humanoid robots for damage control and maintenance onboard Navy ships. Following the Fukushima Daiichi disaster, DARPA announced their newest grand-challenge, the DARPA Robotics Challenge (DRC), with tasks inspired by a nuclear power plant disaster scenario. Both of these programs feature the same core elements, requiring robots capable of functioning as first-responders. They take-place in man-made environments, Navy ships and nuclear power plants, which are designed around the human morphology. Both require the use of a variety of tools to perform different tasks, such as using fire hoses and turning valves. Lastly, they feature highly unpredictable and changing terrain. While wheeled platforms can be designed to very effectively address a wide variety specific circumstances, emergency response scenarios require flexibility to cope with the inevitable uncertainty and changing needs posed by disasters. For this, bipedal humanoid robots are uniquely well-suited. The human form-factor allows the robot to potentially utilize the environment and tools designed for people with the dexterity and adroitness of humans.

Bipedal robots extend beyond humanoid robots, however, and includes powered exoskeletons. One of the greatest challenges to the sufferers of hemiplegia and paraplegia is overcoming mobility limitations, which is commonly addressed using wheelchairs. However, wheelchairs introduce entirely new medical challenges and concerns, while still having reduced mobility in man-made environments designed around bipedal locomotion. While seated in a wheelchair in the normal position, the legs are not properly loaded which can lead to a degradation of bone density [4]. This can be alleviated through standing with the use of braces [5], though this does not typically help increase mobility. Sitting for prolonged periods of time has also been shown to cause pressure sores, which can lead to a variety of other medical issues [6, 7]. Lack of use can also lead to muscle atrophy and joint contracture in the legs [8]. In addition, the dependence on the use of arms for movement can result in shoulder overuse syndrome [9, 10]. With advancements in technology, however, more options than just wheelchairs are becoming available, such as function electrical stimulation (FES) or robotic exoskeletons, which can augment or even replace traditional therapy. Besides just the physical medical benefits, with sufficient research and development, robotic exoskeletons have the potential to restore user mobility. This fundamentally relies on developments in bipedal walking robots.

The underlying dynamical systems for humanoid robots and powered exoskeletons can be described very similarly, with the biggest challenges to designing walking controllers coming from the nature of the dynamics. This system can be characterized by:

- Nonlinear, multibody dynamics,
- High dimensionality,
- Redundant kinematics,
- Hybrid system between contact phases,
- Unilateral friction-limited contacts,
- Underactuated floating-base,
- Actuation limitations through torque and speed limits.

Even when assuming rigid-body dynamics with inflexible, solid links, the dynamics themselves remain highly nonlinear. Thus, even somewhat simplified models pose a challenging problem for controls engineers. It is one, though, that nature is particularly adept at solving, with legged animals capable of highly dynamic, robust, and compliant locomotion [11].

While considerable progress has been made from the first humanoid robots and powered exoskeletons [2, 12], they still are relatively unnatural and inefficient during operation, and very liable for failure. This body of work focuses on achieving natural-like walking with straighter legs. To be successful, this requires a wide variety of improvements, from the control of the robot's height to achieve straight legged walking, to appropriate dynamic planning, to control authority improvements for both straight legged and bent legged walking. Through this, it is our hope to create robots capable of not just satisfying competition requirements such as the DRC and 2016 Cybathlon, but are employable in everyday life.

1.2 Background

The literature on walking models and control algorithms for bipedal robots is quite extensive. Here, we will provide a survey of some of the underlying models often employed in walking analysis, as well as dynamic and compliant control techniques for locomotion and an overview of different compliant humanoids, quadrupeds, and exoskeletons.

1.2.1 Walking Models

Before we present an overview of control strategies employed for dynamic locomotion, it is worthwhile to review the underlying dynamic models from which these strategies are derived. As the full rigid-body dynamics have very high dimensionality, they are often reduced to much simpler center of mass (CoM) dynamics. The net Newtonian constraints on the rigid-body dynamics are fully described by the CoM, making it a valid model for examining the

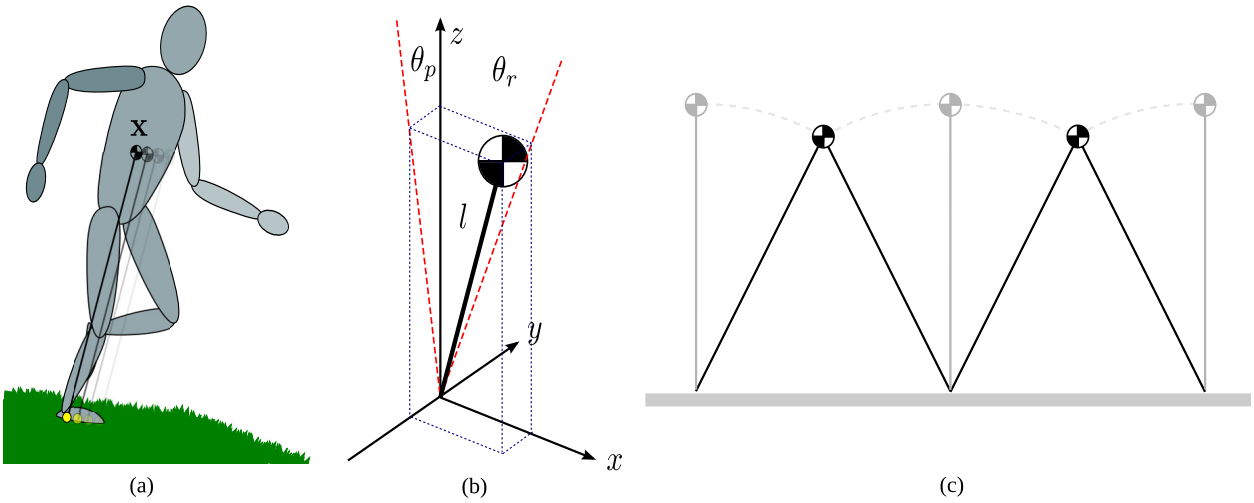


Figure 1.1: (a) Rigid body model highlighting CoM. (b) Inverted pendulum model representation. (c) Inverted pendulum walking gait.

effects of the contact forces on the robotic system. The different models attempt to describe the walking gait in a simple manner. Here, we will present several of these reduced-order models.

Inverted Pendulum

One of the simplest models of walking is to describe the motion as an inverted pendulum, which is thoroughly outlined in [13] and illustrated in Figure 1.1. This model assumes that the walking gait is composed of a series of valuting motions, with an instantaneous transfer between support phases, as shown in Figure 1.1(c). The governing constraint on the pendulum dynamics is then defined by the kinematics

$$\mathbf{x} = \mathbf{f}(\mathbf{q}) = l \begin{bmatrix} \sin\theta_p \\ -\sin\theta_r \\ \sqrt{1-\sin^2\theta_p - \sin^2\theta_r} \end{bmatrix}, \tag{1.1}$$

where $\mathbf{x}$ is the location of the center of mass and $\mathbf{q} = \begin{pmatrix} \theta_r, & \theta_p, & l \end{pmatrix}$. The CoM dynamics can then be defined by

$$m\ddot{\mathbf{x}} = \left(\mathbf{J}^T\right)^{\#} \begin{bmatrix} \tau_r \\ \tau_p \\ f \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}, \tag{1.2}$$

where m is the mass, $\#$ is the pseudo-inverse operator, τ_r and τ_p correspond to torques about θ_r and θ_p, f is the force along the direction of the pendulum, g is gravity, and $\mathbf{J}$ is the

kinematic Jacobian, $\mathbf{J} = \frac{\delta \mathbf{f}}{\delta \mathbf{q}}$. From Equation 1.2, the dynamics in the x-y plane are defined as

$$m\left(z\ddot{x} - x\ddot{z}\right) = \frac{\sqrt{1 - \sin^2\theta_r - \sin^2\theta_p}}{\cos\theta_p}\tau_p + mgx, \tag{1.3}$$

$$m\left(-z\ddot{y} + y\ddot{z}\right) = \frac{\sqrt{1 - \sin^2\theta_r - \sin^2\theta_p}}{\cos\theta_r}\tau_r - mgy. \tag{1.4}$$

This shows a coupling between the x and y dynamics and the z dynamics, but, notably, no coupling between x and y.

If we assume that there is no torque about the base of the pendulum, i.e. $\tau_p = \tau_r \equiv 0$, Equation 1.3 and Equation 1.4 simplify to

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \frac{1}{z}\begin{bmatrix} gx + x\ddot{z} \\ gy + y\ddot{z} \end{bmatrix}. \tag{1.5}$$

This can be likened to a feedback controller that moves the base of the pendulum to maintain balance. By moving the base, the controller utilizes the passive dynamics defined by Equation 1.5 for stabilization, rather than directly injecting energy for balance through torques at the base.

Linear Inverted Pendulum

While the inverted pendulum model leads to a vastly simpler and more tractable set of equations than than the full rigid-body dynamics, they are still coupled in the vertical and horizontal directions, leading to nonlinearities in the dynamics. For further simplification, the inverted pendulum can be linearized by limiting the motion to a plane by requiring that $\ddot{z} \equiv 0$, leading to the canonical linear inverted pendulum model (LIPM) [13]. Applying this constraint to Equation 1.3 and Equation 1.4 leads to

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \frac{g}{z_{com}}\begin{bmatrix} x \\ y \end{bmatrix} + \frac{\sqrt{1 - \sin^2\theta_r - \sin^2\theta_p}}{mz_{com}}\begin{bmatrix} \frac{1}{\cos\theta_p}\tau_p \\ -\frac{1}{\cos\theta_r}\tau_r \end{bmatrix}. \tag{1.6}$$

As can be seen, these dynamics are still nonlinear with respect to the control input at the base, τ_r and τ_p. However, the passive dynamics are, in fact, linear. This leads to a linear control system when moving the base of the pendulum, $\mathbf{p}$, as described earlier.

Other Walking Models

A variety of other models have been proposed to capture the walking gait. The most basic one for modeling the passive dynamics is the rimless wheel, shown in Figure 1.2(a), first proposed for walking in [14]. This model encapsulates the idea of the inverted pendulum gait

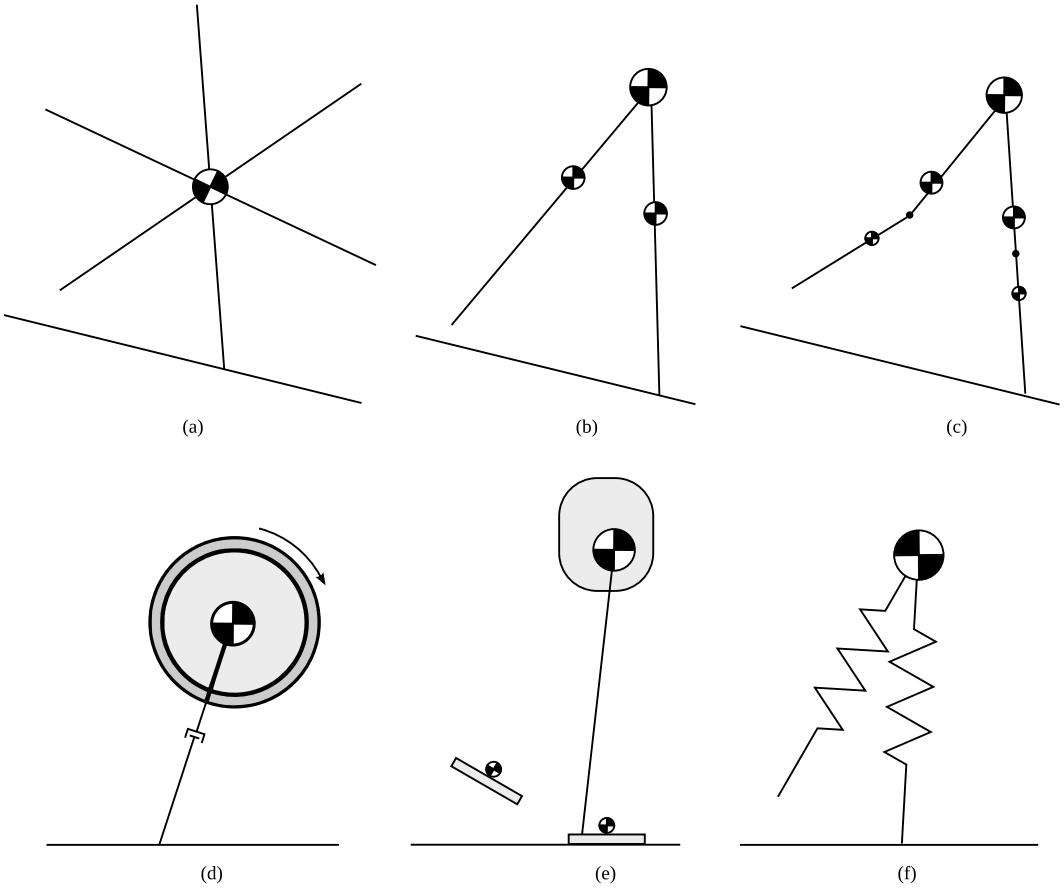


Figure 1.2: (a) Rimless wheel model [14]. (b) Compass gait walker [15]. (c) Kneed walker [16]. (d) Linear inverted pendulum with flywheel model [17]. (e) Dynamic model with three masses [18]. (f) Bipedal spring-mass model [19]

shown in Figure 1.1(c), with instantaneous transfers between support phases. The compass gait walker, shown in Figure 1.2(b), modifies the rimless wheel model by incorporating the dynamics of the swing leg. This is done by adding a point mass to each of the legs, resulting in equivalent dynamics to an Acrobot, with a double inverted-pendulum. A revolute joint is then added at the hip, while the swing leg retracts during each step to clear the ground [15]. Expanding on the compass walker, the knee walker, shown in Figure 1.2(c), adds a knee joint to each of the legs [16]. This model switches between a two-link and a three-link pendulum model throughout the gait cycle, with the stance leg always being locked, and the swing leg straightening at a certain point in the swing phase. This fairly accurately captures many of the essentials of walking, particularly when analyzing passive-dynamic walkers. However, when these models are applied to higher dimensional systems, we do not believe that much additional information is gained over the simple LIPM.

A modification to the classic LIPM was proposed in [17], where a flywheel was introduced at

the center of mass, shown in Figure 1.2(d). This addition made an analysis of the inclusion of angular momentum possible. This proved critical for illustrating the differences between the centroidal moment pivot (CMP) point, and the ZMP, which will be elaborated on later in subsection 1.2.2. Takenaka et al. introduced the three mass model, shown in Figure 1.2(e), which includes point masses to account for the leg inertias. This was used to modify the ZMP dynamics improve control of the CoM. Lastly, the spring loaded inverted pendulum (SLIP) model (Figure 1.2(f)), while commonly used for running, has also been extended to explain the compliant leg motions exhibited during walking [19].

While this is by no means a comprehensive list of walking models, it does represent the additional complexity that improvements make, when compared to the LIPM. The LIPM enjoys a long history of applications, however, and we use it throughout this work as a basis for the underlying walking dynamics, particularly during the swing phase.

1.2.2 Dynamic Locomotion Control

ZMP/CoP Control

The zero moment point (ZMP) can be defined as the point on the ground at which the torque produced by the inertial and gravitation forces is parallel to the ground [20], resulting in the robot not tipping. The ZMP has also been shown to be formulaically equivalent to the center of pressure (CoP) when the ground is flat [21], allowing control of the ZMP to be achieved by closing the loop around the measured CoP. Intuitively, the ZMP can be thought of as the base of inverted pendulum, $\mathbf{p}$. Therefore, by using the ZMP location as the control input, the system behaves as the classic cart-table problem [22], as described in section 1.2.1. The ZMP dynamics can be defined from Equation 1.6 simply as

$$\ddot{\mathbf{x}} = \frac{F_z}{m} \begin{bmatrix} \frac{1}{z_{com}} (x - p_x) \\ \frac{1}{z_{com}} (y - p_y) \\ 1 - \frac{mg}{F_z} \end{bmatrix}, \tag{1.7}$$

where $\mathbf{x}$ is the CoM location. This formulation is so widely used for two reasons. First, its dynamics are simple and linear, making the computation of trajectories relatively easy. Second, as long as the desired ZMP remains within the support polygon, the reference trajectories are physically achievable.

When using ZMP-based trajectory tracking control, the desired ZMP trajectory is typically chosen based on the available or prescribed footholds. From there, a CoM trajectory is often calculated that produces the desired ZMP trajectory and satisfies the boundary conditions [23, 22]. Kajita et al. [13] achieved this through pattern generation of CoM motions based on an analytical solution of the LIPM dynamics. Preview control was first utilized in [22] to minimize the CoM jerk as defined by the derivative of the ZMP dynamics while

tracking a reference ZMP trajectory. Preview control was also used by the authors in [24] to adjust the future foot positions and compensate for ZMP error. Wieber proposed a full constrained quadratic program (QP) for model predictive control (MPC) in [25] that extended [22] to track desired ZMP trajectories. This was then expanded in [26] by adding constraints on the ZMP position to the QP, and further extended in [27] to allow for footstep adjustment in the MPC. In almost all these approaches, the controllers utilize an inverse kinematics solver to determine desired joint positions to achieve the reference CoM location while satisfying the ZMP conditions. When implementing on hardware, this typically uses high-impedance, position controlled robots to accurately track the joint trajectories. It is important to note that the reference joint trajectories generated from the ZMP trajectory may cause the robot to fall by design, even if the ZMP remains inside the support polygon. While the ZMP is often used as a stability criterion, it is a necessary (via physical constraints) but not sufficient condition to avoid falling [28].

The recent surge in force-controlled compliant robots, however, has lead to several alternative implementations of ZMP controllers that are not reliant on the use of inverse kinematics. These robots are particularly amenable to momentum-control schemes, enabling different ZMP control approaches. The ZMP dynamics have been formulated in an infinite horizon time-varying LQR problem in [29], whose cost-to-go is utilized as a momentum controller embedded in the quadratic programmed whole-body controller in [30]. Trajectories are also generated from the ZMP dynamics using differential dynamic programming in [31], which utilizes a whole-body controller to track the desired CoP objective.

Passive Dynamics

While the ZMP approach to controlling dynamic walking relies on actively maintaining balance through control inputs, there are a class of walkers capable of dynamically walking using only passive dynamics for stabilization. Passive-dynamic walkers have shown to be capable of walking in a straight line down slopes with no actuation [14, 32], in a gait similar to the compass walker. These rely on gravity to inject the required energy into the system to overcome the losses that occur from impact during the gait-switching phase. With the addition of a small degree of actuation to inject this energy instead, highly efficient, controlled walking has been possible, as well [33]. The passive dynamics approach has even been demonstrated to have some degree of disturbance rejection capability [34, 35].

This illustration that walking gaits, if designed properly, can be stabilized with little-to-no corrective control is important, as it highlights the inherent robustness of the walking motion. In the face of larger disturbances, however, more control authority for stabilization is necessary. Additionally, for the robots to be useful in daily life, they must be able achieve more than simply walk passively over flat surfaces carrying minimal weight. However, by utilizing this knowledge and designing controllers around the underlying passive dynamics, the required control authority can be greatly reduced, improving the effectiveness of the

overall control strategy.

Instantaneous Capture Point and Divergent Component of Motion

The dynamics of the linear inverted pendulum can be shown to be separable into first order stable and unstable parts, with the unstable one being defined as the Instantaneous Capture Point (ICP) [17]. This was simultaneously and independently defined as Divergent Component of Motion (DCM) [18] and the Extrapolated Center of Mass [36]. The DCM was then extended to its three-dimensional equivalent in [37] to enable planning and tracking in three dimensions. Of interest in this work is definition of the 3D DCM (henceforth just referred to as the DCM) as a general state transformation, only related to the linear inverted pendulum through the evaluation of the dynamics' time constant. By allowing this time constant to vary, the time-varying DCM was introduced in [38] as an attempt to more effectively control the robot's momentum over varying height.

The ICP and DCM are effectively the point at which, if held constant, the CoM will converge to. Feedback control of only this unstable component of the CoM dynamics has been shown to be an effective method for stabilizing walking motions [37, 38, 39, 40, 41, 42, 43, 44]. By controlling where the CoM is going to go, the CoM is indirectly controlled, as well, while letting the stable CoM dynamics evolve naturally. The work in [37, 38, 39, 42, 43, 44] focuses on designing and tracking desired ICP and DCM trajectories. A major difference between the ICP trajectories designed in [37, 42] and [38] is that, in [38], the DCM trajectories are derived from a desired CoP trajectory, guaranteeing dynamic feasibility, while [37, 42] achieved dynamic feasibility through careful design of their heel-toe DCM trajectories. The work in [40, 45] utilized the ICP dynamics to control the upcoming footstep location and stance foot contact forces during swing. [46] proposed an alternative to traditional feedback controllers in the form of a MPC algorithm based on the ICP dynamics that utilizes CoP feedback to achieve stable walking motions.

The use of controlling the DCM and ICP was so promising, it was utilized in the DARPA Robotics Challenge by several separate teams. Based on the work in [38], we utilized the DCM as described in [44, 47] to traverse flat, compliant, and uneven terrain, as required by the DRC tasks. IHMC utilized the ICP, as described in [48], based on work proposed in [42], and were able to achieve second place by doing so. Both of these teams illustrated the effectiveness of utilizing ICP/DCM strategies in combination with whole-body compliant control algorithms.

1.2.3 Compliant Torque-Control

The need for robust control strategies for bipedal walking has lead to the development of new, compliant control approaches. This methodology focuses on controlling the interaction forces with the environment to achieve higher level goals, such as desired momentum rates, task-

space velocities or accelerations, joint-space velocities or accelerations, or general behaviors such as staying upright. By focusing on these interaction forces, the control algorithm no longer has to rely on high-impedance actuation to achieve joint positions that track kinematic trajectories. Low joint stiffness can help to minimize the damage of hardware due to impact, and offer safer interactions with humans. Using low-impedance force control, robots can exhibit naturally compliant behaviors capable of adapting to uncertainties found in real-world environments [49]. Below, we outline some of the compliant control strategies proposed for force-controllable robots.

One of the first torque-controlled strategies developed for walking robots was virtual model control [50]. This approach fixes virtual actuators to the links of the robot to impose virtual forces to achieve desired motions. These forces are realized via the Jacobian transpose approach, derived from the principal of virtual work, to compute the desired joint torques. These virtual elements can be used to achieve objectives to make the leg swing, the robot maintain a certain height, or move forward at a certain speed [51]. Virtual model control has been successfully demonstrated on a variety of platforms, including planar walking on Spring Turkey [50] and Spring Flamingo [51] out of the MIT Leg Lab, and M2V2 out of IHMC [40], which used ICP control methods to achieve 3D walking. It has also been augmented with an adaptation mechanism to compensate for external disturbances and unmodeled dynamics [52, 53]. Virtual model control was also combined with inverse-dynamics in [54] to achieve balance for simulated characters, where the inverse-dynamics were used to allow low-gain tracking for compliance.

A separate strategy was developed by Hyon et al., who used passivity based control on the contact forces to maintain balance [55]. This strategy was then augmented in [56], which introduced CoM control by regulating the impedance of joints during fast moves with a dynamic balancer. These strategies were shown to be capable of maintaining balance during both standing and stepping motions.

While virtual model control has the benefit of not requiring a dynamic model, model based inverse-dynamics approaches have shown extreme promise. Often referred to now as whole-body or full-body control, inverse-dynamics based control is capable of fulfilling task-space and joint-space objectives, and even resolving multiple, often redundant motion objectives. Often credited as introducing the first whole-body controller, Sentis et al. [57] extended manipulator prioritized task-space control methods to robots with a free-floating base. This approach projects a series of tasks into the operational space of the constrained rigid-body dynamics [57, 58, 59, 60]. Lasa et al. [61] redefined this problem as a series of prioritized quadratic objectives, where subsequent goals are solved inside the previous solution null-space. The problem was simplified by removing the contact forces entirely through orthogonal decomposition [62]. This, however, did not reflect the unilateral, friction limited nature of the contact.

This prioritized approach has the distinct advantage of requiring little tuning, beyond setting the priority level of the different tasks, as it works entirely within the null-space of previous

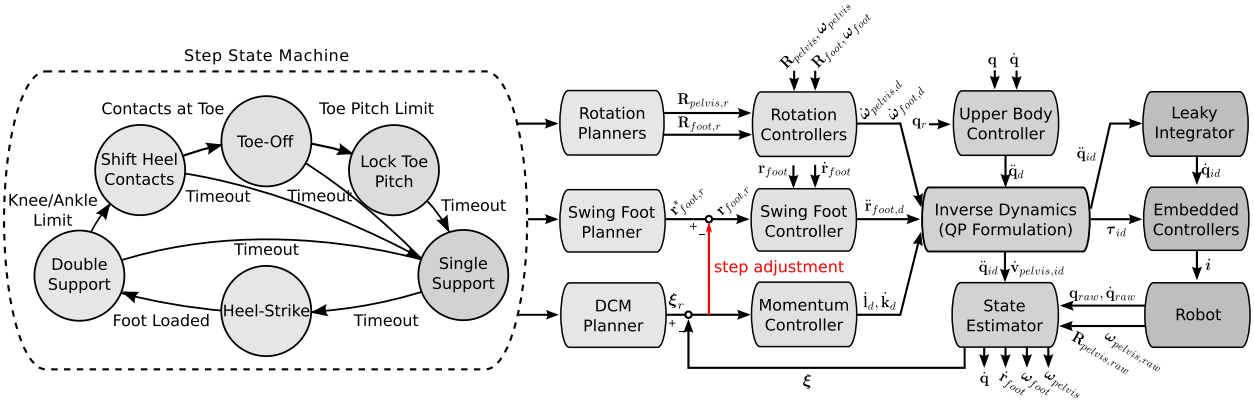


Figure 1.3: Control architecture for the whole-body control scheme developed in [47].

solutions. It does, however, have the disadvantage of not having the inherent ability to resolve task redundancy, with redundant tasks being ignored. Based on different desired behaviors and problem conditions, there may be scenarios where some trade-off between two motion objectives is wanted. As an example, when recovering from an external disturbance, it may be desirable to pitch the torso and windmill the arms to help maintain balance. Using null-space projection, achieving this combination of motions may be extremely difficult, if not impossible. However, by describing the problem in a convex optimization form, this becomes possible.

Stephens et al. first introduced whole-body control using convex optimization [63]. They solved the problem in two steps, where first they optimized the contact forces for balance, and then minimized the joint torques and accelerations subject to the rigid-body dynamics. This optimization framework has subsequently been utilized by a variety of groups [64, 65, 66, 67, 68, 69], eventually becoming a constrained quadratic program [66, 67, 68, 69]. This formulation enables the true, unilateral, friction limited nature of the contact forces to be captured, by conservatively describing them as a linear constraint [68]. Additional constraints and objectives can also be imposed, such as minimizing joint accelerations or actuator saturation limits, without the concerns of over-constraining the problem, as may happen in strict prioritization methods.

The capabilities of optimization-based whole-body control were probably best demonstrated in the 2015 DARPA Robotics Challenge. Several teams separately developed whole-body controllers for their robots, capable of reliable locomotion over uncertain terrain, uneven surfaces, and stairs. As in other works, these approaches computed admissible joint torques and accelerations based on the rigid body dynamics. Using the Atlas humanoid, supplied by Boston Dynamics, Feng et al. developed a whole-body controller that included the desired CoP as an objective [31, 70, 71]. They additionally solved a separate optimization to determine desired joint velocities and accelerations. Also on Atlas, Kuindersma et al. utilized

a closed-form time-varying LQR solution for tracking the ZMP for stable walking [30, 72]. Notably, they also utilized their whole-body controller to stabilize full-body plans computed offline [30]. Johnson et al. and Koolen et al. tracked desired ICP trajectories using momentum feedback in a whole-body controller [48, 73]. Using Atlas as well, they managed to place second in the competition with this compliant control approach.

For the DRC, we also developed a whole-body controller for our custom humanoid, ESCHER [74]. Similar to the work in Johnson et al. and Koolen et al. [48, 73], we used momentum control methods to track desired DCM trajectories [43, 44, 47]. The control architecture utilized in [47], similar to that used in [48] and [70], is shown in Figure 1.3. This cascaded design relies on desired behaviors determined by the high-level state machine, which correspond to a series of objectives, such as momentum rate of change goals and spatial acceleration tasks. Each of these tasks then utilizes a feedback controller to determine the setpoints for the whole-body controller. The whole-body controller then resolves the motion objectives into torque, acceleration, and contact force setpoints for the robot.

1.2.4 Compliant Bipeds and Quadrupeds

While compliant control strategies require force-controllable robots, the reasons for designing robots with compliance go beyond simply enabling force control. Compliance in nature has been observed to provide a number of benefits, including improved metabolic efficiency, energy storage, and passive stability [75]. Compliance has be utilized by researchers to improve efficiency, both on exoskeletons [76] and robots [77, 78], and is most often implemented as series elasticity to enable low-impedance, force-controllable actuators. The remainder of this section will provide an overview of different force-controllable bipeds and quadrupeds.

Electric Robots

To achieve compliant force control with electric motors, Pratt and Williamson developed the series elastic actuator (SEA) [79]. This design includes a spring element in series with the actuator transmission, essentially reducing mechanical impedance. This actuator configuration is capable of precise, stable, closed-loop force control while increasing the shock tolerance of the actuator by effectively including a mechanical low-pass filter in the dynamics [79, 80, 81, 82]. Many electric robots utilize some form of SEAs to achieve their compliant control.

Linear SEAs Building off their development of the first SEAs, the MIT Leg Lab developed several platforms using them. The M2 biped [83] utilized linear ball screw driven SEAs with inline die springs at each of the 12-DoF in the lower body. The Yobotics IHMC M2V2 biped, based on the design of M2 [84], used similar ball-screw driven SEAs, and was demonstrated to be capable of compliant walking and push recovery using ICP feedback and virtual model

(a) (b) (c)

Figure 1.4: (a) ESCHER humanoid, by Virginia Tech. (b) Valkyrie humanoid, by NASA. (c) WalkMAN humanoid, by IIT.

control [40]. Planar SEA driven robots with point feet, such as UT Austin's HUME, have also been shown as capable of achieving stable walking motions [85], and are making strides towards 3D walking [86]. The Valkyrie humanoid, developed by NASA as their entry into the DRC, uses SEAs in both the arms and legs [87]. Valkyrie feature a unique combination of linear and rotary SEAs to achieve a very modular design. Notably, it has perhaps the largest sensor package of any current humanoid robot.

The SAFFiR prototype, a 12-DoF biped utilizing linear SEAs at the ankles, achieved 3D walking over a variety of compliant terrain [88]. Similar to M2V2, THOR uses linear SEAs in all 12-DoF of the lower body, while maintaining the parallel actuation strategy of the hips and ankles developed on SAFFiR [89]. THOR features a unique Hoeken's linkage design at the hip and knee pitch to achieve a uniform mechanical advantage throughout the range of motion [90]. Utilizing DCM tracking feedback and whole-body control, THOR became the first humanoid robot to extinguish fires as part of SAFFiR program [91]. Designed for the DRC, ESCHER is an iteration on THOR, featuring the same actuator package [74]. Notably, ESCHER also utilizes dual actuators that function in parallel at each knee to provide higher joint torques. As with THOR, ESCHER also utilized DCM tracking feedback and whole-body control to compete in the DRC [49]. The SEA design used on both THOR and ESCHER

features a unique compliant mechanism consisting of a titanium beam loaded in bending, as well as an inline load cell for direct force measurement.

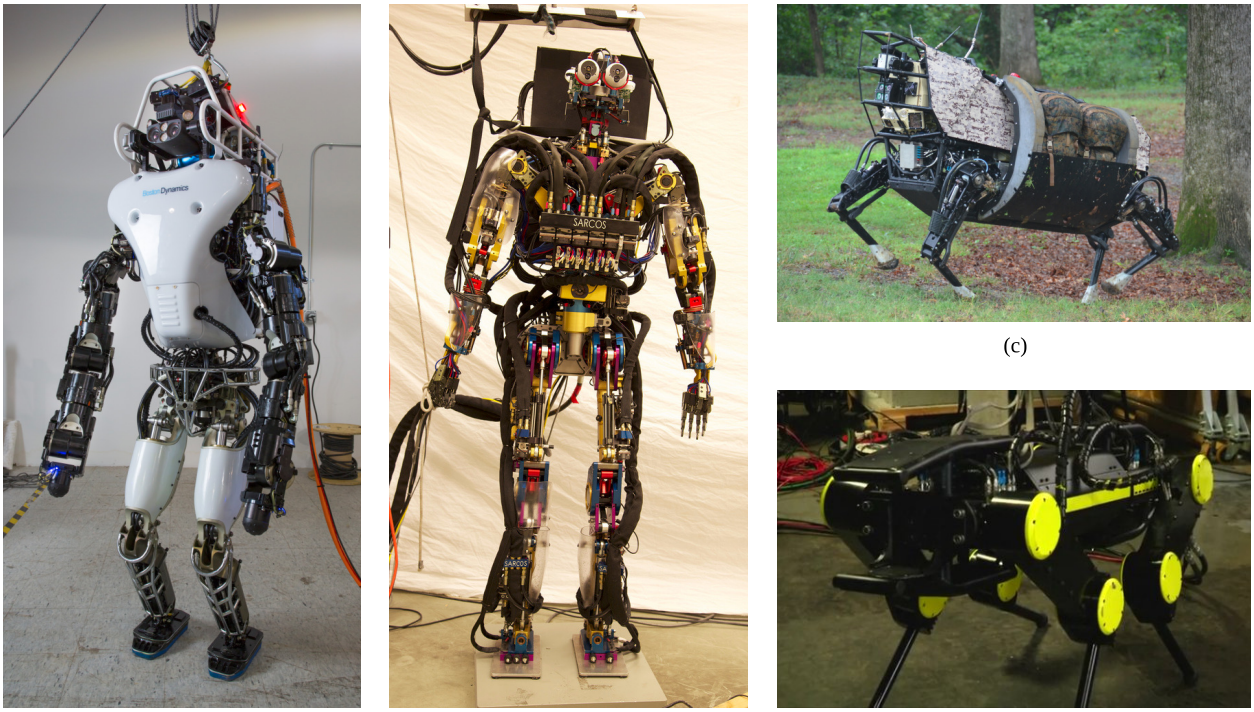
Rotary SEAs Rotary SEAs are an alternative design that often consist of an electric motor and gearbox coupled with a rotary spring. COMAN from IIT uses rotary SEAs in the hip, knee, and ankle pitch to achieve compliant balancing and walking [92]. The Valkyrie humanoid also includes rotary SEAs in the hips, knees, and arms [87]. The Atrias biped uses two rotary SEAs to drive a parallelogram shaped four bar linkage, and is capable of compliant walking and running [93]. WalkMAN is a fully articulated humanoid designed by IIT for disaster response [94, 95]. They designed the lower-body to have human-like weight distribution by careful placement of the leg actuators through the use of four-bar linkages. The motors themselves utilize a harmonic-drive gear reduction with built in compliance and direct torque measurement. Quadrupeds StarlETH [96] and ANYmal [97] were developed by ETH Zurich, and utilize compliant rotary SEAs that enable walking over a variety of terrain using virtual model control. They are also exploring the use of full inverse dynamics control in walking.

Cable Driven SEAs The first robots to utilize series elasticity, MIT's Spring Turkey and Spring Flamingo, did so via cable drives to allow relocation of the motor and transmission away from the joint [50, 51]. Both robots utilize ball-screw driven SEAs and cable drives to reduce the inertias in the legs by placing the actuators in the body of the robots. TU Delft's Flame and Tulip humanoids were designed to compete in RoboCup [98], and used cable drives with inline tension springs for elasticity.

Force-Controllable Motors While the inclusion of elasticity enables higher precision torque control, it is not always required to achieve true force-controllable actuation. TORO from DLR uses rotary harmonic drive gearboxes with torque sensors in series with the output link [99]. TORO has been used to successfully demonstrate a variety of multi-contact balancing and control strategies, as well as much of the development of DCM-based planning and control strategies. IIT's iCub [100] is a child sized, open-sourced articulated humanoid package. It also utilizes harmonic drive gear reductions and direct torque measurement to achieve torque control at each joint, making it capable of whole-body control with dynamic balance and walking.

Hydraulic Robots

As an alternative to electric actuation, robots can utilize hydraulic actuators for torque-controllable joints. Closed-loop force control can be implemented on hydraulic actuators by using hydraulic pressure sensors or inline load cells. One of the first such platforms, the SARCOS Primus biped utilizes hydraulic actuation to achieve torque control, and has



(a) (b) (c) (d)

Figure 1.5: (a) Atlas humanoid, by Boston Dynamics. (b) SARCOS Primus humanoid. (c) Alpha dog, by Boston Dynamics. (d) HyQ2Max, by IIT.

been successfully utilized to demonstrate dynamic balancing [63]. Possibly the most famous hydraulic humanoid, Atlas, was designed by Boston Dynamics to provide a platform for several teams in the DARPA Robotics Challenge. It has been successfully used to demonstrate dynamic balancing and walking with whole-body control [30, 70, 73]. The IIT HyQ [101] and HyQ2Max [102] quadrupeds utilize hydraulic actuation for compliant walking, and are capable of dynamic walking and jumping maneuvers.

1.3 Current Limitations

While bipedal walking performance has made significant strides forward, considerable progress is still needed for robots to rival the efficiency, mobility, and reliability of humans. A healthy human can reliably walk over an extreme variety of terrains, from stone fields to ice, with rarely falling. As illustrated in the 2015 DARPA Robotics Challenge, however, robots are not nearly this capable yet, taking nearly an hour to complete a set of tasks that would take a human less than ten minutes. Indeed, all but one team that used a walking robot fell at least once throughout the competition. This was all achieved with the robot using extremely

bent knees, with little regard given to the overall robot efficiency, or how "bio-inspired" the gait is.

Exoskeletons, as well, are still far from enabling those with spinal cord injuries the ability to walk as adeptly as an able-bodied individual. Exoskeleton capabilities were thoroughly demonstrated in the 2016 Cybathlon, where pilots took upwards of ten minutes to complete a course designed to model every day environments that able-bodied individuals completed in one. Not only that, but most pilots at the time of completion were left completely exhausted. The design and control of the exoskeleton walking motions remain very simplistic, with little-to-no feedback. Until the gait design from humanoid robots becomes more natural, though, there can be little cross-over to exoskeletons, as that.

To be more usable, bipedal robot walking and balance must become both more natural and more robust. Humans can recover from large disturbances, quickly and reliably determining where and when to step to best avoid a fall. The requirements facing robot walking illustrate a common theme: situations that people face in day to day life. From bumping into doors to stepping on an unseen toy when walking to the bathroom in the dark, people are able to cope with external disturbances and uncertainties exceptionally well. To recover from large disturbances, simple feedback control using either an instantaneous feedback control law or MPC controllers are no longer sufficient, instead requiring step adjustment. Most step adjustment feedback methods, however, involve some sort of heuristic [37, 44], rather than dynamics-based solution, and are not extendable to multi-contact problems.

To achieve natural, straight-legged walking, the only current known approaches either involve advanced CoM height planning or whole-body planning techniques, neither of which are conducive to online implementation due to long solve times. Indeed, many of the properties of natural walking, such as toe-off, are often not exhibited well in humanoid robots. Better strategies are needed to achieve these actions online.

1.4 Approach

In this dissertation, we present the development of several essential elements to achieve robust, natural bipedal walking gaits. We discuss two in-depth studies of step adjustment, force feedback, and swing speed-up strategies. We also discuss an approach for swing and transfer timing selection to achieve dynamic plans that require minimal knee bend. Lastly, we present an approach for achieving natural locomotion gaits that involve straight leg motions and toe-off. Experimental and simulation results using the THOR, ESCHER, and Atlas hardware platforms, which demonstrate the feasibility of the presented approach.

As discussed in the previous section, many step adjustment strategies are highly heuristic. Instead, we present several non-heuristic based step adjustment strategies. This is essential to increasing the control authority of the robot, which is significantly reduced when walking with straight legs. Model predictive control (MPC) shows significant promise in this area, as

illustrated in [27]. To explore its usefulness, we extend this work to utilize the time-varying DCM formulation, improving MPC strategies for compliant locomotion. By developing MPC algorithms that can also consider step rotation, they are improved to perform better in cluttered environments, and can also function as a local check on the dynamic feasibility of a desired footstep plan. We also include an analysis of the practicality of using MPC for walking control, as where as simulation results with the torque controlled humanoid ESCHER.

In addition to traditional MPC algorithms, we also explore the use of casting a proportional feedback controller in an optimization framework, capable of adjusting the upcoming footsteps. We believe that this formulation is better able to utilize CoP feedback and angular momentum feedback before adjusting the future footsteps, only doing so when CoP feedback has become saturated by reaching the edge of the support polygon. We utilize two separate dynamics-based approaches for determining the necessary amount of step adjustment, one which is fairly dynamic-plan specific, the other that is independent of the dynamic plan. We also include two separate algorithms for swing-speed up. The proposed approaches enable disturbance rejection while walking, greatly increasing the overall control authority of the robot. We include simulation and experimental results on the hydraulic humanoid, Atlas, and discuss the practical considerations and limitations for implementing such algorithms on real hardware.

To ensure that the desired dynamic plans do not require significant knee-bend to execute, and are in turn relatively natural, we utilize an algorithm that optimizes the swing and transfer durations. Despite the best efforts of a controller or center of mass height planner, if the horizontal dynamics kinematically require the robot to bend its knees to be achieved, straight leg walking is impossible. Step timings have traditionally been considered a tuning variable. Instead, we note their relation to the result kinematics, and adjust them until the kinematic motions are satisfied. We include simulation and experimental results on the hydraulic humanoid, Atlas, showing that, with this algorithm active, the robot is able to achieve smoother, more dynamic walking gaits that require less knee-bend.

Finally, to achieve the desired natural, straight leg walking motions, we present an approach that capitalizes on the whole-body controller framework's ability to compute the necessary joint torques for the robot to achieve higher level motion tasks. We believe that, instead of specifying all the desired motions for the robot, such as toe-off action and desired center of mass height, through correct design we can allow the controller to decide these for itself. This then allows these properties to become emergent, as they are in humans. Toe-off and a high center of mass are both actions to achieve a goal. Instead of specifying the action, we can specify the goal, and allow the controller to decide the action on its own. To do this, we do not directly specify a desired center of mass vertical motion. Instead, we bias the whole-body controller to always use straight legs by projecting a desired straight leg configuration into the null-space of the whole-body controller quadratic program. Similarly, we do not specify a desire toe-off action, leaving the pitch of the trailing foot unconstrained. This approach is implemented in the same software framework as the previous algorithms,

and relies on their improvements for success. It is demonstrated on the Atlas humanoid in both hardware and simulation experiments.

1.5 Contributions

The specific proposed contributions of this dissertation are as follows:

1. Development of MPC algorithm that utilizes time-varying DCM for improved control over varying-height terrain.
2. Inclusion of step positions in MPC algorithm for model-based step adjustment for disturbance recovery.
3. Inclusion of step rotation in MPC algorithm for navigation of cluttered environments and local checks on dynamic feasibility of footstep plans.
4. Strategies to mitigate the long solve time of MPC based controllers.
5. Derivation of a planner-derived optimization proportional feedback controller capable of adjusting upcoming footsteps and using angular momentum for disturbance recovery.
6. Derivation of an error-based optimization proportional feedback controller capable of adjusting upcoming footsteps and using angular momentum for disturbance recovery.
7. Development of dynamics based swing-time adjustment strategy for improved push recovery when stepping.
8. Development of optimization based swing-time adjustment strategy for improved push recovery when stepping.
9. Implementation of null-space manipulation for straight leg walking strategies on humanoid robots.
10. Improvement of toe-off strategies for whole-body controllers for bipedal locomotion.
11. Design of ICP-based timing selection of walking trajectories for straight-leg walking.
12. Strategies for improved robustness to height uncertainties when walking.

1.6 Outline

This dissertation is organized in a manuscript format. Chapters 2, 3, 4, and 5 serve as independent manuscripts that describe the necessary elements to achieve natural, efficient bipedal walking in a whole-body control framework. These algorithms are implemented on a variety of platforms, including THOR and ESCHER from Virginia Tech and Atlas, made by Boston Dynamics.

Chapter 2 presents the development of a full model-predictive controller for trajectory optimization using the divergent component of motion to achieve step adjustment, force feedback, and step rotation optimization. Simulation results using the Gazebo simulation environment with the ESCHER humanoid are included.

Chapter 3 builds off what was learned in Chapter 2, and instead highlights two separate, optimization based feedback control laws for achieving force-feedback, angular-momentum feedback, and step adjustment, as well as two approaches for incorporating a third balance mechanism - swing speed up. Simulation and experimental results using the Atlas humanoid are included, using the IHMC Simulation Construction Set and the physical hardware.

Chapter 4 presents an approach for step timing selection to generate dynamic plans for humanoid locomotion that result in walking with minimally-bent knees. We again present results using the IHMC Simulation Construction Set, as well as the Atlas humanoid robot from Boston Dynamics.

Chapter 5 outlines the approaches used to achieve straight leg walking on the Atlas humanoid robot. This includes the strategy for indirect control of the center of mass height for the straight leg walking itself, as well control techniques for achieving a reliable toe-off motion and improved swing foot control for both lift-off and touchdown. Both simulation and experimental results are presented using the Atlas humanoid robot.

Chapter 6 concludes this manuscript with example applications bipedal robots are useful for that the authors were involved in over the course of this research, including the 2014 Office of Naval Research SAFFiR program, the 2015 DARPA Robotics Challenge, and the 2016 Cybathlon. We also include a discussion of future research directions.

1.7 Attribution

Chapter 2 is a journal paper submitted to the International Journal of Humanoid Robots. It is an expansion off of work published in the 2016 IEEE International Conference on Robotics and Automation (ICRA) [103]. Michael Hopkins provided much of the initial guidance for the algorithmic development. Alexander Leonessa then provided much needed feedback and insights for the preparation of the full manuscript.

Chapter 3 contains material published in the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [104]. This chapter, however, builds considerably off of this work, and is in preparation as a separate journal manuscript. This paper includes four co-authors. Sylvain Bertrand provided considerable insight into the understanding of the dynamic planning algorithm presented in [37], as well as with experimental testing. Georg Wiedebach was invaluable for assisting with testing on the robot, as well as with for brainstorming the problem formulation and the gradient descent approach. Alexander Leonessa and Jerry Pratt contributed valuable insight and feedback for the early drafts of the manuscript.

Chapter 4 contains material submitted to the 2017 IEEE-RAS International Conference on Humanoid Robots (Humanoids) [105]. This paper includes four co-authors. Georg Wiedebach provided advice on the gradient descent approach. Sylvain Bertrand assisted in understanding the scope of the total problem. Alexander Leonessa also greatly helped designing the iterative feedback approach for the gradient descent algorithm.
Jerry Pratt contributed valuable insight and feedback for the early drafts of the manuscript.

Chapter 5 contains material submitted to the 2018 IEEE International Conference on Robotics and Automation (ICRA). This paper includes four co-authors. Sylvain Bertrand and Jerry Pratt assisted greatly on the design of when the robot should use straight legs, and how to cast the problem in the correct framework. Georg Wiedebach provided a good deal of brainstorming input, helping to trouble-shoot many problems related to toe-off. Alexander Leonessa contributed valuable insight and feedback for the early drafts of the manuscript.

1.8 Contributed Works

As part of this research, the following works have been written, and either presented at conferences, published in conferences or journals, or submitted for publication and are currently under the review process. Note that some of them compose the foundation of this thesis, while others were published independently of this exact research thrust.

- Michael Hopkins, Robert Griffin, and Alexander Leonessa, "Compliant Locomotion: A Model-Based Approach". In: *ASME Dynamics Systems and Control Magazine*. June 2015.
- Michael Hopkins, Robert Griffin, Alexander Leonessa, Brian Lattimer, and Tomonari Furukawa, "Design of a Compliant Bipedal Walking Controller for the DARPA Robotics Challenge". In: *15th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. September 2015.
- Robert Griffin and Alexander Leonessa, "Model Predictive Control for Dynamic Footstep Adjustment Using the Divergent Component of Motion". In: *IEEE International Conference on Robotics and Automation (ICRA)*. May 2016.

- Robert Griffin, Alan Asbeck, and Alexander Leonessa, "Disturbance Compensation and Step Optimization for Push Recovery". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016.

- Brian Lattimer, Joseph Starr, Josh McNeil, Chris Nogales, John Peterson, Jason Ziglar, James Burton, Coleman Knabe, Yoonchang Sung, John Seminatore, Robert Griffin, Dennis Hong, Daniel Lee, Jack Newton, Viktor Orekhov, Michael Rouleau, Michael Hopkins, and John Farley, "Humanoid Firefighting Robot for Structure Fires". In: *14th International Conference and Exhibition on Fire Science and Engineering (Interflam)*. 2016.

- Georg Wiedebach, Sylvain Bertrand, Tingfan Wu, Luca Fiorio, Stephen McCrory, Robert Griffin, Francesco Nori, and Jerry Pratt, "Walking on Partial Footholds Including Line Contacts with the Humanoid Robot Atlas". In: *16th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. September 2016.

- Coleman Knabe, Robert Griffin, James Burton, Graham Cantor-Cooke, Lakshitha Dantanarayana, Graham Day, Oliver Ebeling-Koning, Eric Hahn, Michael Hopkins, Jordan Neal, Jack Newton, Chris Nogales, Viktor Orekhov, John Peterson, Michael Hopkins, John Seminatore, Yoonchang Sung, Jacob Webb, Nikolaus Wittenstein, Jason Ziglar, Alexander Leonessa, Brian Lattimer, and Tomonari Furukawa, "Team VALOR's ESCHER: A Novel Electromechanical Biped for the DARPA Robotics Challenge". In: *Journal of Field Robotics*. February 2017.

- Robert Griffin, Georg Wiedebach, Sylvain Bertrand, Alexander Leonessa, and Jerry Pratt, "Walking Stabilization Using Step Timing and Location Adjustment on the Human Robot, Atlas". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. September 2017. In Print.

- Robert Griffin, Tyson Cobb, Travis Craig, Mark Daniel, Nick van Dijk, Jeremy Gines, Koen Kramer, Shriya Shah, Olger Siebinga, Jesper Smith, and Peter Neuhaus, "Design and Approach of Team IHMC in the 2016 Cybathlon". In: *IEEE Robotics and Automation Magazine*. 2017. In Print.

- Robert Griffin, Georg Wiedebach, Alexander Leonessa, and Jerry Pratt, "Angular Momentum, Step Timing and Location Feedback Through Online Optimization". In: *Robotics Science and Systems (RSS). Workshop on Challenges in Dynamic Legged Locomotion*. July 2017.

- Robert Griffin, Sylvain Bertrand, Georg Wiedebach, Alexander Leonessa, and Jerry Pratt, "Capture Point Trajectories for Reduced Knee Bend Using Step Time Optimization". In: *17th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2017. In Review.

- Cameron Ridegwell, Robert J. Griffin, Tomonari Furukawa, and Brian Lattimer, "Online Estimation of Friction Constraints for Multi-contact Whole Body Control". In: *17th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2017. In Review.

- Robert J. Griffin, Georg Wiedebach, Sylvain Bertrand, Alexander Leonessa, and Jerry Pratt, "Straight-Leg Walking Through Underconstrained Whole-Body Control". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018. In Review.

- Robert Griffin and Alexander Leonessa, "Model Predictive Control for Stable Walking Using the Divergent Component of Motion with Footstep Location and Yaw Adaptation". In: *International Journal of Humanoid Robots*. 2017. In Progress.

Chapter 2

Model Predictive Control For Footstep Location and Rotation Adaptation Using the Divergent Component of Motion

This paper presents an extension of previous model predictive control (MPC) schemes to the stabilization of the time-varying divergent component of motion (DCM). In order to address the control authority limitations caused by fixed footholds, the step positions and rotations are treated as control inputs, allowing the generation and execution of stable walking motions, both at high speeds and in the face of disturbances. Rotation approximations are handled by formulating a quadratically constrained mixed-integer quadratic program, which, when combined with the use of the time-varying DCM to account for the effects of height changes, improves the versatility of MPC. Simulation results of fast walking and step recovery with the ESCHER humanoid demonstrate the effectiveness of this approach.

2.1 Introduction

While bipedal walking performance has made significant strides forward, considerable progress is still needed for bipeds to rival the mobility of humans. A healthy human can reliably walk over an extreme variety of terrains, from rocky surfaces to ice, with rarely ever falling. Humans can even recover from large disturbances, quickly and reliably determining where and when to step to best avoid a fall. These example illustrate a common theme: situations that people face in day to day life. From bumping into doors to stepping on an unseen toy when walking to the bathroom in the dark, people are able to deal with external disturbances and uncertainties exceptionally well. While humanoids have illustrated the ability to walk

extremely reliably in structured, controlled environments, they must able to deal with this same variability to provide comparable performance in the real world.

Humanoid robot balance is commonly achieved by either directly or indirectly controlling the center of mass (CoM) dynamics. This approach reduces the high degree-of-freedom, highly nonlinear dynamics of the articulated body to a simple three degree-of-freedom system that still obey Newton's laws. Due to the friction-limited, unilateral contact nature, control of the CoM dynamics is highly dependent on the location of the footsteps. Despite this, the step positions found by many high level footstep planners do not account for the CoM dynamics during computation, and rely on heuristically tuned algorithms to generate feasible footholds [106, 107, 108]. To address this, some planners simply give the robot a desired velocity, and they adjust the foot positions to maintain that velocity [51, 109]. The recent preference towards high-level planners, however, is driven by the desire to achieve precise foot placements through the integration of terrain maps from perception data. This enables the robot to traverse complex terrain, such as debris fields and stairs. As an illustration of this issue, we can examine Figure 2.1, which shows desired DCM trajectories resulting from the footsteps, which guide the CoM motions. If poorly tuned, it is possible (although unlikely) for a high level planner that does not include the CoM dynamics to produce a plan such as in Figure 2.1(a). When properly tuned, a heuristic-based planner could produce as plan such as the one in Figure 2.1(b), which, at first glance, results in satisfactory CoM motions. When accounting for the CoM dynamics, however, the footstep planning can be improved to produce plans like that in Figure 2.1(c), which demonstrates smooth CoM motions while ending at the same target point. Capturing these dynamic considerations, then, is an important aspect of footstep planning.

Once a set of desired footstep is obtained, a common approach for CoM trajectory planning is to find acceptable center of pressure (CoP) trajectories and then use the linear inverted pendulum (LIP) dynamics to compute the desired CoM trajectory [13, 22, 110, 111], noting the CoP equivalence to the zero-moment point in x and y [21]. Pratt et al. [17] split the CoM dynamics into stable and unstable parts by defining the instantaneous capture point (ICP) as the location above which the CoM will stop. This point was simultaneously and independently defined as the divergent component of motion (DCM) [18] and the extrapolated center of mass [36]. The DCM was then extended to its three dimensional equivalent [37] to enable planning and tracking in three-dimensions. Hopkins et al. [38] then proposed the time-varying DCM, pictured in Figure 2.2, by allowing the time-constant of the LIP to vary, in an attempt to improve the control of the CoM height over varying terrain. Feedback control of only the unstable component of the CoM dynamics in the form of the ICP and DCM has been demonstrated as an effective method for stabilizing walking motions [37, 38, 41, 42, 43, 44]. By controlling where the CoM is going to go rather than where it currently is, the CoM is indirectly being controlled as well, while letting the stable CoM dynamics evolve naturally. A major difference between the DCM trajectories designed in Hopkins et al. [38] and Engslberger et al. [42] is that, in Hopkins et al. [38], the DCM trajectories are derived from a desired CoP trajectory, guaranteeing dynamic feasibility, while Engslberger et al. [42]

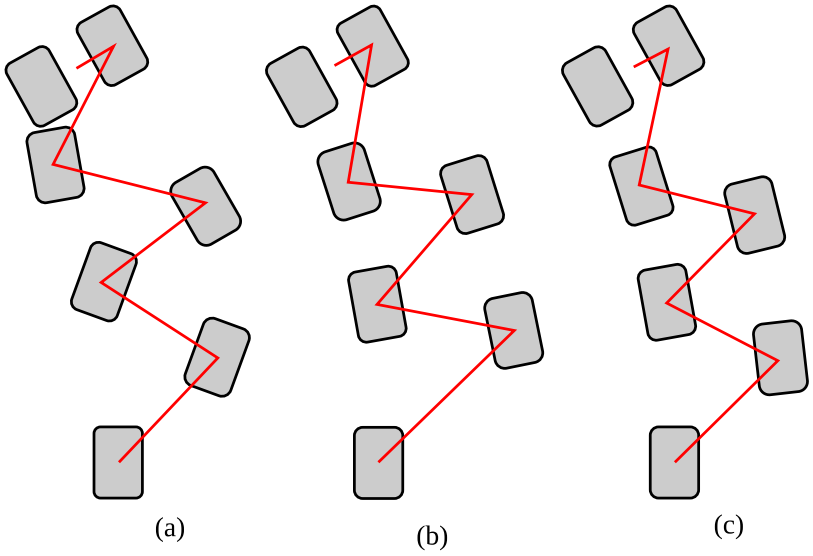


Figure 2.1: Left: Potential output of a poorly tuned footstep planner. Middle: Potential output of a well tuned footstep planner. Right: Adjustment to footstep plan to improve the CoM dynamics. The line represents the DCM trajectory, which guides the CoM motions.

achieved dynamic feasibility through careful design of their heel-toe DCM trajectories.

An alternative to tracking preplanned trajectories with feedback controllers is online model predictive control (MPC) schemes. This typically is done by determining a trajectory that satisfies some condition of optimality from the current state to the desired final state, subject to constraints. For walking, these constraints include the friction-limited, unilateral nature of the ground contacts, which can be described by requiring the CoP to remain inside the support polygon. Originally, this was used to plan feasible CoP trajectories in Kajita et al. [22]. Wieber [25] proposed a full MPC to stabilize the CoM under disturbances. The structure of the MPC was relaxed in Diedam et al. [27] and Herdt et al. [109] by treating the step positions as control inputs used to define the desired CoM trajectory, allowing successful rejection of disturbances using step adjustment. Recently, this formulation was expanded further to include CoM height trajectories using robust MPC techniques [112], allowing the typically nonlinear problem to be treated as a linear one. MPC has also been extended to the unstable ICP dynamics to determine desired CoP points to track using a ZMP position controller [46]. A full MPC in the form of a time-varying linear quadratic regulator (TV-LQR) of the ZMP dynamics was formulated by Kuindersma et al. [30] and included the whole-body controller quadratic program. Tedrake et al. [29] further examined this formulation, by developing an efficient, closed-form, convex solution for the desired CoM motion. The authors exploited the nature of the ZMP dynamics to develop an efficient solution to the Riccati equation, an otherwise more challenging problem. MPC avoids solving the Riccati equation, opting instead to solve the problem using relatively simple matrix manipulation.

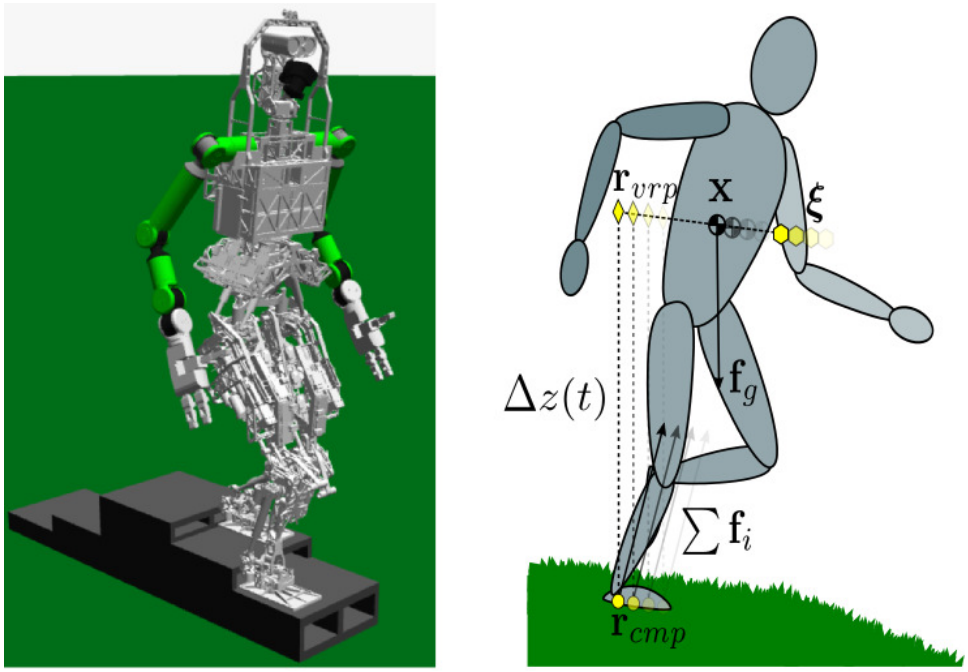


Figure 2.2: Left: ESCHER walking on uneven terrain in simulation, using the proposed MPC algorithm. Right: DCM dynamics and external forces acting on an articulated humanoid. The DCM, $\boldsymbol{\xi}$, is represented by the yellow hexagon, the VRP by the yellow diamond, and the CMP by the yellow circle. $\mathbf{x}$ represents the CoM location, while $\mathbf{f}_g$ is the force of gravity acting on the CoM. $\sum \mathbf{f}_i$ in this case is the sum of all the ground contact forces.

To recover from large disturbances, simple feedback control using either an instantaneous feedback control law or MPC controllers are no longer sufficient. To remain dynamically realizable, the CoP has limits, as it must remain in the support polygon. Generation of angular momentum can be used for stabilization, but the magnitude of angular momentum has limits [113]. Additionally, the angular momentum that is generated during recovery must then be removed from the system to stay within actuator and joint limits. While the variation of height can be used to maintain balance [114], the remaining primary strategy is to adjust the desired footsteps. This effectively expands the base of support, allowing the CoP to move further and stabilize the CoM dynamics. This is often the strategy employed by humans. When pushed, we quickly take a step to the appropriate location. This concept can be described by capturability, where stabilizing with one step is "1-Step Capturable", and if N-steps is required, this is "N-Step Capturable" [45]. As illustrated in Figure 2.3(b), when there is a slight tracking error, the DCM dynamics can be stabilized with only feedback and no step adjustment. If there is large enough error, a step adjustment is required, as shown in Figure 2.3(c). As noted by Feng et al. [115], using step adjustment for dynamic walking can be much more robust than tightly controlling the CoP for balance, as the robot has much higher control authority, and only has to step far enough to catch itself, rather than place the

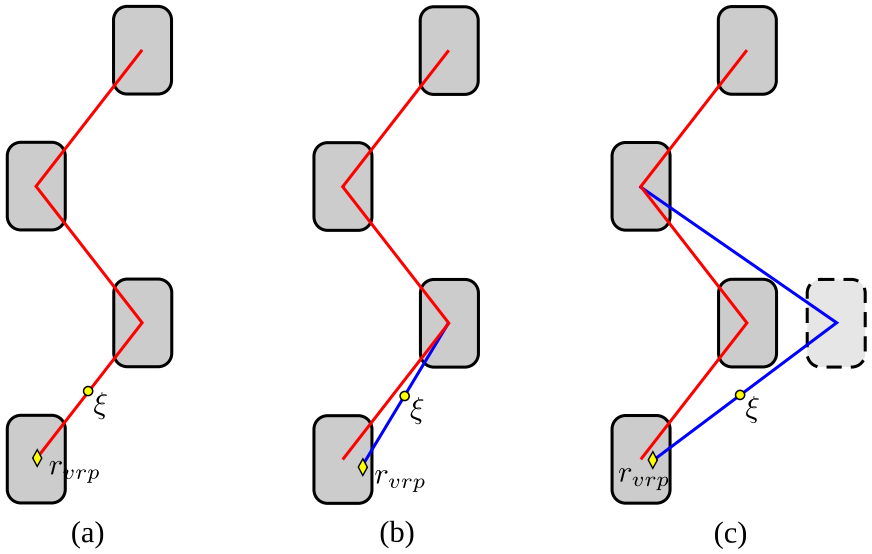


Figure 2.3: The DCM, $\boldsymbol{\xi}$, is represented by yellow circle, while the VRP is the yellow diamond. Left: Standard footstep plan with good DCM tracking. Middle: Standard footstep plan with mild DCM tracking error. The DCM in this problem can be stablized without requiring footstep adjustment. Right: Standard footstep plan with large DCM tracking error. In order for the DCM to be stablized, the footstep plan must be modified.

CoP at the exactly precise point. Most step adjustment feedback methods, however, involve some sort of heuristic [37, 44], rather than dynamics-based solution. Exceptions to these include [45], which instantaneously adjust the upcoming support polygon to keep the robot 0-Step capturable. Some MPC schemes have encoded this footstep adjustment feedback into the MPC problem, by using the footstep position to define the CoM dynamics [27, 109]. Feng et al. [115] presented a convex optimization technique to predict the necessary footstep based on the current CoM state for capturability. Their strategy, however, does not include CoP control, effectively having their robot walk with point feet.

While MPC schemes have been quite successful, they often do not have the same versatility as traditional feedback control methods. The effects of changes in CoM height over uneven terrain are typically not considered in CoP based methods, with several notable exceptions [30, 116, 117]. To account for these effects, we propose a novel formulation of the MPC problem using the time-varying formulation of the DCM dynamics, building off our work in Griffin et al. [103]. This formulation treats the step positions as control inputs, and, uniquely, considering rotation as control inputs. By including rotation, this formulation also enables modifying plans from a high level planner to consider the CoM dynamics. Rotation is a crucial component, as it determines the achievable steps, ultimately determining the stepping path the robot can take. As both height and heading changes are essential to navigating environments outside the lab, these shortcomings limit the current use of real-time MPC methods in fielded robotics. By allowing the step positions to vary, the MPC

algorithm can also adjust the upcoming steps to recover from large disturbances that cannot be rejected otherwise. Currently, this is not addressed for algorithms based on the ICP and DCM dynamics. This paper extends our previous work [103] by improving the formulation and offering more simulation verifications of the proposed method. Specific contributions of this paper include

1. More thorough analysis and presentation of the MPC formulation.
2. Modification to only update the MPC at finite intervals, and using a feedback control between.
3. Addition of a footstep solution regularization cost term to the objective.
4. Incorporation of a DCM forward time prediction method to account for the changing dynamics during the MPC solve time.
5. Thorough illustration of the effectiveness of the current MPC algorithm, including a variety of new experiments.
6. Detailed analysis of the proposed MPC algorithm.

This paper is organized as follows: section 2.2 reviews CoM control methodologies including the LIP model and the DCM definition. section 2.3 reviews the trajectory planning and execution presented in Hopkins et al. [38] for the time-varying DCM. The proposed MPC formulation is presented in section 2.4, proposing methods for the treatment of step positions and rotations as control inputs, as well considerations for the design of the algorithm itself. section 2.6.2 presents the proposed DCM forward time prediction method. section 4.5 presents planning results of the algorithm, as well as simulation results of the 38-DoF model of the ESCHER humanoid [74], demonstrating the capabilities of the proposed MPC scheme.

2.2 Control of Center of Mass Dynamics

The problem of bipedal balancing can be analyzed using center of mass (CoM) models and their relation to ground reaction forces. The following gives a brief background of these models, from the classical linear inverted pendulum (LIP) model to the more recent divergent component of motion (DCM). These dynamics are fundamental to the proposed MPC algorithm.

2.2.1 Linear Inverted Pendulum Model

By focusing on the CoM, the high dimensionality of humanoid dynamics is reduced to much more tractable three degree-of-freedom dynamics. The LIP model has been widely used for

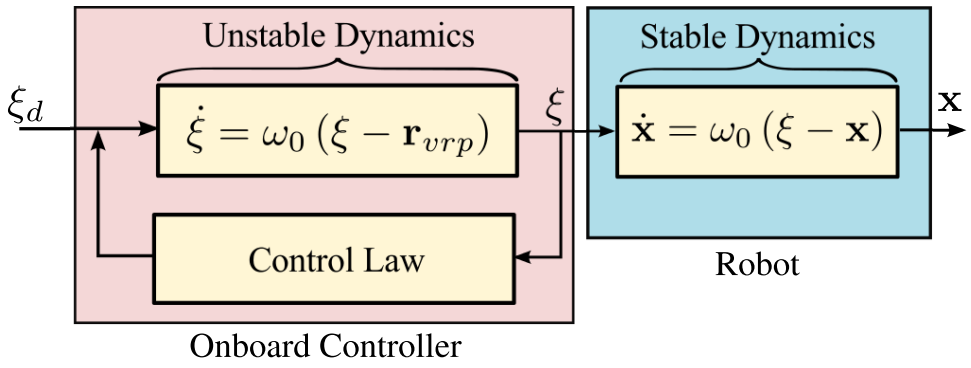


Figure 2.4: Coupling of the unstable and stable CoM dynamics, where ξ represents the DCM position and $\mathbf{x}$ the CoM position.

biped walking [13, 22, 110, 111], as it allows the centroidal dynamics of the robot to be linear and decoupled in x and y. The dynamic equations for the horizontal acceleration of the CoM in the LIP are given by

$$\ddot{x}_{com} = \omega_0^2 (x_{com} - p_x), \quad \ddot{y}_{com} = \omega_0^2 (y_{com} - p_y), \tag{2.1}$$

where $\mathbf{x} = [x_{com}, y_{com}, z_{com}]^T$ is the CoM position, $\omega_0 = \sqrt{\frac{g}{\Delta z_{com}}}$ is the natural frequency of the inverted pendulum, Δz_{com} is the height of the CoM above the ground, and $\mathbf{p} = [p_x, p_y, p_z]^T$ is the location of the torque-free pendulum base joint. This point is equivalent to the CoP, meaning the ground reaction force is collinear with the vector $(\mathbf{x} - \mathbf{p})$. This model does have the assumption that the CoM height above the ground remains constant, which is inherent to the linearization. CoM trajectories can then be found from feasible CoP trajectories and executed using either feedback or MPC strategies on both position and force controlled robots.

2.2.2 DCM Definition

An alternate description of the CoM state is to transform it into stable and unstable first order components, as shown in Figure 2.4. The unstable part is called either the instantaneous capture point (ICP), which can be thought of as the two dimensional point above which the CoM will stop [17], or the DCM, which is the three dimensional point at which the CoM will come to rest [37]. For the rest of this work, we will be referring exclusively to the DCM, without noting the equivalence to the ICP in the x-y plane. The DCM is defined by the following linear transformation:

$$\boldsymbol{\xi} = \mathbf{x} + \frac{1}{\omega_0}\dot{\mathbf{x}}, \tag{2.2}$$

where $\dot{\mathbf{x}}$ is the CoM velocity. The unstable DCM dynamics are then given by

$$\dot{\boldsymbol{\xi}} = \omega_0 \left(\boldsymbol{\xi} - \mathbf{r}_{vrp} \right), \tag{2.3}$$

where $\mathbf{r}_{vrp}$ is the virtual repellent point (VRP). The VRP is an unstable equilibrium point that repels the DCM at a rate inversely proportional to its distance, and is defined by

$$\mathbf{r}_{vrp} = \mathbf{r}_{ecmp} + \begin{bmatrix} 0, & 0, & \Delta z_{com} \end{bmatrix}^T \tag{2.4}$$

where $\mathbf{r}_{ecmp}$ represents the enhanced centroidal momentum pivot, defined by

$$\mathbf{r}_{ecmp} = \mathbf{x} - \frac{\sum \mathbf{f}_c}{m\omega_0^2}, \tag{2.5}$$

which encodes the direction and magnitude of the contact forces, $\sum \mathbf{f}_c \in \mathbb{R}^3$ acting on the CoM.

From Equation 2.2, the stable first-order CoM dynamics are

$$\dot{\mathbf{x}} = \omega_0 \left(\boldsymbol{\xi} - \mathbf{x} \right), \tag{2.6}$$

showing that the CoM converges to the DCM with a time constant $\frac{1}{\omega_0}$. See that these equations hold for all free-floating robot models, with only the definition of the time constant ω_0 coming from the LIP model [42], allowing the proposed work to be applied to other free-floating robot models by replacing ω_0 with some other appropriate time constant.

To add some intuition, the DCM can be thought of as representing what the CoM is going to do, as it combines both the position and velocity. Thus, by controlling the DCM, we are controlling where the CoM will end up. This can be done by moving the VRP, which repels the DCM. By moving the VRP further away from the DCM, we are commanding larger forces, driving the DCM and by result the CoM away more quickly. The DCM-VRP relationship, then, provides a mechanism to determine the desired ground reaction forces to control the CoM.

2.3 Time-Varying Divergent Component of Motion

While time-invariant DCM and ICP approaches have been shown to work well, including over varying height terrains, this is more a testament to the robustness of the control technique and the underlying dynamics than it is to the accuracy of omitting height changes in the controller. To account for this, we can allow both the CoM height and the natural frequency of the inverted pendulum to vary. By including this in the MPC, the algorithm becomes more versatile and usable in a variety of situations, rather than simply flat ground walking.

First introduced by Hopkins et al. [38], the time-varying DCM allows the natural frequency ω_0 to vary with time, resulting in

$$\boldsymbol{\xi} = \mathbf{x} + \frac{1}{\omega(t)}\dot{\mathbf{x}}. \tag{2.7}$$

From here on, we will use the notation $\omega \equiv \omega(t)$ when convenient, omitting the explicit time dependency.

This relaxation in the definition of the DCM allows the height of the CoM to vary with changes in uneven terrain, resulting in improved planning of the vertical CoM trajectory during locomotion. The time-varying version of the DCM dynamics in Equation 2.3 are then

$$\dot{\boldsymbol{\xi}} = \left(\omega - \frac{\dot{\omega}}{\omega}\right)(\boldsymbol{\xi} - \mathbf{r}_{vrp}), \tag{2.8}$$

where $\mathbf{r}_{vrp}$ is now the time-varying VRP, defined in terms of the time-varying eCMP as

$$\mathbf{r}_{vrp} = \mathbf{r}_{ecmp} + \frac{\mathbf{g}}{\omega^2 - \dot{\omega}}, \tag{2.9}$$

where $\mathbf{g} = \begin{bmatrix} 0, & 0, & g \end{bmatrix}^T$. The eCMP can be found by applying Newton's second law to find the CoM acceleration, $\ddot{\mathbf{x}} = \frac{1}{m}\sum \mathbf{f}_c - \mathbf{g}$, leading to

$$\mathbf{r}_{ecmp} = \mathbf{x} - \frac{\sum \mathbf{f}_c}{m\left(\omega^2 - \dot{\omega}\right)}. \tag{2.10}$$

2.3.1 Planning the Height Trajectory

Hopkins et al. [43] presented a framework for planning and executing time-varying DCM trajectories using reverse-time numerical integration and feedback control. The first step of this approach is to define a desired CoM height trajectory.

Given the desired footholds, the CoM height trajectory can be planned using the method outlined in [44]. Then, with known values for z_{com} and $\ddot{z}_{com}$, the ω trajectory can be computed through reverse time integration of the nonlinear ω dynamics, defined by

$$\dot{\omega} = \omega^2 - \alpha^2(t), \quad \omega(t_0) = \omega_0, \tag{2.11}$$

where $\alpha(t) = \sqrt{\frac{\ddot{z}_{com}+g}{z_{com}-z_{ecmp}}}$ is the natural frequency of the LIP. These trajectories can be found at the beginning of double support and not changed, as the current MPC optimization only considers changes in the x-y plane.

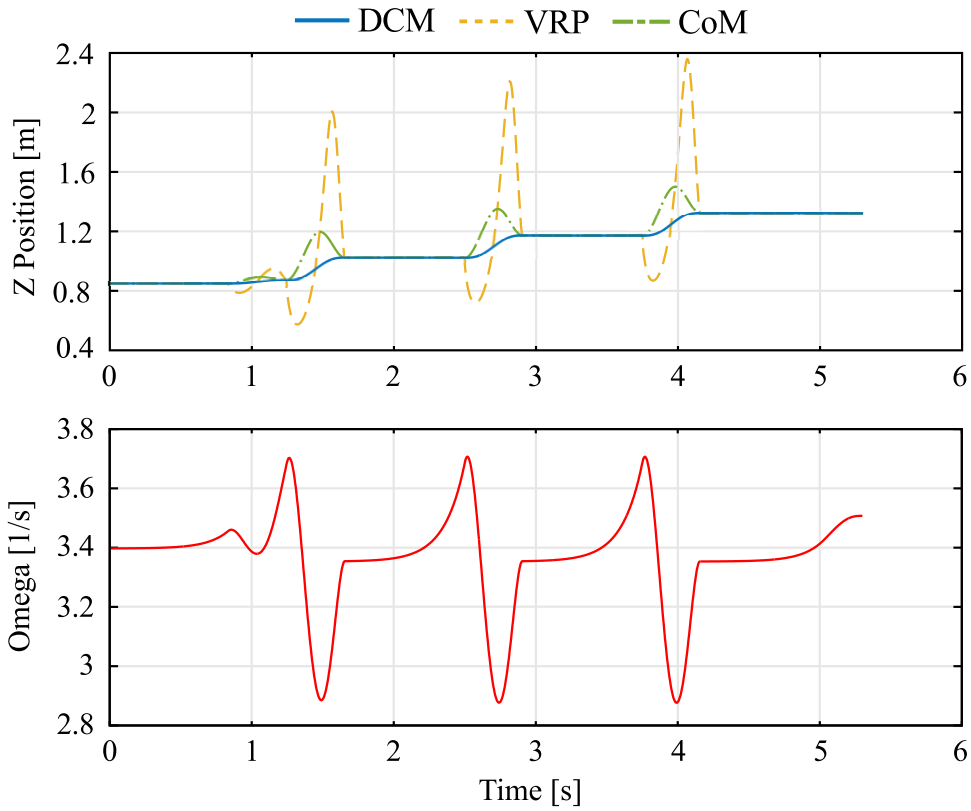


Figure 2.5: Top: DCM, VRP, and CoM height trajectories for a four step plan with changes in height. Bottom: ω trajectory for the same step plan. The VRP is able to better control the DCM height by allowing ω to vary.

2.3.2 Planning DCM Trajectories

When planning CoM and DCM trajectories, angular momentum is typically held at zero, requiring no torque about the CoM. This implies the collocation of the CoP and eCMP, making Equation 2.9

$$\mathbf{r}_{vrp} = \mathbf{r}_{cop} + \frac{\mathbf{g}}{\omega^2 - \dot{\omega}}. \tag{2.12}$$

Once the desired $\omega(t)$ trajectory has been determined, Equation 2.12 allows the VRP trajectory to be found. This result can be used with the $\omega(t)$ trajectory to generate the DCM trajectory using Equation 2.8.

Based on the definition of the DCM dynamics, the time-varying DCM is unstable with respect to the VRP. However, in reverse-time, the VRP acts as an attractor for the DCM,

$$\dot{\boldsymbol{\xi}}_r = -\left(\omega - \tfrac{\dot{\omega}}{\omega}\right)\left(\boldsymbol{\xi} - \mathbf{r}_{vrp}(t_r)\right), \quad \boldsymbol{\xi}(t_f) = \boldsymbol{\xi}_f, \tag{2.13}$$

where $\dot{\boldsymbol{\xi}}_r = -\dot{\boldsymbol{\xi}}$ is the reverse-time derivative of the DCM. The stable DCM trajectory

can then be found by integrating the DCM dynamics backwards in time. The resulting trajectories can be seen in Figure 2.5. As Figure 2.5 shows, the DCM and VRP trajectories no longer have a constant ω, better characterizing the unstable dynamics of the CoM as it changes heights.

Planning the DCM trajectory based on a defined CoP trajectory has the benefit of having a DCM that is always dynamically feasible. In this work, the CoP trajectories are specified to maintain the position in the center of the stance foot for the duration of single support with a slight inward offset, as outlined in [44]. This is not required, however, and motions can move from the heel to the toe of the foot during single support, if desired, as in [42].

2.3.3 Tracking the DCM Trajectory

Assuming the MPC can be updated at each time step, zero tracking error will be produced, as the trajectories start from the initial conditions. However, only the trajectories in the x and y directions are being updated, requiring some kind of control law in the vertical direction. Additionally, updating the MPC this frequently is both not necessary and not feasible, when it can be tracked using a standard feedback law between solutions. This, in turn, improves the robustness to unmodeled dynamics that may result in the robot not behaving precisely according the DCM dynamics. To acheive this, the control law [38],

$$\mathbf{r}_{vrp} = \boldsymbol{\xi} - \frac{1}{\omega - \frac{\dot{\omega}}{\omega}} \left(\dot{\boldsymbol{\xi}}_r + \mathbf{k}_\xi \left(\boldsymbol{\xi}_r - \boldsymbol{\xi} \right) + \mathbf{k}_\Xi \int \left(\boldsymbol{\xi}_r - \boldsymbol{\xi} \right) dt \right), \tag{2.14}$$

is used, where $\boldsymbol{\xi}_r$ and $\dot{\boldsymbol{\xi}}_r$ are the reference DCM position and velocity, respectively, and $\mathbf{k}_\xi$ and $\mathbf{k}_\Xi$ are non-negative feedback gains. The first term in the control law is designed to cancel the nominal DCM dynamics, while the second represents a proportional-integral controller. While this controller has been demonstrated to work well on a variety of terrain [44], it does not allow the use of the desired step positions or rotations as control inputs, relying entirely on the achievable ground reaction forces from step positions found by a high level footstep planner for stabilization.

From this control law, if the DCM is perfectly tracking the desired trajectory, the VRP calculated by the MPC is directly used. If there is tracking error, however, the desired VRP position is modified using the control law in Equation 2.14. The desired VRP output can then be related to the ground reaction force, by finding a desired linear momentum rate of change through

$$\dot{\mathbf{l}}_d = m \left(\omega^2 - \dot{\omega} \right) \left(\mathbf{x} - \mathbf{r}_{vrp} \right). \tag{2.15}$$

2.4 DCM Model Predictive Controller

The MPC scheme in [38] can be generalized in a way similar to that presented in [27] to compute optimal DCM trajectories in x and y for a defined preview window using DCM accelerations and step positions and rotations as the control input. The DCM height trajectories can be planned and tracked using the approach in [44], which is also used to find the final value for the preview window.

The DCM acceleration is chosen as a control input as it relates to minimizing the CoM jerk, resulting in smooth CoM trajectories. To allow for direction changes while walking, necessary for navigating real-world environments, rotation must also be considered. As such, the proposed approach allows the MPC scheme to decide the DCM acceleration, step positions in the x-y plane, and step rotation, making the control inputs $\mathbf{v}_u = \begin{bmatrix} \mathbf{v}_{\ddot{\xi}}^T, & \mathbf{v}_f^T, & \mathbf{v}_\theta^T \end{bmatrix}^T$, where $\mathbf{v}_{\ddot{\xi}} \in \mathbb{R}^{2N}$ is the discretized DCM acceleration vector in x and y of N timesteps, and $\mathbf{v}_f \in \mathbb{R}^{2m}$ is the vector of m foot positions in x and y, and $\mathbf{v}_\theta \in \mathbb{R}^m$ is the vector of m foot rotations, θ. The DCM and VRP trajectory can be assembled from $\mathbf{v}_{\ddot{\xi}}$ and used to compute the desired ground reaction forces.

The following sections describe the formulation of the arguments for the MPC objective function, as well as the constraints on the step positions that allow consideration of step rotations.

2.4.1 VRP Trajectory Definition from Step Positions

To construct the MPC inputs as a function of step positions, we must first find the relationship between these step positions and the DCM dynamics. To be dynamically feasible, the CoP trajectory is determined by the step positions. In the following section, we will define the reference CoP trajectory as a function of these step positions, and will show that this also defines the reference VRP trajectory.

It can be seen from Equation 2.9 that, in the x-y plane, $\mathbf{r}_{vrp} = \mathbf{r}_{ecmp}$, as $\mathbf{g} = [0,\ 0,\ g]^T$. Then, with the assumption that there is no desired angular momentum, the assumption that $\mathbf{r}_{vrp} = \mathbf{r}_{cop}$ is valid in x-y.

The CoP trajectory can be defined by a set of piecewise minimum-jerk trajectories, and can be found from the step positions using a series of linear operators. The operator $\mathbf{F}_e \in \mathbb{R}^{4\times m} : \mathbb{R}^{m\times 2} \mapsto \mathbb{R}^{4\times 2}$, is defined as a map from the step positions in $\mathbf{v}_f$ to a vector of end conditions for each minimum-jerk segment. The operator $\mathbf{M} \in \mathbb{R}^{N\times 4} : \mathbb{R}^{4\times 2} \mapsto \mathbb{R}^{N\times 2}$ maps the vector of end conditions $\begin{pmatrix} \mathbf{p}_0, & \dot{\mathbf{p}}_0, & \mathbf{p}_f, & \dot{\mathbf{p}}_f \end{pmatrix}^T$ to the minimum-jerk trajectory of N discretized points. The reference VRP input along both axes, $\mathbf{v}_{y,s}$, is then defined as

$$\mathbf{v}_{y,s} = \mathbf{M}\left(\mathbf{P}_s + \mathbf{F}_e \mathbf{v}_f\right), \tag{2.16}$$

where $\mathbf{P}_s$ is a matrix consisting of initial CoP position and velocity and initial poses of both feet. Note that through this definition, $\mathbf{r}_{cop}$ is defined as going through the middle of the foot, offset by values in $\mathbf{P}_s$, guaranteeing no-tipping conditions are satisfied without the use of constraints.

The trajectory $\mathbf{v}_{y,s}$ represents the perfect, nominal VRP trajectory to be used, and is defined entirely by the footstep positions. Other, more advanced CoP trajectories can be defined here, such as those consisting of heel-toe motions as in [37]. If desired, this can be used to find a DCM trajectory, providing no VRP feedback action, only feed-forward.

2.4.2 DCM and VRP Recursive Dynamics

In order to compute the MPC arguments, the discrete-time DCM and VRP dynamics must be defined. Due to the uncoupled nature of the DCM dynamics, the discretization is the same along both x and y axes, and is defined in [38] as

$$\begin{bmatrix} \xi_{k+1} \\ \dot{\xi}_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \xi_k \\ \dot{\xi}_k \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix}}_{\mathbf{B}} u_k, \tag{2.17}$$

$$y_k = \underbrace{\begin{bmatrix} 1, & \frac{-\omega_k}{\left(\omega_k^2 - \dot{\omega}_k\right)} \end{bmatrix}}_{\mathbf{C}_k} \begin{bmatrix} \xi_k \\ \dot{\xi}_k \end{bmatrix}, \tag{2.18}$$

where Δt is the sample period, ξ_k and $\dot{\xi}_k$ are the DCM position and velocity, u_k is the DCM acceleration, and y_k is the VRP at time step k. $\mathbf{C}_k$ is computed at each k due to the time-varying formulation of ω. Note that if the time invariant DCM is used, $\mathbf{C}_k$ simplifies to $\mathbf{C}_k \equiv \mathbf{C} = \begin{bmatrix} 1, & \frac{-1}{\omega_0} \end{bmatrix}$, simplifying some of the following equations, but losing versatility when considering uneven terrain. The discrete-time DCM position and velocity trajectory, $\mathbf{v}_\xi \in \mathbb{R}^{2N}$, can be defined along a single axis as a function of linear operators $\mathbf{\Phi}_0^\xi \in \mathbb{R}^{2N\times 2}$ and $\mathbf{\Phi}_u^\xi \in \mathbb{R}^{2N\times N}$ by

$$\mathbf{v}_\xi = \mathbf{\Phi}_0^\xi \boldsymbol{\xi}_0 + \mathbf{\Phi}_u^\xi \mathbf{v}_{\ddot{\xi}}. \tag{2.19}$$

The discrete-time VRP trajectory, $\mathbf{v}_y \in \mathbb{R}^N$ can likewise be defined along a single axis as a function of linear operators $\mathbf{\Phi}_0^y \in \mathbb{R}^{N\times 2}$ and $\mathbf{\Phi}_u^y \in \mathbb{R}^{N\times N}$ by

$$\mathbf{v}_y = \mathbf{\Phi}_0^y \boldsymbol{\xi}_0 + \mathbf{\Phi}_u^y \mathbf{v}_{\ddot{\xi}}, \tag{2.20}$$

where $\boldsymbol{\xi}_0 \in \mathbb{R}^2$ is the initial DCM position and velocity [38]. The definitions of $\mathbf{\Phi}_0^\xi$, $\mathbf{\Phi}_u^\xi$, $\mathbf{\Phi}_0^y$ and $\mathbf{\Phi}_u^y$ can be found in Appendix A.

If we want the desired DCM dynamics to be entirely defined by the step positions, we can set $\mathbf{v}_y = \mathbf{v}_{y,s}$, where $\mathbf{v}_{y,s}$ is the nominal VRP trajectory, defined in Equation 2.16. This,

however, does not allow for any use of the VRP control for the DCM. This is effectively the same as walking with point feet, and limits the controllability of the robot. To direct the DCM dynamics, we now must introduce a way to constrain the final DCM position.

2.4.3 DCM Reverse-Time Integration

As shown in Equation 2.13, the VRP acts as an attractor in reverse-time, so stable trajectories can be found through reverse-time integration of the DCM. Numerical integration can be performed using Huen's method, leading to the discrete dynamics

$$\begin{aligned}\xi_{k-1} = &\underbrace{\left(D_{k-1} + \left(1 - \Delta t D_{k-1}\right) D_k\right)}_{A_{k-1}} \xi_k \\ &+ \underbrace{\left(-\tfrac{1}{2} D_{k-1}\right)}_{B_{k-1}} y_{k-1} + \underbrace{\left(-\tfrac{1}{2}\left(1 - \Delta t D_{k-1}\right)\right)}_{C_{k-1}} y_k,\end{aligned} \tag{2.21}$$

where $D_k = \omega_k - \frac{\dot{\omega}_k}{\omega_k}$, and y_k is the VRP signal at time step k. The reverse-time integrated DCM signal can be calculated via linear operators that encode Equation 2.21 by

$$\mathbf{v}_{\xi,s} = \mathbf{\Phi}_0^v \xi_f + \mathbf{\Phi}_u^v \mathbf{v}_{y,s}, \tag{2.22}$$

where $\mathbf{\Phi}_0^v \in \mathbb{R}^{N\times 1}$ and $\mathbf{\Phi}_u^v \in \mathbb{R}^{N\times N}$ in dimension, $\mathbf{v}_{y,s} \in \mathbb{R}^N$ is the discretized VRP input, and ξ_f is the desired final DCM position. Note that $\dot{\xi}_f$ is encoded in y_f. The definitions of $\mathbf{\Phi}_0^v$ and $\mathbf{\Phi}_u^v$ can be found in Appendix A. The choice of $\boldsymbol{\xi}_f$ is defined by N, and is discussed in subsection 2.4.7.

The state Equation 2.19 and Equation 2.20 are functions of the current DCM state and the DCM acceleration decision variables. The new state Equation 2.22, however, is a function of the final DCM objective and the VRP input constructed from the footstep locations.

Now that the DCM dynamics have been defined as functions of the initial state, final state, and control inputs, what remains is to formulate the objective function and constraints for the MPC.

2.4.4 Objective Function

The MPC scheme can be defined as a quadratic program (QP) with an objective function formulated by

$$\min_{\mathbf{v}_{\ddot{\xi}}, \mathbf{v}_f, \mathbf{v}_\theta} J\left(\mathbf{v}_{\ddot{\xi}}, \mathbf{v}_f, \mathbf{v}_\theta\right) \tag{2.23}$$

$$J\left(\mathbf{v}_u\right) = \mathbf{v}_u^T \mathbf{G} \mathbf{v}_u + \mathbf{g}^T \mathbf{v}_u + g, \tag{2.24}$$

$$
\begin{aligned}
J\left(\mathbf{v}_{\ddot{\xi}}, \mathbf{v}_f, \mathbf{v}_\theta\right) = & \tfrac{1}{2}\left(\mathbf{v}_\xi - \mathbf{v}_{\xi,s}\right)^T \mathbf{Q}_f \left(\mathbf{v}_\xi - \mathbf{v}_{\xi,s}\right) \\
& + \tfrac{1}{2}\left(\mathbf{v}_y - \mathbf{v}_{y,s}\right)^T \mathbf{Q}_v \left(\mathbf{v}_y - \mathbf{v}_{y,s}\right) \\
& + \tfrac{1}{2}\left(\mathbf{v}_f - \mathbf{v}_{f,p}\right)^T \mathbf{Q}_{f,p} \left(\mathbf{v}_f - \mathbf{v}_{f,p}\right) \\
& + \tfrac{1}{2}\left(\mathbf{v}_f - \mathbf{v}_{f,s}\right)^T \mathbf{R}_f \left(\mathbf{v}_f - \mathbf{v}_{f,s}\right) \\
& + \tfrac{1}{2}\left(\mathbf{v}_\theta - \mathbf{v}_{\theta,s}\right)^T \mathbf{R}_\theta \left(\mathbf{v}_\theta - \mathbf{v}_{\theta,s}\right) \\
& + \tfrac{1}{2}\mathbf{v}_{\dot{\xi}}^T \mathbf{Q}_{\dot{\xi}} \mathbf{v}_{\dot{\xi}} + \tfrac{1}{2}\mathbf{v}_{\ddot{\xi}}^T \mathbf{R}_{\ddot{\xi}} \mathbf{v}_{\ddot{\xi}},
\end{aligned}
\tag{2.25}
$$

where $\mathbf{Q}_f$, $\mathbf{Q}_v$, $\mathbf{Q}_{\dot{\xi}}$, $\mathbf{R}_f$, $\mathbf{R}_\theta$, and $\mathbf{R}_{\ddot{\xi}}$ are the positive definite objective weighting matrices, described below. The objective then becomes to find $\mathbf{v}_u^* = \text{argmin} J\left(\mathbf{v}_u\right)$. Separate weights for x and y can be specified in a desired frame, and transformed to the inertial frame via rotation matrices.

The weight $\mathbf{Q}_v$ defines the relationship between the DCM acceleration solution to Equation 2.23, $\mathbf{v}_{\ddot{\xi}}$, and the step positions, penalizing deviations of the solution VRP signal, $\mathbf{v}_y$ found from Equation 2.20, from the nominal VRP trajectory from the step location, $\mathbf{v}_{y,s}$. This is caused by the direct relation between the DCM and VRP signals. By not introducing this as a hard constraint, we allow the VRP to be modified from the nominal trajectory, similar to how standard feedback controllers utilize the VRP to control the DCM.

The weight $\mathbf{R}_{\ddot{\xi}}$ directly minimizes the DCM acceleration to produce smooth CoM motions. The weight $\mathbf{Q}_{\dot{\xi}}$ was introduced on implementation to damp the DCM trajectory by penalizing high frequency oscillations. While this is somewhat redundant to the acceleration objective, it is more effective than $\mathbf{R}_{\ddot{\xi}}$ at preventing these oscillations.

The weights $\mathbf{R}_f$ and $\mathbf{R}_\theta$ penalize deviations from the nominal step locations and rotations, respectively. These weights require careful tuning, as slight changes can greatly affect the high level behavior, whether it attempts to use VRP feedback or step adjustment as the primary means of balance control.

The weight $\mathbf{Q}_{f,p}$ is a regularization weight on the footstep solution. It minimizes the difference between the current solution and the previous footstep solution, $\mathbf{v}_{f,p}$. This term was introduced to reduce small deviations in the solutions that would not otherwise create large costs.

The weight $\mathbf{Q}_f$ penalizes the DCM position at t_f constructed from the solution DCM acceleration, $\mathbf{v}_{\ddot{x}i}$, to track the nominal DCM position at t_f found from the step positions, $\mathbf{v}_{\ddot{\xi}}$. This is done by applying a selection matrix, $\mathbf{S}_f \in \mathbb{R}^{1\times N}$, to restrict tracking to the final value of the trajectory in Equation 2.19 and Equation 2.22 to limit the terminal discontinuities. While this can be (and often is) done via a constraint on the terminal position, such an approach increases the likelihood of producing an empty null space for the solution. As an example, a scenario may arise where the final position cannot be achieved because of the footstep reachability constraints. In this case, a hard constraint on the terminal position would produce an empty null space, while a highly penalized objective would allow a solution to be found.

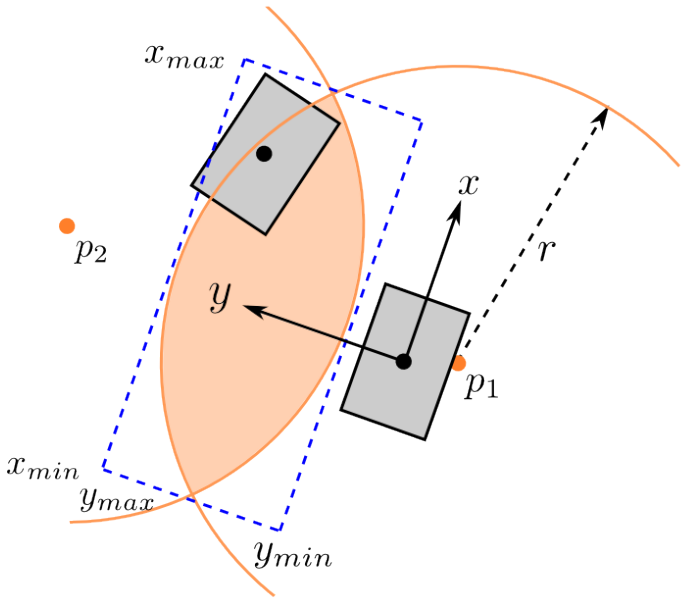


Figure 2.6: The reachability set is described by two intersecting circles, shown by the shaded area, with limits represented by the dotted blue line.

2.4.5 Reachability Constraints

In order to ensure that the step positions found by the MPC scheme are achievable, the positions must be constrained to be kinematically reachable by the robot. When considering a series of steps, this reachability region is strongly affected by the rotation of each step, ultimately determining the path the robot takes for the plan. As such, the rotation of the adjustment step is important to consider when formulating the reachability constraints.

First proposed by Deits et al. [106], the reachable set can be approximated by the intersecting region of two circles, as shown in Figure 2.6. The radii and foci of the two circles can be found by

$$r = -\frac{(y_{min} - \tilde{y})^2 + (x_{max} - \tilde{x})^2}{2\,(y_{min} - \tilde{y})}, \tag{2.26}$$

$$\mathbf{p}_1 = \left[\tilde{x}\ (y_{min} - r)\right]^T, \mathbf{p}_2 = \left[\tilde{x}\ (y_{max} - r)\right]^T, \tag{2.27}$$

where $\tilde{x}$ and $\tilde{y}$ are the averages of the maximum and minimum reachability in the x and y directions, r is the radius of the circle, and $\mathbf{p}_1$ and $\mathbf{p}_2$ are the locations of the circle focii, all defined relative to and in the frame of the fixed foot.

The step position can be constrained to lie in the intersecting region by requiring that, for

each footstep, j,

$$\left\| \begin{bmatrix} x_j \\ y_j \end{bmatrix} - \left(\begin{bmatrix} x_{j-1} \\ y_{j-1} \end{bmatrix} + \begin{bmatrix} \cos\theta_{j-1} & -\sin\theta_{j-1} \\ \sin\theta_{j-1} & \cos\theta_{j-1} \end{bmatrix} \mathbf{p}_i \right) \right\| \leq r. \tag{2.28}$$

When θ_{j-1} is fixed, the left hand side of Equation 2.28 is a convex function w.r.t. the foot locations $\mathbf{v}_f$, which is the case for the stance foot. However, if θ is defined as a decision variable, the trigonometric functions make the constraint non-linear and non-convex. This can be avoided by introducing two variables, s_j and c_j, for each footstep j. These variables approximate $\sin\theta_j$ and $\cos\theta_j$ using constraints presented in the next section, forming

$$\left\| \begin{bmatrix} x_j \\ y_j \end{bmatrix} - \left(\begin{bmatrix} x_{j-1} \\ y_{j-1} \end{bmatrix} + \begin{bmatrix} c_{j-1} & -s_{j-1} \\ s_{j-1} & c_{j-1} \end{bmatrix} \mathbf{p}_i \right) \right\| \leq r, \tag{2.29}$$

which maintains its convexity.

Oftentimes, optimization packages prefer handling quadratically constrained quadratic programs as second order cone problems, which can be solved efficiently using interior point methods. The reachability constraint in Equation 2.29 can be converted to a second order cone constraint, by observing that it can be abstractly written as

$$\|\mathbf{Du} + \mathbf{a}\|^2 \leq r^2 \Rightarrow \mathbf{u}\mathbf{D}^T\mathbf{Du} + 2\mathbf{a}^T\mathbf{Du} + \mathbf{a}^T\mathbf{a} \leq r^2, \tag{2.30}$$

which can be converted to a second order cone by defining

$$\begin{aligned} \mathbf{y} &= \mathbf{Du} \\ \mathbf{y}^T\mathbf{y} &\leq t \\ 2\mathbf{a}^T\mathbf{Du} + t &= r^2 - \mathbf{a}^T\mathbf{a}, \end{aligned} \tag{2.31}$$

where $\mathbf{y} \in \mathbb{R}^2$ and $t \in \mathbb{R}$. This formulation requires the addition of $6m$ continuous variables to the optimization, and replaces the $2m$ complex quadratic inequality constraints with $6m$ linear equality constraints and $2m$ simple quadratic inequality constraints.

Note that, for the first step, the reachability constraint does not require linear approximations of $\sin\theta_0$ and $\cos\theta_0$, as the rotation of the stance foot is fixed.

2.4.6 Linear Rotation Approximation

To calculate s_j and c_j, a set of piecewise linear constraints are introduced [106]. Let $S \in \{0,1\}^{L\times L}$ and $C \in \{0,1\}^{L\times L}$ be binary matrices, where L is the number of segments that compose the approximations. Then the constraints

$$S_{l,j} \Rightarrow \begin{cases} \phi_l \leq \theta_j \leq \phi_{l+1} \\ s_j = g_l\theta_j + h_l \end{cases}, C_{l,j} \Rightarrow \begin{cases} \phi_l \leq \theta_j \leq \phi_{l+1} \\ c_j = g_l\theta_j + h_l \end{cases}, \tag{2.32}$$

are linear approximations of $\sin\theta$ and $\cos\theta$ in L segments, where g_l and h_l are the slope and intercept of segment l. As in [106], we have found $L = 5$ to be a sufficient number of segments to approximate $\sin\theta$ and $\cos\theta$. The *implies* operator in Equation 2.32 can be converted to linear constraints using the standard big-M formulation [118] in the form

$$\phi_l - M\sum_{i=1}^{h} z_i \leq \theta_j \leq \phi_{l+1} + M\sum_{i=1}^{h} z_i, \tag{2.33}$$

where M is some arbitrarily large value and $\{z_h\} = \{z\} - \{z_j\}$ represents all integer variables other than the one currently being considered.

By adding the additional constraints

$$\sum_{l=1}^{L} S_{l,j} = 1, \quad \sum_{l=1}^{L} C_{l,j} = 1, \quad \forall j = 1, \cdots, (m-1), \tag{2.34}$$

the selection of only one of the piecewise approximations in Equation 2.32 is enforced. This formulation results in a mixed-integer quadratically constrained quadratic program (MIQCP) [106].

By using $m > 1$, the MPC can act as a local check on the feasibility of the footstep plan generated by a high level planner. Often, these plans do not include a dynamic model in order to increase computational efficiency for online implementation. By allowing step position and rotation to be chosen by the dynamic planner, footstep plans with better dynamic performance are then found at execution.

2.4.7 Defining the Preview Window

Selecting the appropriate preview window size is critical for proper implementation when treating step location as a decision variable. Sliding preview windows of fixed size have been used [22, 38, 46]; however, when allowing the step position to vary, this is not a viable option for the DCM, as the change in step position results in a change in the final desired value, $\boldsymbol{\xi}_f$, as shown in blue in Figure 2.7. As the dynamic trajectory is ultimately determined by the footstep location, altering this location changes every point on the DCM trajectory until the upcoming double support is finished. To compensate for this, the preview window can be defined to span to the end of the double support following the $m^{th} + 1$ step This results in a constant $\boldsymbol{\xi}_f$, shown in red in Figure 2.7, regardless of the degree of footstep adjustment, as required when considering this adjustment as a cost term.

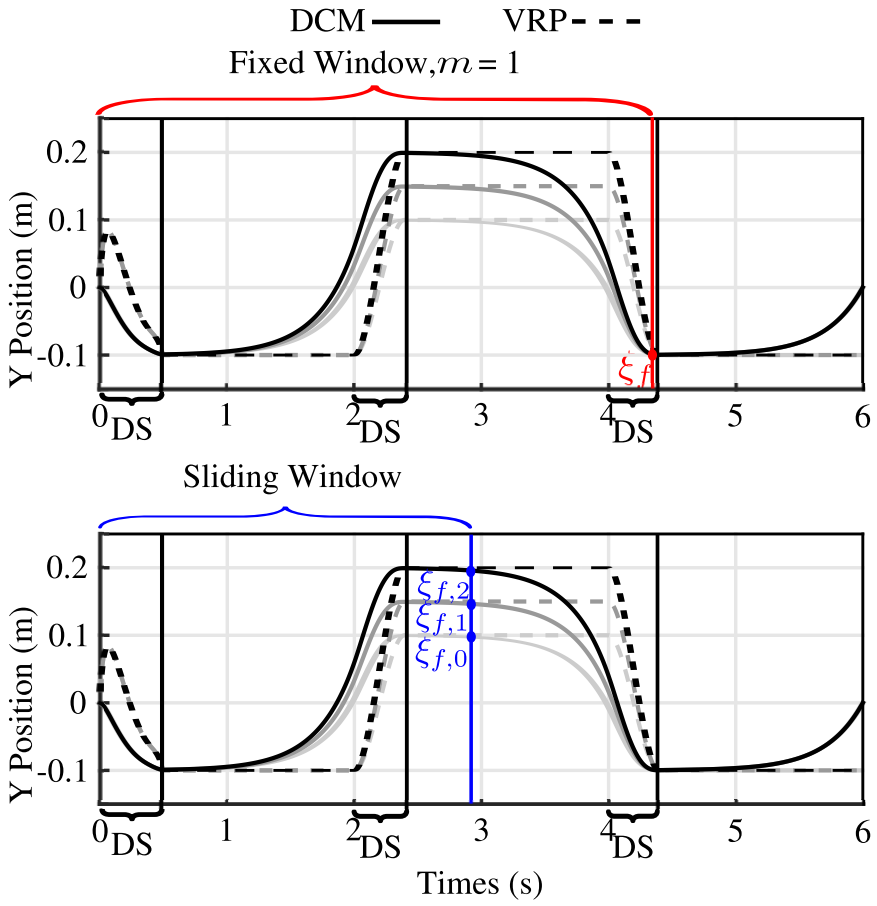


Figure 2.7: DCM and VRP trajectories with footstep adjustment. The darker colored lines denote trajectories with a higher degree of step adjustment. The top figure highlights the final DCM location when considering a preview window of a fixed size, while the bottom one shows a sliding preview window. As can be seen, when using a sliding preview window, step adjustment leads to a change in the final DCM value, a problem not encountered when using a fixed preview window.

2.4.8 Complete Formulation

The entire problem can be written formally as

$$\begin{aligned}\min_{\mathbf{v}_{\ddot{\xi}},\mathbf{v}_f,\mathbf{v}_\theta} \; & \tfrac{1}{2}\left(\mathbf{v}_\xi - \mathbf{v}_{\xi,s}\right)^T \mathbf{Q}_f \left(\mathbf{v}_\xi - \mathbf{v}_{\xi,s}\right) \\ & + \tfrac{1}{2}\left(\mathbf{v}_y - \mathbf{v}_{y,s}\right)^T \mathbf{Q}_v \left(\mathbf{v}_y - \mathbf{v}_{y,s}\right) \\ & + \tfrac{1}{2}\left(\mathbf{v}_f - \mathbf{v}_{f,p}\right)^T \mathbf{Q}_{f,p} \left(\mathbf{v}_f - \mathbf{v}_{f,p}\right) \\ & + \tfrac{1}{2}\left(\mathbf{v}_f - \mathbf{v}_{f,s}\right)^T \mathbf{R}_f \left(\mathbf{v}_f - \mathbf{v}_{f,s}\right) \\ & + \tfrac{1}{2}\left(\mathbf{v}_\theta - \mathbf{v}_{\theta,s}\right)^T \mathbf{R}_\theta \left(\mathbf{v}_\theta - \mathbf{v}_{\theta,s}\right) \\ & + \tfrac{1}{2}\mathbf{v}_{\dot{\xi}}^T \mathbf{Q}_{\dot{\xi}} \mathbf{v}_{\dot{\xi}} + \tfrac{1}{2}\mathbf{v}_{\ddot{\xi}}^T \mathbf{R}_{\ddot{\xi}} \mathbf{v}_{\ddot{\xi}},\end{aligned} \tag{2.35}$$

subject to, for $j = 1, \ldots, (m-1)$

piecewise linear $\sin\theta$ and $\cos\theta$ for $l - 1, \ldots, L$:

$$S_{l,j} \Rightarrow \begin{cases} \phi_l \leq \theta_j \leq \phi_{l+1} \\ s_j = g_l\theta_j + h_l \end{cases}, C_{l,j} \Rightarrow \begin{cases} \phi_l \leq \theta_j \leq \phi_{l+1} \\ c_j = g_l\theta_j + h_l \end{cases}, \tag{2.36}$$

$$\sum_{l=1}^{L} S_{l,j} = 1, \sum_{l=1}^{L} C_{l,j} = 1, \forall j = 1, \cdots, (m-1), \tag{2.37}$$

and, for $j = 1, \ldots, m$

approximate reachability, for $i = 1, 2$:

$$\left\| \begin{bmatrix} x_j \\ y_j \end{bmatrix} - \left(\begin{bmatrix} x_{j-1} \\ y_{j-1} \end{bmatrix} + \begin{bmatrix} c_{j-1} & -s_{j-1} \\ s_{j-1} & c_{j-1} \end{bmatrix} \mathbf{p}_i \right) \right\| \leq r. \tag{2.38}$$

2.5 DCM Prediction

While the formulation in the previous section provides a methodology for determining the desired DCM trajectories and accompanying VRP inputs, there are no guarantees offered on the speed of the solution due to the varying size of N. That is to say, in the time it takes for a solution to the problem to be found, Δt_s, the DCM location may not, and most likely will not, stay the same, but will evolve according to its dynamics. This means that the solution $\mathbf{v}_u^*$ no longer results in the minimum cost, $J^*(\mathbf{v}_u) \neq \min J(\mathbf{v}_u)$. This can be illustrated in Figure 2.8. The DCM starts at the position $\boldsymbol{\xi}^*$, and over Δt_s evolves to $\boldsymbol{\xi}$ while the optimization problem is being solved. The light blue line represents the optimal trajectory, $J^*(\mathbf{v}_u)$, found by the optimization given the initial value, while the dark blue line represents the theoretical optimal trajectory starting from where the DCM is located at the end of the optimization, $\text{argmin} J(\mathbf{v}_u)$. As this shows, the control action associated with the actual solution is not optimal, as the initial DCM for the optimization has moved.

To achieve the truly optimal DCM trajectory, we can submit the actual DCM value, $\boldsymbol{\xi}$, to the optimization. The challenge then becomes determining what this value will be after Δt_s, given the current DCM, $\boldsymbol{\xi}^*$ and the reference VRP location, $\mathbf{r}_{vrp,r}$. To do this, we can utilize the time-invariant DCM, whose dynamics are first order and given by Equation 2.3. These dynamics have a simple closed-form solution

$$\boldsymbol{\xi} = \mathbf{r}_{vrp,r} + e^{\omega \Delta t_s} (\boldsymbol{\xi}^* - \mathbf{r}_{vrp,r}), \tag{2.39}$$

allowing the DCM position after Δt_s to be easily estimated. From this, we can define a new solution using the predicted DCM location, $\mathbf{v}_u'$.

Using this simple solution of the DCM dynamics has several inherent assumptions, however.

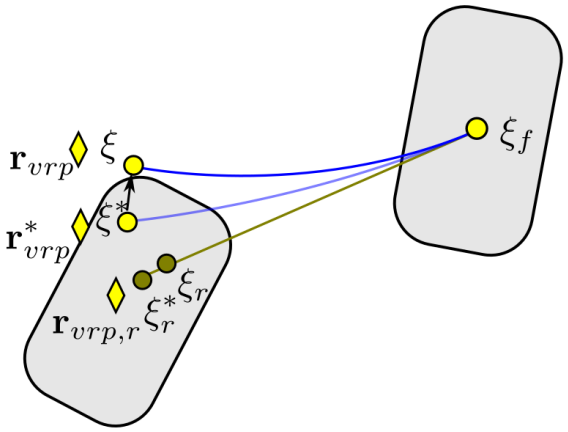


Figure 2.8: The actual DCM, represented by the yellow circles, moves while the new trajectory $\mathbf{v}^*_u$ is being computed, shown by the star annotation. This results in a sub-optimal VRP, $\mathbf{r}^*_{vrp}$ shown by the yellow diamonds, when compared to $\mathbf{r}_{vrp}$, when the controller is trying to drive ξ back to the nominal trajectory, shown by the green circles.

First, we assume that the VRP remains approximately constant at $\mathbf{r}_{vrp,r}$. This implies either no changing control feedback between solutions, or perfect DCM tracking. Secondly, this implies that the dynamics behave exactly like an linear inverted pendulum, in that the height of the center of mass does not change, and that the pendulum frequency ω remains constant. If these assumptions do not hold completely, however, the solution is still not optimal, $\mathbf{v}'_u \neq \text{argmin} J(\mathbf{v}_u)$. As long as the dynamics produce a DCM input that is closer to the actual input that the initial, meaning that the dynamics evolve in the correct direction, the solution remains closer to the true optimal, i.e., $\|\mathbf{v}^*_u - \text{argmin} J\| \geq \|\mathbf{v}'_u - \text{argmin} J\|$.

2.6 Results

The following presents three different sets of experiments for the current MIQCP scheme. The first involve testing the MPC's ability to act as a local check on the footstep plan, also demonstrated the general plan they produce. The second tests the ability of the MPC to stabilize undisturbed walking trajectories. The third and final experiment applies pushes to the robot, testing the MPC's ability to function as a step adjustment algorithm. In all the following, the MIQCP was solved using the commercial optimization package, Gurobi [119], which is designed to be highly efficient at solving this specific type of problem. Dynamic simulations were performed using Gazebo [120].

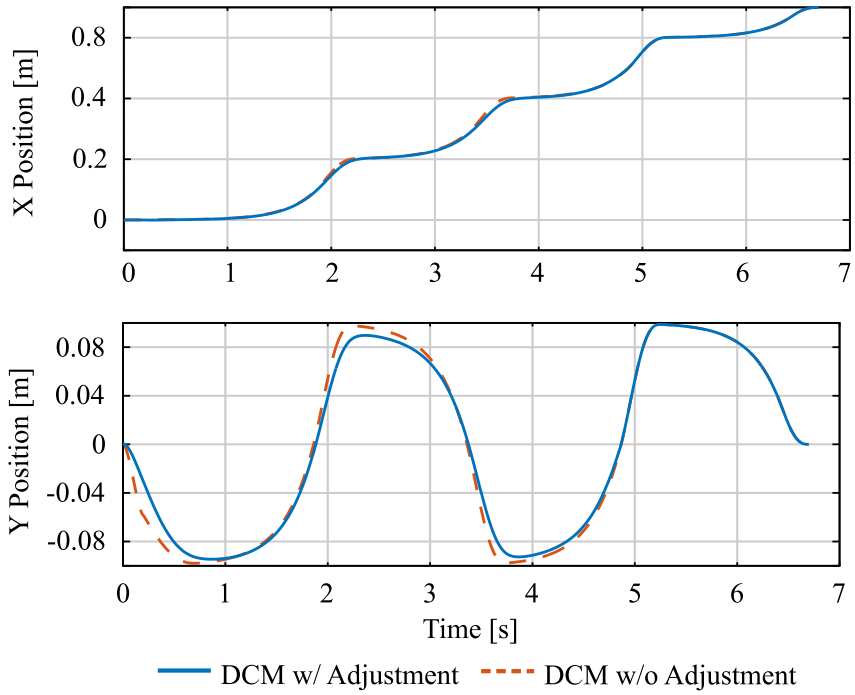


Figure 2.9: DCM trajectories generated using MPC scheme, with and without allowing the step positions to vary.

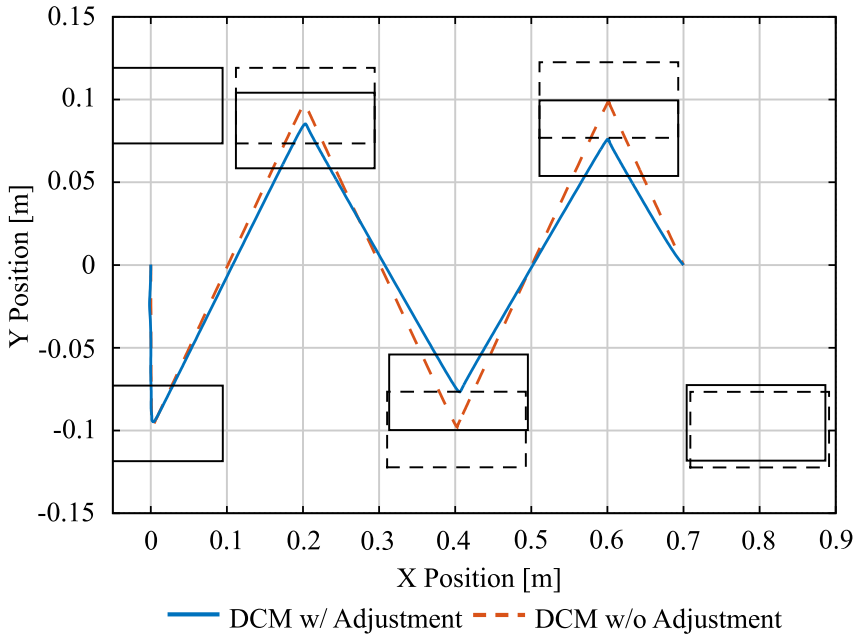


Figure 2.10: Four step plan generated with lower weight on step placement. The original step positions are shown as dashed boxes.

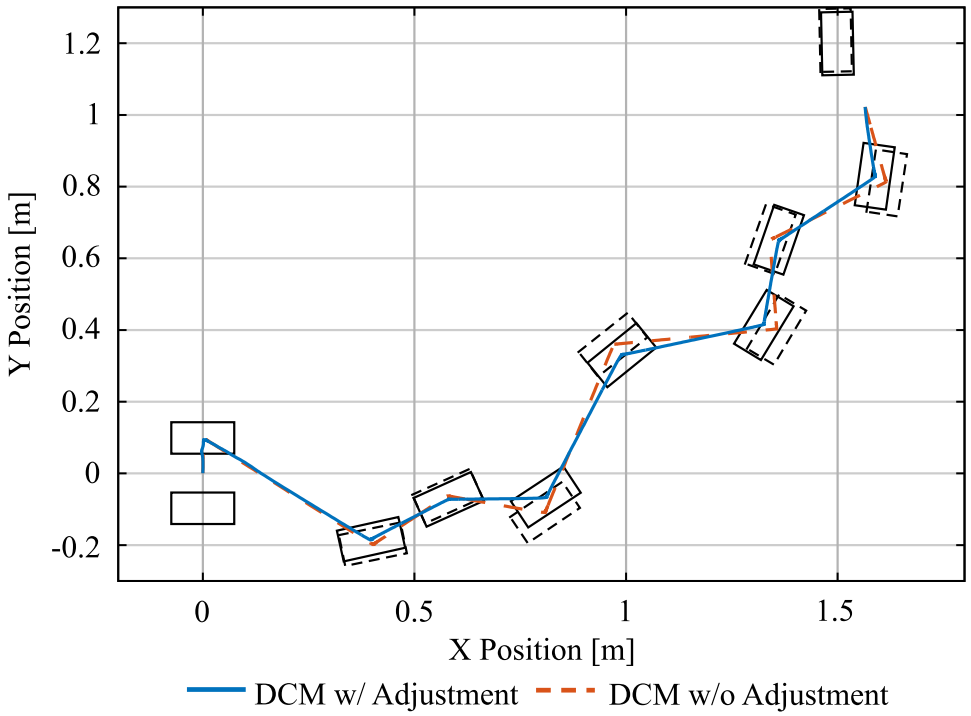


Figure 2.11: Six step plan generated with lower weight on step placement. The original step positions are shown as dashed boxes.

Table 2.1: DCM MPC Planner Weights

Plan	$\mathbf{Q}_{\dot{\xi}}$	$\mathbf{Q}_f$	$\mathbf{Q}_v$	$\mathbf{Q}_{f,p}$	$\mathbf{R}_{\ddot{\xi}}$	$\mathbf{R}_f$	$\mathbf{R}_\theta$	m
1	1e-2	1e6	100	5e-4	5e-2	10	2.5	2
2	1e-2	1e6	100	5e-4	5e-2	5	2.5	4
3	1e-2	1e6	100	5e-4	5e-2	2	2.5	8

2.6.1 Planning Results

Figure 2.9 shows the resulting DCM trajectories for a four step plan of steps $0.2m$ in length, with a $1.5s$ step duration, where the first two steps were being optimized. The objective weights used are listed in Table 2.1. As can be seen, there was some degree of lateral step adjustment to improve the DCM dynamics. This effectively reduces the amount of sway, representing the robot expending less energy in the y direction to change the lateral motion. However, the general DCM trajectory remains largely unchanged, leading to smooth CoM motions. Figure 2.10 shows trajectories generated for a four step plan using the same parameters used to generate the trajectories in Figure 2.9, but with $\mathbf{R}_f$ decreased to 5. In this case, all six steps were considered by the optimization, resulting in inward adjustment, as expected, to reduce lateral sway. The steps are adjusted to a greater degree, however, as to be expected by lowering the weight on the footsteps.

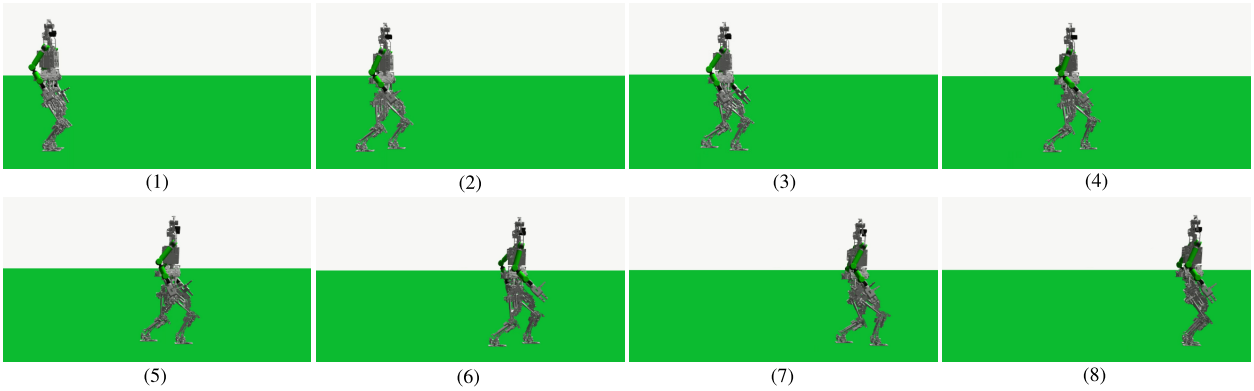


Figure 2.12: Screen capture of forward walking from Figure 2.13. The desired step time is $1s$, with a desired step length of $0.5m$. The robot is able to take long, quick steps, and remain stable using the presented MPC algorithm.

A final experiment is conducted to test the planning results when turning, and is presented in Figure 2.11. In this experiment, eight steps are used, turning $0.2rad$ on each step, resulting in a total turn of $1.6rad$. The original steps are purposefully selected to be sub-optimal for the CoM dynamics. As it is not challenging to fall away from the stance foot, the first step is not heavily adjusted. Subsequent steps are adjusted, however, primarily inward, with some motion in the forward direction as well. The turning angle is not adjusted in this case, a result of the chosen CoP plan design. As the CoP goes from center-of-foot to center-of-foot, the angle of the feet does not affect the plan, resulting in no effect on the DCM dynamics. Were the CoP plan to be a function of the foot angle, such as going from heel-to-toe in the support foot, the step angle may be modified by the optimization.

2.6.2 Simulation Results

When implementing the MPC scheme in the motion framework, the desired momentum rate of change objective is achieved using a model-based whole-body controller presented in [44]. For real time implementation, $\mathbf{v}_{\dot{\xi}}$ is discretized at $\Delta t = 0.05s$, yielding an average solution time of $0.0346s$ when considering m=2 steps that have a $1.5s$ step duration. When reduced to m=1 steps, this solution time is reduced to $0.0192s$. These results are found on a 2.4 GHz four-core 2^{nd} generation i7 processor. The MPC solution is polled every $0.05s$, updating the DCM reference trajectory for tracking and the desired foot positions. The trajectories will not be updated until the MPC solution is available, however it is unlikely that the new MPC solution will not be ready when it is polled at $0.05s$. Between these solutions, the DCM reference trajectory is tracked using the control law defined in Equation 2.14. Using this fixed update time makes the behavior of the MPC uniform, rather than updating at faster rates as the problem gets smaller during execution of the step.

When the MPC solution is found, minimum jerk swing foot pose and torso orientation trajectories are also replanned to the new step position. The MPC is stopped for the last $0.1s$ of swing to prevent rapid step position changes before heel strike. Whole-body motions are executed using a 38-DoF model of ESCHER [74], simulated in Gazebo. ESCHER is a 77.5-kg torque-controlled humanoid developed for the DARPA Robotics Challenge.

The weights used in the whole body controller are 5 for the linear momentum rate of change in the x and y, 1 for the z direction, 4 for the angular momentum rate of change about x, 1 about y, and 0 about z. The DCM tracking controller in Equation 2.14 used $\mathbf{k}_\xi = \begin{bmatrix} 3, & 3, & 3 \end{bmatrix}$, and $\mathbf{k}_\Xi$ set to 0, making it a proportional only controller.

Walking

Figure 2.13 shows the results of a six step plan with a closing step executed by ESCHER, with a step length of $0.5m$, desired CoM height of $0.925m$, total step duration of $1s$ and a double support duration of $0.2s$. The reference CoP trajectories used to define the DCM dynamics are designed to pass through $2cm$ on the inside of the feet, as this has been shown to increase the stability when walking [44]. The MPC weights selected for these simulations are listed in Table 2.2. As seen in Figure 2.13, the robot was able to generate and track smooth DCM setpoints using the presented MPC scheme, resulting in slight adjustment to stabilize the rapid CoM motion from the fast stepping. A screen capture of the walking motion is shown in Figure 2.16. Due to the weight on the DCM acceleration and velocity, the planner does tend to adjust the feet backwards, as the walking speeds require high DCM acceleration. While this effect could be reduced by changing the weights in Table 2.2, primarily $\mathbf{Q}_{\dot{\xi}}$ and $\mathbf{R}_{\ddot{\xi}}$, these weights were kept as they resulted in good push recovery in the following experiments.

Figure 2.14 changes the step length from the previous plan to $0.3m$ and adds a total turning of $0.9rad$, resulting in step adjustment to improve and stabilize the DCM dynamics, producing smooth DCM setpoints. This demonstrates the ability for the MPC scheme to adjust the step positions from the nominal plan for improved dynamics and stabilization. In this case, the feet are mainly adjusted slightly outward, a departure from earlier results in subsection 2.6.1. Most notably, the fourth step is adjusted forward. Both of these changes result in less overall change in the DCM direction, allowing the DCM dynamics to evolve more smoothly.

The final undisturbed walking experiment demonstrates the MPC algorithm's ability to walk over varied terrain, and is illustrated in Figure 2.15, and can be pictured in Figure 2.16. These cinder blocks require two $0.075m$ steps up, one $0.15m$ step up, two $0.075m$ steps down, and

Table 2.2: DCM MPC Controller Weights

$\mathbf{Q}_{\dot{\xi}}$	$\mathbf{Q}_f$	$\mathbf{Q}_v$	$\mathbf{Q}_{f,p}$	$\mathbf{R}_{\ddot{\xi}}$	$\mathbf{R}_f$	$\mathbf{R}_\theta$	m
1e-4	1e6	100	5e-4	1e-3	2.5	2.5	1

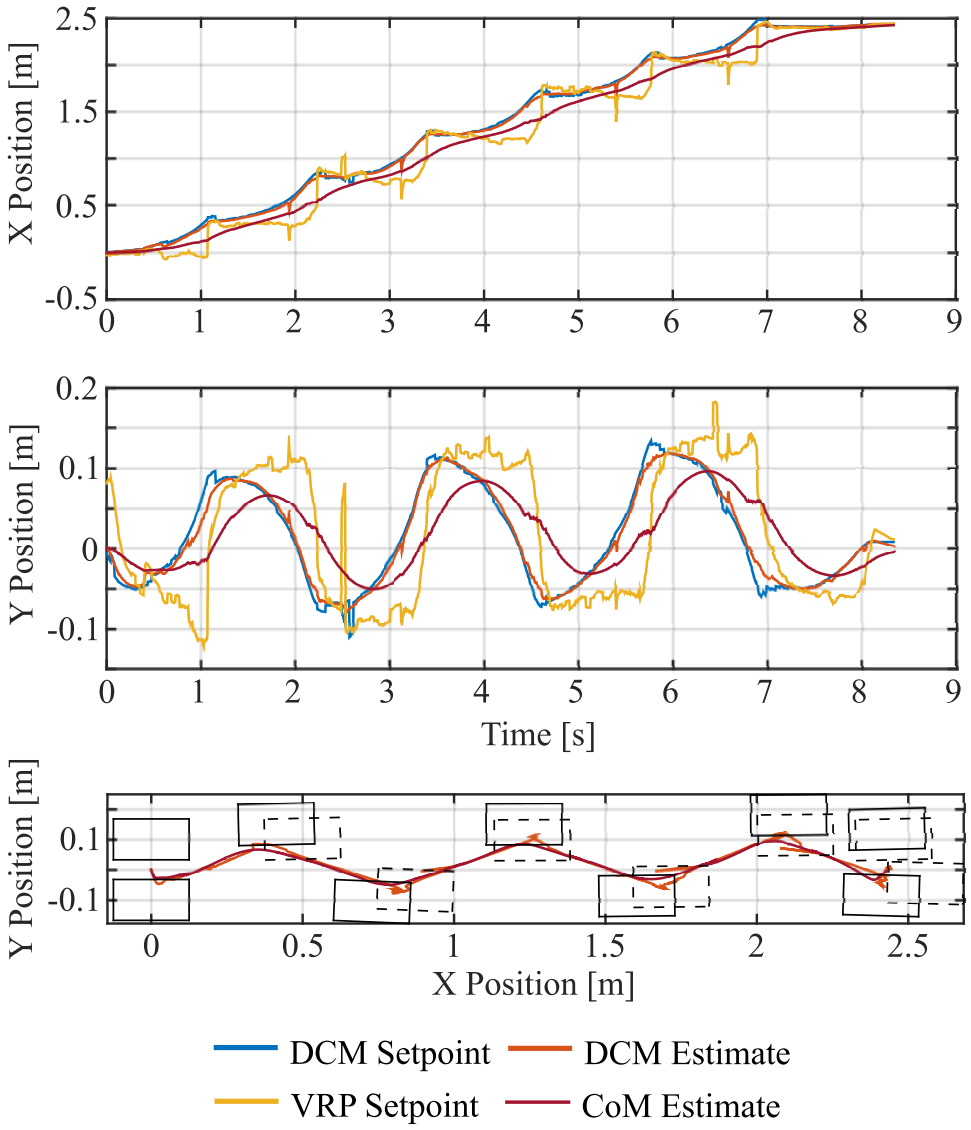


Figure 2.13: Estimated and desired DCM trajectories while walking forward with step adjustment. Each step is $1s$ in duration with a double support duration of $0.25s$.

one last $0.15m$ step down. The step durations for crossing over the cinder blocks are $2.5s$ seconds in duration, with a double support duration of $0.625s$. For this experiment, fairly precise foot placement is required. If the foot is not placed fully on the step, the support polygon is artificially limited, and, unless passing this knowledge to the controller, the desired CoP may be placed outside the base of support, resulting in the foot tipping and the robot falling. Additionally, poor foot tracking results in potential collision hazards with the environment. To account for this, constraints can be placed on the desired footstep location, but this requires some kind of terrain map. The footstep plan could also be regenerated after each step to eliminate tracking error build up. Instead, we increased the weight on the

footstep locations, $\mathbf{R}_f$ to 100. This resulted in fairly precise foot tracking, with the exception of stepping onto the $0.15m$ cinder block, where the location is shifted slightly forward. There is a fairly constant error in the vertical CoM tracking, as well, which is largely due to its low vertical weight in the whole-body controller. The effects of using the time-varying DCM can be seen in the plot of the VRP setpoint, though, as it adapts to the varying natural frequency ω.

Push Recovery

Figure 2.19 shows the resulting trajectories when a $210N$ lateral disturbance, equivalent to 27% of the weight of the robot, is applied to the torso for $0.1s$ midway through the third step. The plan consists of six steps with a total step duration of $1.5s$, double support duration of $0.375s$, and step length of $0.3m$. The disturbance resulted in a $13.3cm$ outward adjustment and $4.5cm$ backwards adjustment, for a total of $14.0cm$ of adjustment. The subsequent step includes the forward error, but converges back to a stable trajectories after one step. As can be seen from the figure, the VRP is moved to the center of the support polygon to control the DCM. The forward plan is mostly unchanged, however. This VRP deviance does result in the generation of a large amount of angular momentum, but the whole-body controller removes this from the system following heel-strike

Figure 2.20 shows the resulting trajectories when a $180N$ lateral and $220N$ forward disturbance, equivalent to 37% of the weight of the robot, is applied to the torso for $0.1s$ midway through the third step of the same step plan. This disturbance resulted in a $14.1cm$ for-

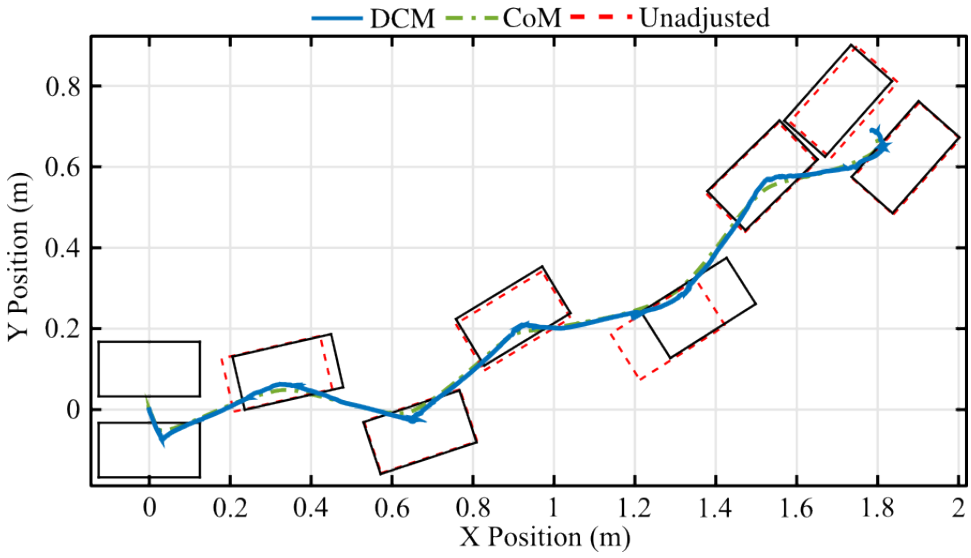


Figure 2.14: Step adjustment while turning. Each step is $1s$ in duration with a double support duration of $0.3s$.

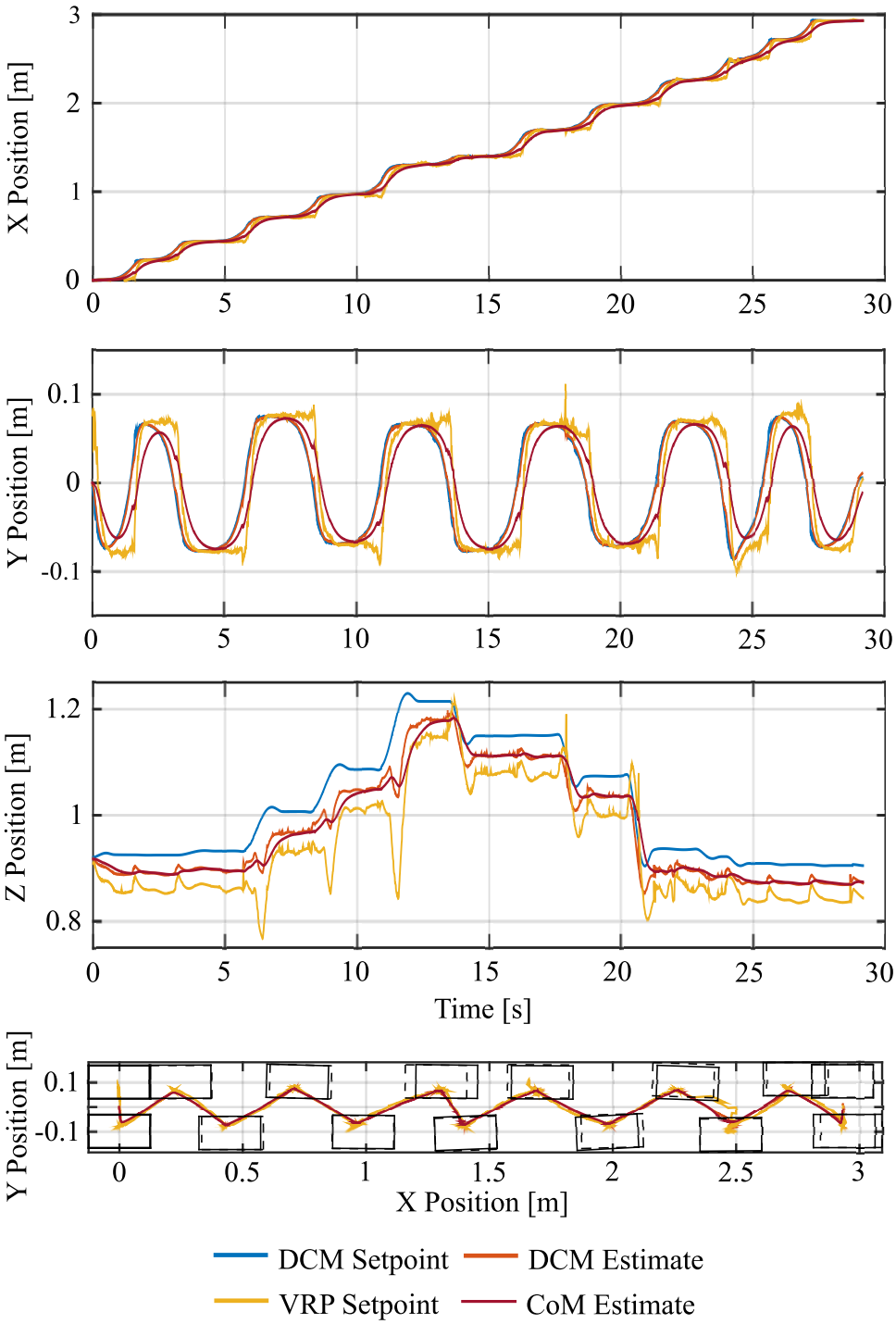


Figure 2.15: Estimated and desired DCM trajectories while walking over a set of cinder blocks. This experiment uses a higher weight on the footstep location to get the correct step locations.

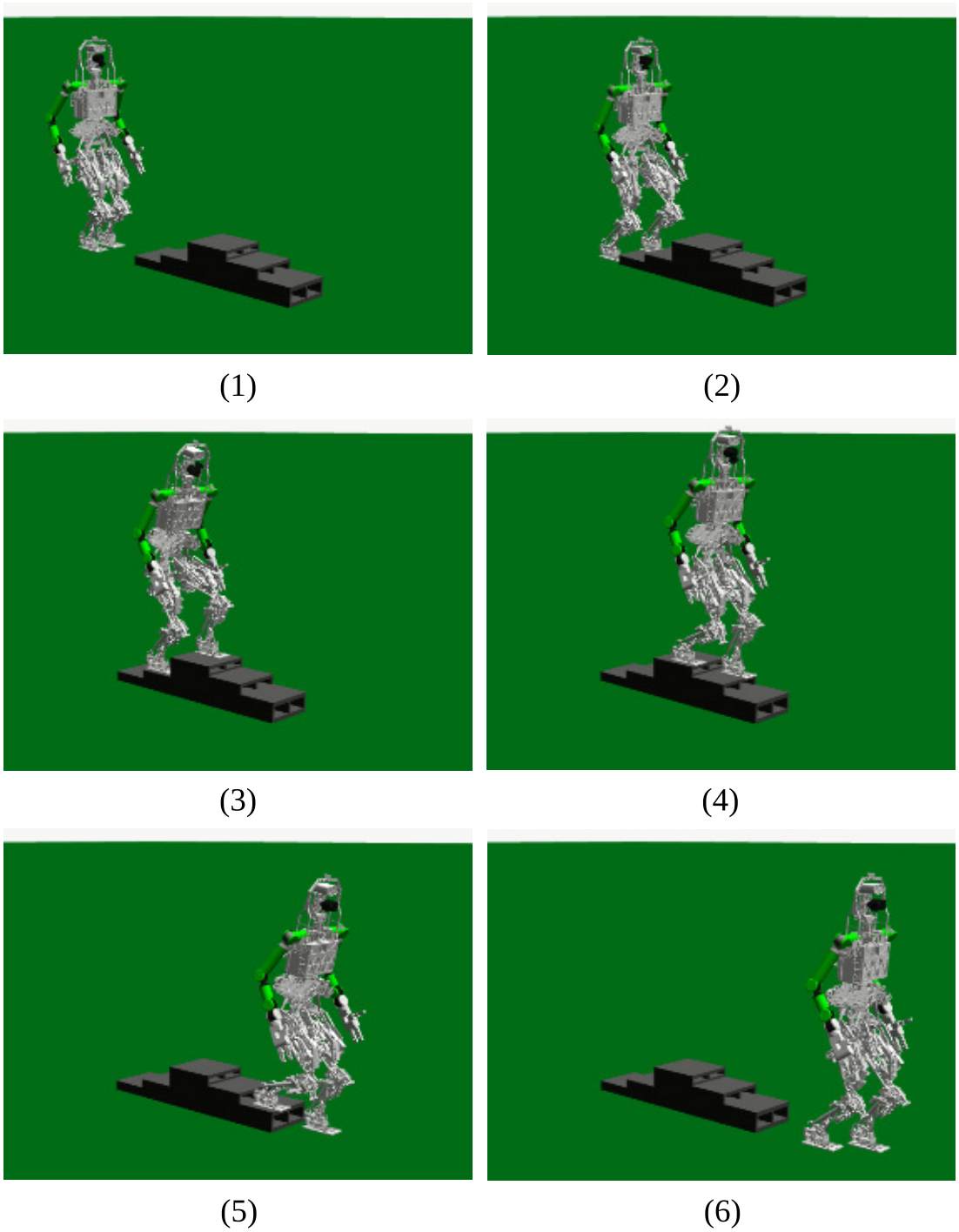


Figure 2.16: Screen capture of the dynamics shown in Figure 2.15. The cinder blocks consist of four $0.075m$ steps and two $0.15m$ steps. Placing a higher weight on the footstep locations enables the required tracking to navigate the cinder blocks.

ward adjustment and 7.9cm outward adjustment, for a total adjustment of 16.2cm. The subsequent step is also adjusted outward, but only negligible adjustment occurs after this.

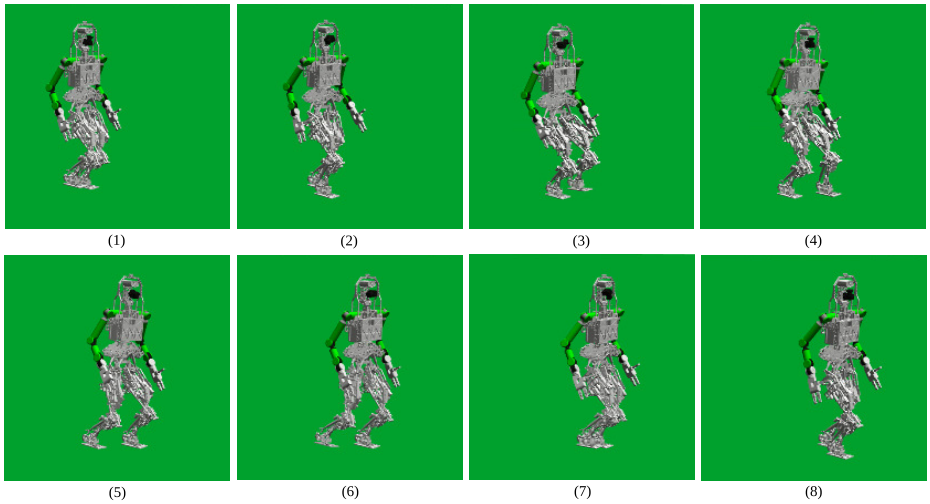


Figure 2.17: Six step plan with a 1.5s step duration, 0.375s double support duration, and 200N lateral push on the third step. The force is applied at the start of pane (3). Pane (6) is the heel strike of the recovery step. The robot steps outward to shift the base of support. The rotation of the upper body in pane (5) illustrates the generation of angular momentum as the stabilizing VRP is moved outside the base of support.

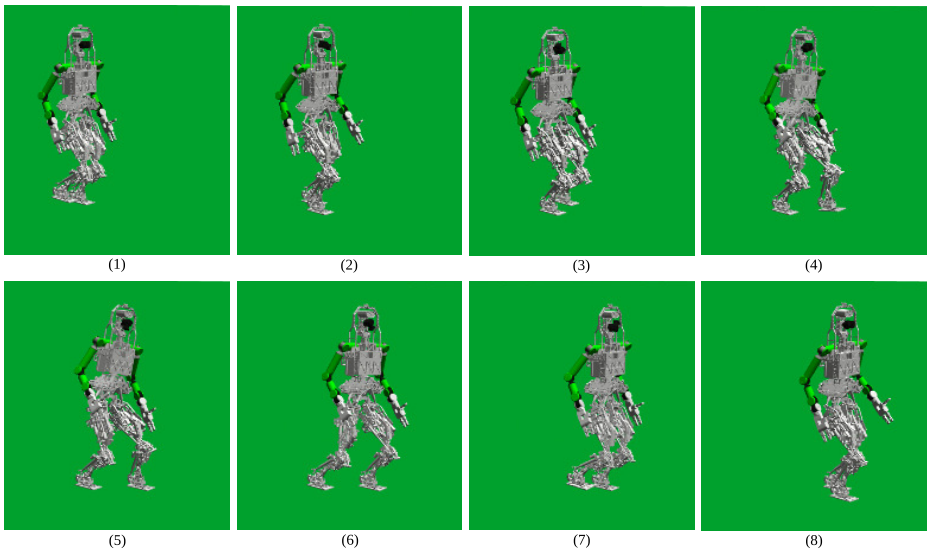


Figure 2.18: Six step plan with a 1.5s step duration, 0.375s double support duration, 180N lateral and 220N forward push on the third step. The force is applied at the start of pane (3). Pane (6) is the heel strike of the recovery step. The robot steps out and forward, in the direction it is pushed. The rotation of the upper body in pane (5) illustrates the generation of angular momentum as the stabilizing VRP is moved outside the base of support.

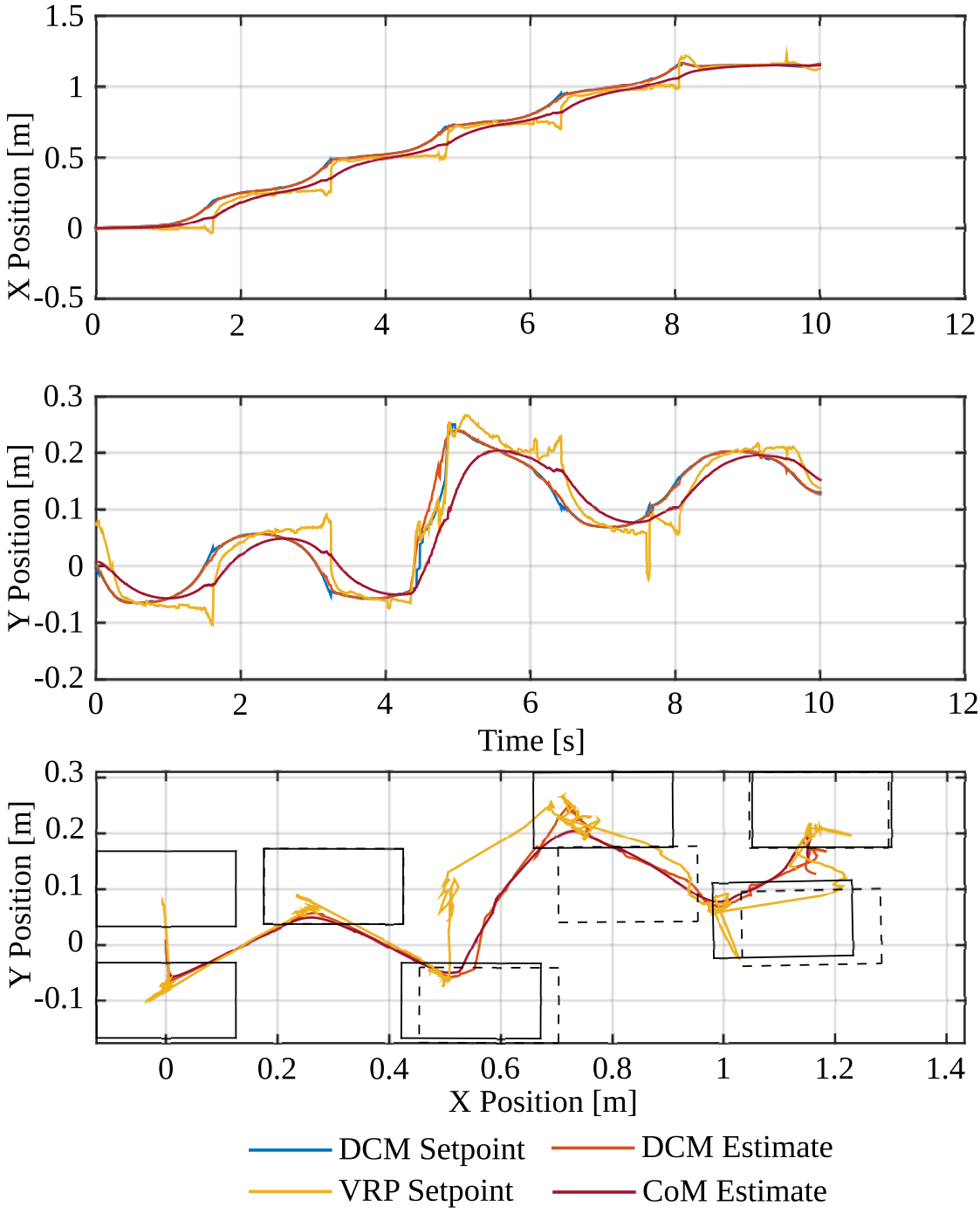


Figure 2.19: Six step plan with a 1.5s step duration, 0.375s double support duration, and 200N lateral push midway through the third step. This results in a large outward adjustment, modifying the base of support to recover and return to stable trajectories.

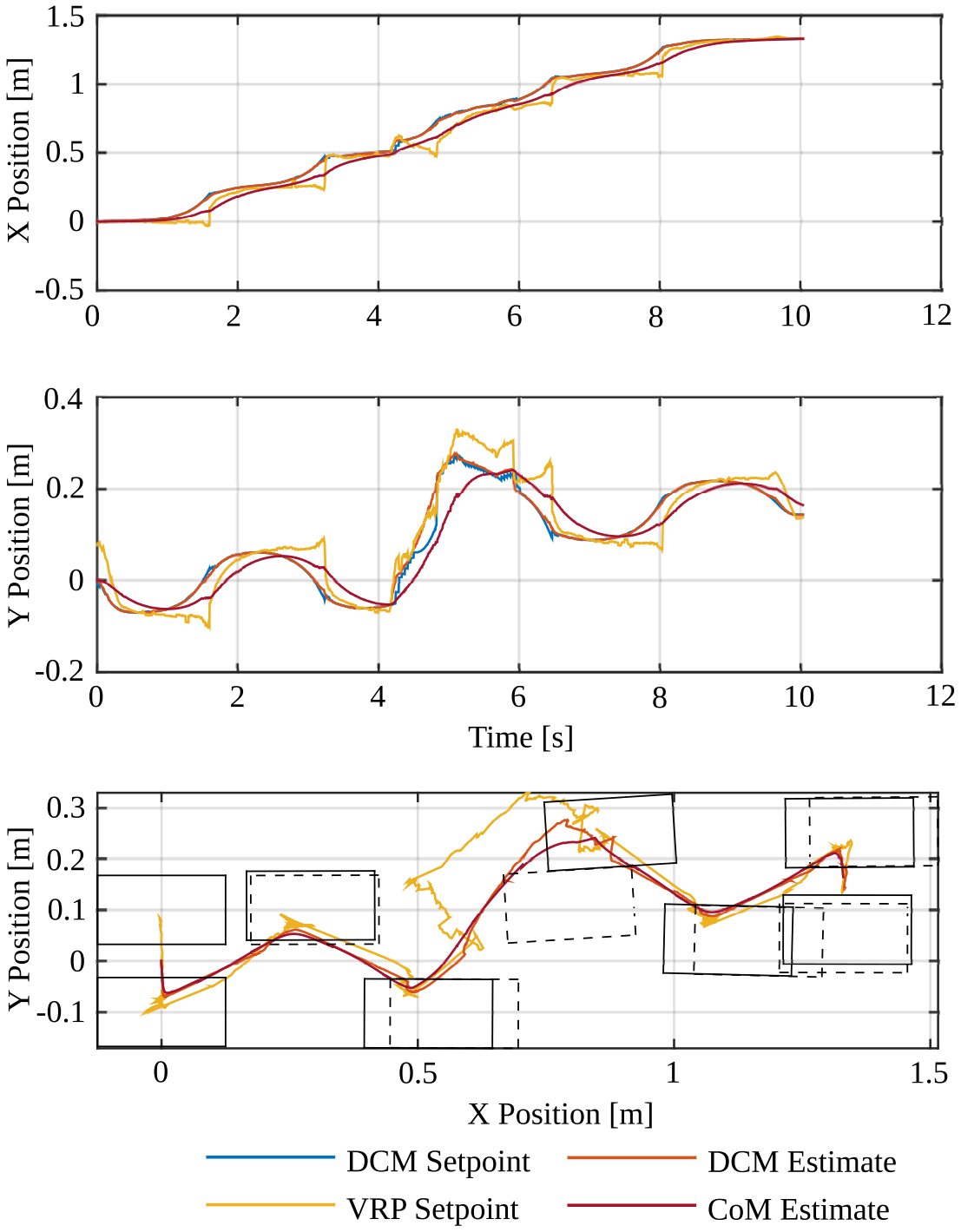


Figure 2.20: Six step plan with a 1.5s step duration, 0.375s double support duration, 180N lateral and 220N forward push on the third step. This resulted in a large outward and slightly forward adjustment, in the direction of the push.

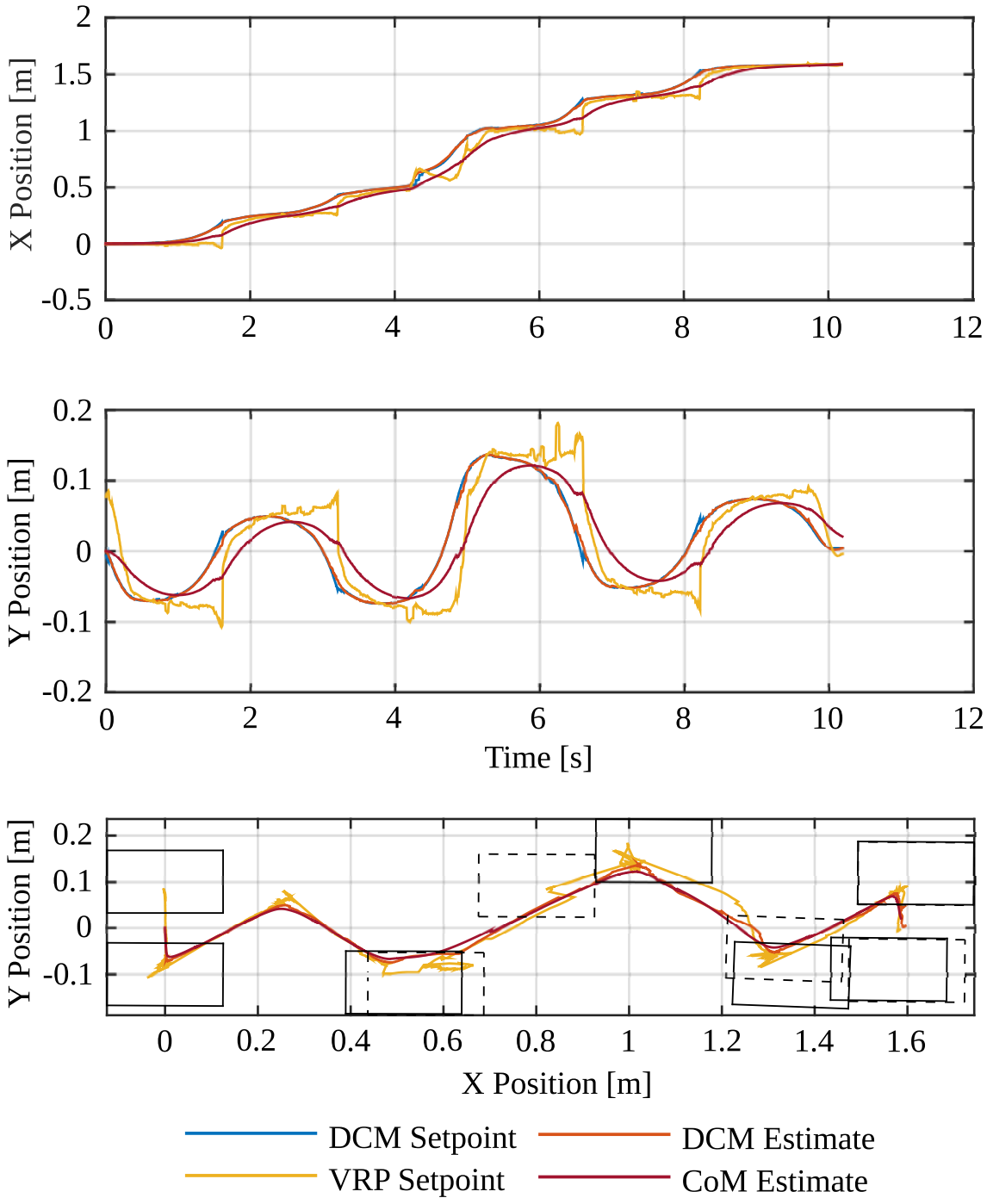


Figure 2.21: Six step plan with a 1.5s step duration, 0.375s double support duration, 270N forward push on the third step. This resulted in a large adjustment forward and slightly outward, in the direction of the DCM velocity.

The final experiment with pushes is illustrated in Figure 2.21, which shows the resulting trajectories when a $270N$ forward disturbance, equivalent to 35% of the weight of the robot, is applied to the torso for $0.1s$ midway through the third step of the same step plan. This disturbance resulted in a $23.7cm$ forward adjustment and $6.6cm$ outward adjustment, for a total adjustment of $24.6cm$. The lateral plan is also adjusted, as it is relatively easy to simply lengthen the step to correspond to the current DCM dynamics, which causes it to grow both forward, and sideways. That is to say, the robot is still following its normal motion, falling towards the foothold, which is simply further away.

The VRP reference trajectories for each of the three pushes is shown in Figure 2.23. As can be seen in each of the three subfigures, the VRP reference trajectory does not stay inside the support polygon when the robot is being pushed. This represents a change from the nominal trajectory, which goes from the center of the support foot to the center of the next support foot, and results in the generation of angular momentum. It is this mechanism that attempts to drive the DCM towards some nominal trajectory. Were the VRP not allowed to vary from the reference value, the required adjustment would be much greater, as this effectively limits the robot to only using step adjustment as a control mechanism, not allowing the CoP to be shifted in the foot for balance.

DCM Prediction

To test the capability of the proposed DCM prediction methodology, a test was run where the robot was given the same footstep plan as in the forward walking test, shown in Figure 2.13 in section 2.6.2. The data from this test is shown in Figure 2.25. As can be shown at the bottom of Figure 2.25, turning off the DCM prediction results in the footsteps being shifted backward when compared to when it is turned on. This can be explained as follows: if the robot is trying to walk forward and it is not using the prediction algorithm, each updated plan starts further back along the reference trajectory. Then, to achieve the same forward motion, higher DCM accelerations and velocities are required. Using the same weights results in the footsteps shifted backward, instead of tracking the actual desired trajectories to minimize the resulting DCM accelerations and velocities.

The accuracy of the DCM prediction module is examined in Figure 2.24. The DCM prediction is computed every time a new dynamic plan is requested. This prediction represents where the DCM is predicted to be at the time the new dynamic plan is polled, Δt_s. As it is used as the initial position of the new dynamic plan, what is desired is for the predicted DCM to match the actual DCM when the new dynamic plan is polled. The discontinuous plot of the predicted DCM represents the $\Delta t = 0.05s$ delay when this occurs. It can be seen from the plot that it accurately predicts the future DCM locations. The exceptions to this are at two points: first, every time there is a heel strike, as this impulse is not modeled in the DCM prediction; second, the remaining $0.1s$ of the swing, as this is when the MPC algorithm is not being updated. This is the flat, unchanging period of the DCM prediction

shown in Figure 2.24.

2.7 Discussion

While the proposed MPC algorithm is able to successfully allow walking over a variety of terrain, it does result in some degree of footstep adjustment, which is not necessarily a desirable attribute in all situations. Using the current formulation type, there is not a clearly defined way using quadratic programs to prioritize the use of only VRP control first and then adjusting the footsteps once VRP feedback has reached its saturation limits. This could be attempted by using an optimization similar to that in [46], and then allowing the footsteps to vary if the VRP is saturated, but this results in a two-staged controller, which may not be desirable.

A strength of the convex optimization formulation is that it easily allows other convex constraints to be added to the problem. An example task where this could be beneficial is when walking up the cinder blocks. Precise foot placement was required, and in this case, achieved via attaching a high cost to any footstep adjustment. An alternative would have been to constrain each footstep to lie in the convex hull of the step. Then, the step would be allowed to vary within this convex region, still satisfying the precise placement requirements, but also allowed to vary to assist in balance. Another example would be walking through a crowded room. There are clearly places in this situation where the robot cannot step if pushed, as that would result in a collision. This can again be described as an allowable convex region in which the step can be placed. While a strength of utilizing the DCM is that we do not have to place explicit constraints on the VRP location to maintain dynamic feasibility, unlike using CoP-based methods, it remains an option, so that the generation of angular momentum is eliminated.

MPC where the entire trajectory is being optimized has some issues, however. A standard feedback controller, such as in Equation 2.14, attempts to drive the DCM back to its nominal value in a certain amount of time, which is determined in this case by k_ξ. This is illustrated in Figure 2.26(a). The MPC, however, replans the entire trajectory so that the DCM ends at the same location, as shown in the dashed blue line in Figure 2.26(b). Accompanying this is a sort of "looseness" of the controller, where little corrective action is taken as long as the DCM can end up at the desired final location. This can be observed by looking at Figure 2.26(a) and Figure 2.26(b), and simply noting where the VRP is located. While the VRP is outside the support polygon in (a), it will only be there for a moment, until DCM tracking is improved. In (b), the desired VRP location for the duration of single support is near the edge of the support polygon. This afford considerably less control authority in the lateral direction if DCM tracking degrade further. Essentially, having the *reference* VRP that is used to generate the trajectory located near the edge of the support polygon is not best practice, as poor tracking at any level could result in the robot tipping. A potential solution to this problem is to design VRP trajectories that have higher weights on

points further in time, so that they return to the nominal trajectory more quickly, as shown in Figure 2.26(c). Another solution would be to define the MPC scheme in such a way that a proportional feedback controller is embedded in the algorithm. Then, the predictable behavior from a simple PI controller is present, while the reference plan is also being modified via foot adjustment.

When utilizing the proposed algorithm as a local check on the footstep plan, we showed that it did modify the plan to improve the dynamics. However, the adjustments were relatively small, as it incurred a high cost when adjusting the steps. Were the dynamics included in the original footstep planner, this would not be the case, as the selection of the steps location general would be driven primarily by the dynamics and reachability. This shows extreme promise towards a unification of both the footstep and dynamic planning steps, which are so closely related.

The simple DCM prediction algorithm was shown to work fairly well. However, if the dynamics do not behave as predicted by the LIP model, the prediction algorithm could exacerbate poor tracking issues, as it could predict the DCM dynamics evolving in the opposite direction the actual. The MPC would then generate an extremely sub-optimal plan for the actual DCM location. Additionally, the prediction algorithm could be improved by incorporating some kind of learning mechanism. As the only inputs to the predictor are $\boldsymbol{\xi}, \mathbf{r}_{vrp}$, and ω, a Kalman filter or other adaptive function should be feasible. This would then improve the potential errors, such as improper estimation of $\boldsymbol{\xi}$ or ω due to poor CoM state estimation, or poor joint torque tracking resulting in the desired $\mathbf{r}_{vrp}$ not being achieved. This is critical for best implementation on a real robot, where sensor measurements are much less certain than in simulation.

Lastly, footstep adjustment, while an extremely beneficial element to be included, is most effective when the swing speed of the steps is relatively high. Due to the exponential relationship between the DCM position with time, a slight increase in swing time causes an exponential increase in required step length. This in part explains why fast stepping robots can be so much more stable; not only are they able to rapidly change the base of support, they are not required to change it as significantly to account on the same tracking errors. Slower swing speeds, as required for the ESCHER hardware platform, are only so effective for footstep adjustment, as a large adjustment is required for small tracking errors. For this reason, all experiments presented in this paper were conducted in simulation.

2.8 Conclusion

This work presents an extension of the previous model predictive control (MPC) scheme presented in [103], which utilized the time-varying divergent component of motion (DCM) to maintain the robot's balance. These additions greatly expand the versatility of MPC schemes for humanoid walking, as well as provide control mechanisms for more robust humanoid

walking.

Controlling only the unstable center of mass (CoM) dynamics through regulating the momentum rate of change has been shown to be very effective, and using the time-varying DCM further improves control of the CoM height compared to the time-invariant instantaneous capture point and DCM methods. By utilizing the time-varying DCM formulation, the proposed MPC formulation capitalizes on this success and expands the capabilities of MPC. However, the control authority over the CoM is limited when using a fixed base of support, so the proposed MPC formulation includes the upcoming footstep positions as a control variable. Doing so allows the controller to modify the foot positions in response to large disturbances. This is critical for humanoid robots to be able to function in unstructured environments like what may be encountered outside the lab. Including it in novel a convex optimization framework allows determination of the stabilizing footstep locations through the CoM dynamics, rather than typically used heuristic methods. The proposed MPC scheme was further expanded to a mixed-integer formulation to allow for consideration of the step rotation as a control input. This increases the flexibility when designing the nominal center of pressure trajectory to include things such as heel-toe walking.

Additionally, this work uniquely utilizes the MPC algorithm as a local check on the footstep plan. These plans do not typically include any knowledge of the CoM dynamics, and may produce plans that are heuristically optimal, but hard for the robot to execute. To this end, the inclusion of footstep rotation is critical, as the reachability from the stance foot ultimately determines the path that the robot will take through an environment. Through this formulation, we found that the inclusion of the CoM dynamics in the planner can provide improvements to the plans generated without substantial increases time spent tuning.

One of the challenges in MPC is the long solve time for many of the optimization based strategies. To address this, we introduce a DCM prediction module that estimates what the state will be when the solution is actually obtained. We then utilize a PI feedback controller between solutions. This allows us to consider longer plans with more information, as we do not need a solution at each time step, increasing the utility of the actual MPC.

A series of experiments were run to test the proposed algorithm. We optimized a variety of footstep plans, and found that modifications were made to the footstep locations to improve the dynamics, demonstrating that inclusion of the CoM dynamics in the footstep planner can lead to better footstep plans. We then evaulated the performance of the MPC for general walking using simulations of the ESCHER humanoid. The controller was able to achieve fast, stable walking motions over flat terrain, as well as over terrain with varying-height. Additionally, we tested the ability for the MPC algorithm to use step adjustment to reject large disturbances. Through a series of pushes, the controller was shown to be able to modify the base of support to allow ESCHER to continue walking where it otherwise would have fallen.

A major issue remaining for step adjustment strategies are methods to adjust the swing time for stabilization. The nonlinearities this introduces pose challenges when formulating

efficient model-based controllers to run real-time. We are planning to address this in future work. Improved methods to predict the future DCM are also being investigated, as this has important implications not only for control, but also state estimation. Methods to improve the formulation to achieve faster solve times are also being considered. Of particular interest are methods to interpolate the trajectories between the discretized points, similar to direct collocation techniques. We are also exploring methods to embed a more traditional feedback control mechanism into the MPC itself to increase the controller's response to tracking errors. Improvements to the reachability constraint that could allow cross-over are also of interest. The expansion of the proposed MPC scheme into a full footstep planner that considers the CoM dynamics is also being explored.

2.9 Appendix A

The discretized DCM trajectory can be defined solely as a function of the discretized DCM acceleration trajectory, $\mathbf{v}_{\ddot{\xi}}$, and the initial DCM state, $\boldsymbol{\xi}_0$ through two linear operators,

$$\mathbf{v}_{\xi} = \boldsymbol{\Phi}_0^{\xi} + \boldsymbol{\Phi}_u^{\xi} \mathbf{v}_{\ddot{\xi}}, \tag{2.40}$$

where

$$\boldsymbol{\Phi}_0^{\xi} = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^N \end{bmatrix} \tag{2.41}$$

and

$$\boldsymbol{\Phi}_u^{\xi} = \begin{bmatrix} \mathbf{B} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{AB} & \mathbf{B} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{N-1}\mathbf{B} & \mathbf{A}^{N-2}\mathbf{B} & \cdots & \mathbf{B} \end{bmatrix}, \tag{2.42}$$

resulting in $\boldsymbol{\Phi}_0^{\xi} \in \mathbb{R}^{2N \times 2}$ and $\boldsymbol{\Phi}_u^{\xi} \in \mathbb{R}^{2N \times N}$. In this case, the matrices $\mathbf{A}$ and $\mathbf{B}$ are defined from the discrete DCM dynamics. As the matrices $\boldsymbol{\Phi}_0^{\xi}$ and $\boldsymbol{\Phi}_u^{\xi}$ are only functions of the discretization size, they do not have to be assembled while the robot is running, and can be constructed during compilation. This decreases computational load at run-time.

In the same way, the discretized VRP trajectory can be defined as

$$\mathbf{v}_y = \boldsymbol{\Phi}_0^{y} + \boldsymbol{\Phi}_u^{y} \mathbf{v}_{\ddot{\xi}}, \tag{2.43}$$

where

$$\mathbf{\Phi}_0^y = \begin{bmatrix} \mathbf{C}_1\mathbf{A} \\ \mathbf{C}_2\mathbf{A}^2 \\ \vdots \\ \mathbf{C}_N\mathbf{A}^N \end{bmatrix} \tag{2.44}$$

and

$$\mathbf{\Phi}_u^y = \begin{bmatrix} \mathbf{C}_1\mathbf{B} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{C}_2\mathbf{A}\mathbf{B} & \mathbf{C}_2\mathbf{B} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_N\mathbf{A}^{N-1}\mathbf{B} & \mathbf{C_N}\mathbf{A}^{N-2}\mathbf{B} & \cdots & \mathbf{C}_N\mathbf{B} \end{bmatrix}. \tag{2.45}$$

resulting in $\mathbf{\Phi}_0^y \in \mathbb{R}^{N\times 2}$ and $\mathbf{\Phi}_u^y \in \mathbb{R}^{N\times N}$. The matrix $\mathbf{C}_k$ comes from the discrete VRP dynamics. The matrices $\mathbf{\Phi}_0^y$ and $\mathbf{\Phi}_u^y$ can also be found by multiplying $\mathbf{\Phi}_0^\xi$ and $\mathbf{\Phi}_u^\xi$ by $\mathbf{C}_k$, again to decrease computational load during run-time. However, if the pendulum height is assumed to not vary, then $\mathbf{C}_k \equiv \mathbf{C}, \forall k$, greatly simplifying the calculation of $\mathbf{\Phi}_0^y$ and $\mathbf{\Phi}_u^y$.

The discrete DCM trajectory can also be constructed from a VRP trajectory, $\mathbf{v}_{y,s}$, through reverse-time integration, given the final state $\boldsymbol{\xi}_f$, as outlined previously. This can also be done through the use of linear operators that encode the reverse time integration. Recall

$$\xi_{k-1} = A_{k-1}\xi_k + B_{k-1}y_{k-1} + C_{k-1}y_k, \tag{2.46}$$

then we can construct the discrete DCM signal as

$$\mathbf{v}_{\xi,s} = \mathbf{\Phi}_0^v\boldsymbol{\xi}_f + \mathbf{\Phi}_u^v\mathbf{v}_y, s, \tag{2.47}$$

where $\mathbf{\Phi}_0^v$ is defined in Equation 2.48 and $\mathbf{\Phi}_u^v$ is defined in Equation 2.49. This results in $\mathbf{\Phi}_0^v \in \mathbb{R}^{N\times 1}$ and $\mathbf{\Phi}_u^v \in \mathbb{R}^{N\times N}$.

2.10 Appendix B

For illustration, the objective function $J(\mathbf{v}_u)$ can be expanded and written as a function of the cost terms by

$$\begin{aligned} J(\mathbf{v}_u) = &\begin{bmatrix} \mathbf{v}_{\ddot{\xi}} \\ \mathbf{v}_f \\ \mathbf{v}_\theta \end{bmatrix}^T \underbrace{\begin{bmatrix} \mathbf{G}_1 & \mathbf{G}_2 & \mathbf{0} \\ \mathbf{G}_2^T & \mathbf{G}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{G}_4 \end{bmatrix}}_{\mathbf{G}} \begin{bmatrix} \mathbf{v}_{\ddot{\xi}} \\ \mathbf{v}_f \\ \mathbf{v}_\theta \end{bmatrix} \\ &+ \underbrace{\begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix}^T}_{g} \begin{bmatrix} \mathbf{v}_{\ddot{\xi}} \\ \mathbf{v}_f \\ \mathbf{v}_\theta \end{bmatrix} + g, \end{aligned} \tag{2.50}$$

where $\mathbf{G} \in \mathbb{R}^{(N+3m)\times(N+3m)}$ is the composite quadratic cost matrix, $\mathbf{g} \in \mathbb{R}^{N+3m}$ is the composite linear cost matrix, and $g \in \mathbb{R}$ is the residual cost. These are defined by

$$
\begin{aligned}
\mathbf{G}_1 &= \tfrac{1}{2}\left(\mathbf{S}_e\boldsymbol{\Phi}_u^{\xi}\right)^T\mathbf{Q}_f\left(\mathbf{S}_e\boldsymbol{\Phi}_u^{\xi}\right) + \tfrac{1}{2}\left(\boldsymbol{\Phi}_u^{y}\right)^T\mathbf{Q}_v\left(\boldsymbol{\Phi}_u^{y}\right) + \\
&\quad \tfrac{1}{2}\left(\mathbf{S}_v\boldsymbol{\Phi}_u^{\xi}\right)\mathbf{Q}_{\dot{\xi}}\left(\mathbf{S}_v\boldsymbol{\Phi}_u^{\xi}\right) + \tfrac{1}{2}\mathbf{R}_{\ddot{\xi}}, \\
\mathbf{G}_2 &= -\tfrac{1}{2}\left(\mathbf{S}_e\boldsymbol{\Phi}_u^{\xi}\right)^T\mathbf{Q}_f\left(\boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{F}_e\right) - \\
&\quad \tfrac{1}{2}\left(\boldsymbol{\Phi}_u^{y}\right)^T\mathbf{Q}_v\left(\mathbf{M}\mathbf{F}_e\right), \\
\mathbf{G}_3 &= \tfrac{1}{2}\left(\boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{F}_e\right)^T\mathbf{Q}_f\left(\boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{F}_e\right) + \\
&\quad \tfrac{1}{2}\left(\mathbf{M}\mathbf{F}_e\right)^T\mathbf{Q}_v\left(\mathbf{M}\mathbf{F}_e\right) + \tfrac{1}{2}\mathbf{R}_f, \\
\mathbf{G}_4 &= \tfrac{1}{2}\mathbf{R}_\theta \\
\mathbf{g}_1 &= \left(\mathbf{S}_e\boldsymbol{\Phi}_0^{\xi}\boldsymbol{\xi}_0 - \boldsymbol{\Phi}_0^{v}\boldsymbol{\xi}_f - \boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{P}_s\right)^T\mathbf{Q}_f\left(\mathbf{S}_e\boldsymbol{\Phi}_u^{\xi}\right) + \\
&\quad \left(\boldsymbol{\Phi}_0^{u}\boldsymbol{\xi}_0 - \mathbf{M}\mathbf{P}_s\right)^T\mathbf{Q}_v\left(\boldsymbol{\Phi}_u^{y}\right) + \\
&\quad \left(\mathbf{S}_v\boldsymbol{\Phi}_0^{\xi}\boldsymbol{\xi}_0\right)^T\mathbf{Q}_{\dot{\xi}}\left(\mathbf{S}_v\boldsymbol{\Phi}_u^{\xi}\right), \\
\mathbf{g}_2 &= -\left(\mathbf{S}_e\boldsymbol{\Phi}_0^{\xi}\boldsymbol{\xi}_0 - \boldsymbol{\Phi}_0^{v}\boldsymbol{\xi}_f - \boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{P}_s\right)^T\mathbf{Q}\left(\boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{F}_e\right) - \\
&\quad \left(\boldsymbol{\Phi}_0^{u}\boldsymbol{\xi}_0 - \mathbf{M}\mathbf{P}_s\right)^T\mathbf{Q}_v\left(\mathbf{M}\mathbf{F}_e\right) - \mathbf{v}_{f,s}\mathbf{R}_f
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{g}_3 &= -\mathbf{v}_{\theta,s}\mathbf{R}_\theta \\
g &= \left\|\mathbf{S}_e\boldsymbol{\Phi}_0^{\xi}\boldsymbol{\xi}_0 - \boldsymbol{\Phi}_0^{v}\boldsymbol{\xi}_f - \boldsymbol{\Phi}_u^{v}\mathbf{M}\mathbf{P}_s\right\|^2_{\mathbf{Q}_f+\mathbf{Q}_v} \\
&\quad + \left\|\boldsymbol{\Phi}_0^{u}\boldsymbol{\xi}_0 - \mathbf{M}\mathbf{P}_s\right\|^2_{\mathbf{Q}_v} + \left\|\mathbf{S}_v\boldsymbol{\Phi}_0^{\xi}\boldsymbol{\xi}_0\right\|^2_{\mathbf{Q}_{\dot{\xi}}} \\
&\quad + \left\|\mathbf{v}_{f,s}\right\|^2_{\mathbf{R}_f} + \left\|\mathbf{v}_{\theta,s}\right\|^2_{\mathbf{R}_\theta}
\end{aligned}
$$

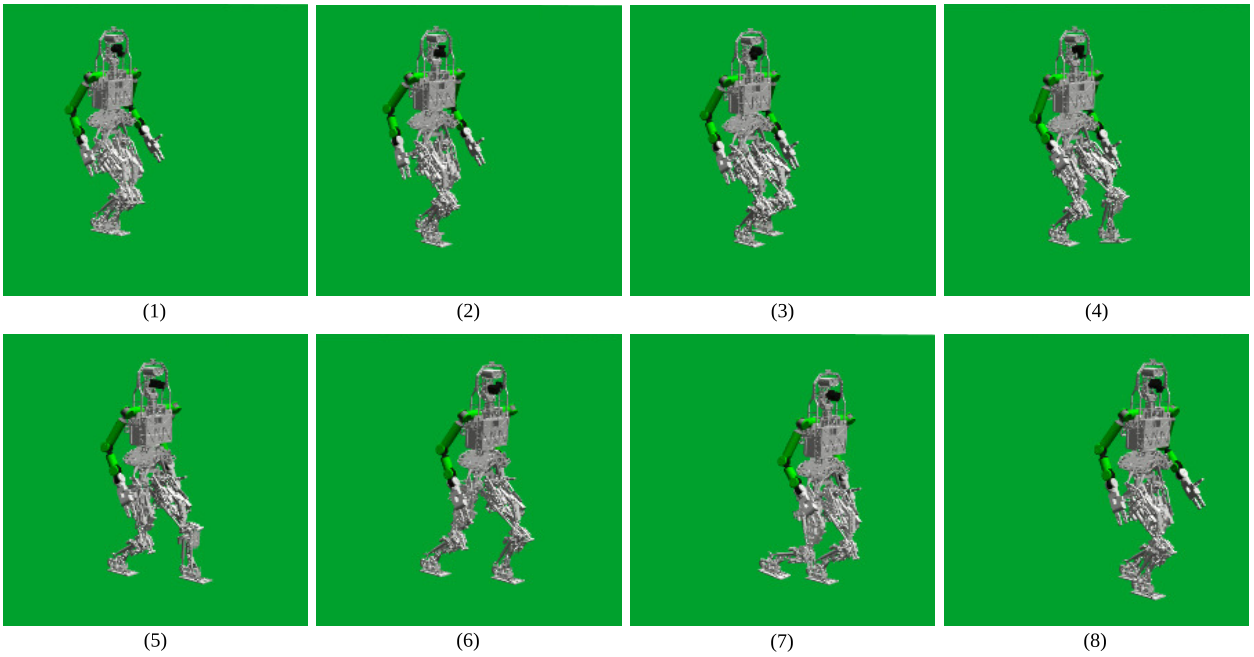


Figure 2.22: Six step plan with a 1.5s step duration, 0.375s double support duration, 270N forward push on the third step. The force is applied at the start of pane (3). Pane (6) is the heel strike of the recovery step. As can be seen, the large forward adjustment results in a lunging motion of the robot.

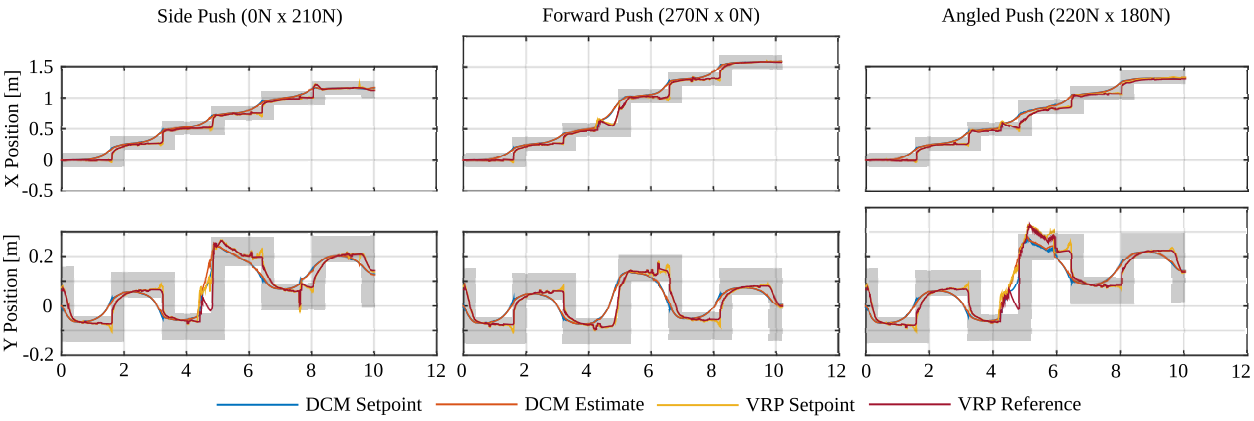


Figure 2.23: Includes the assembled VRP reference trajectory for the three push recovery experiments. It is this desired trajectory that ultimately determines the DCM setpoint trajectory. The gray area represents the current base of support of the robot.

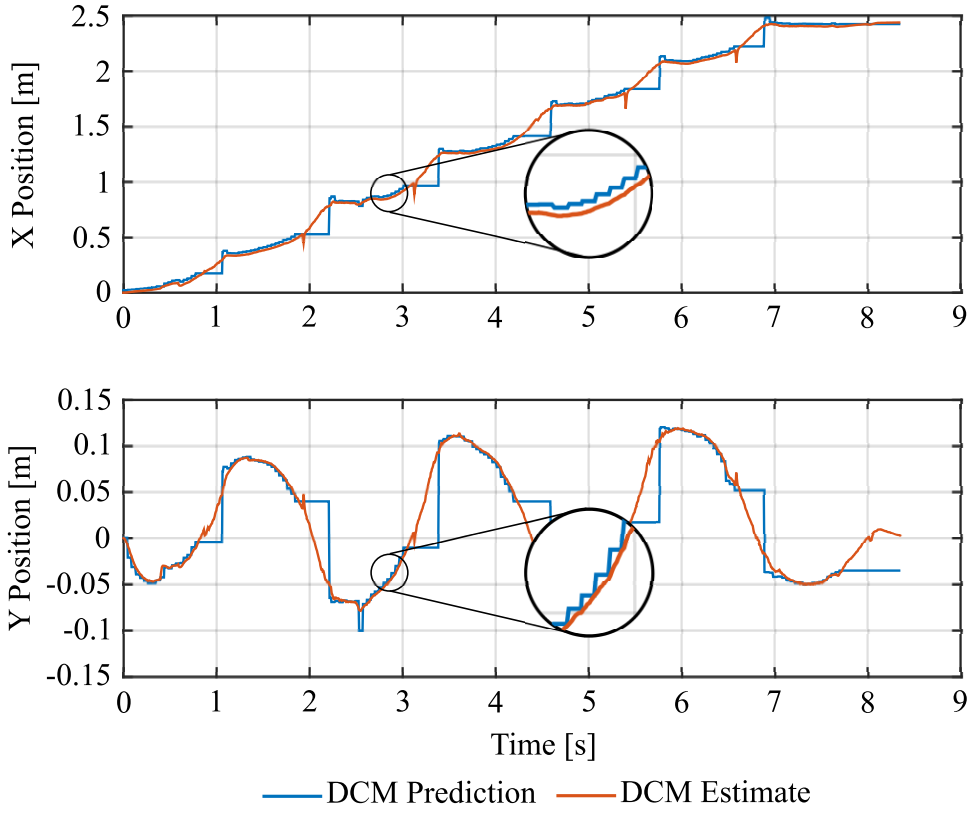


Figure 2.24: Plot of the same simulation as in Figure 2.13. This plot compares what the DCM is predicted to be when the trajectories are updated to the DCM estimate. The flat points in the DCM prediction represent when the MPC is turned off at the end of single support.

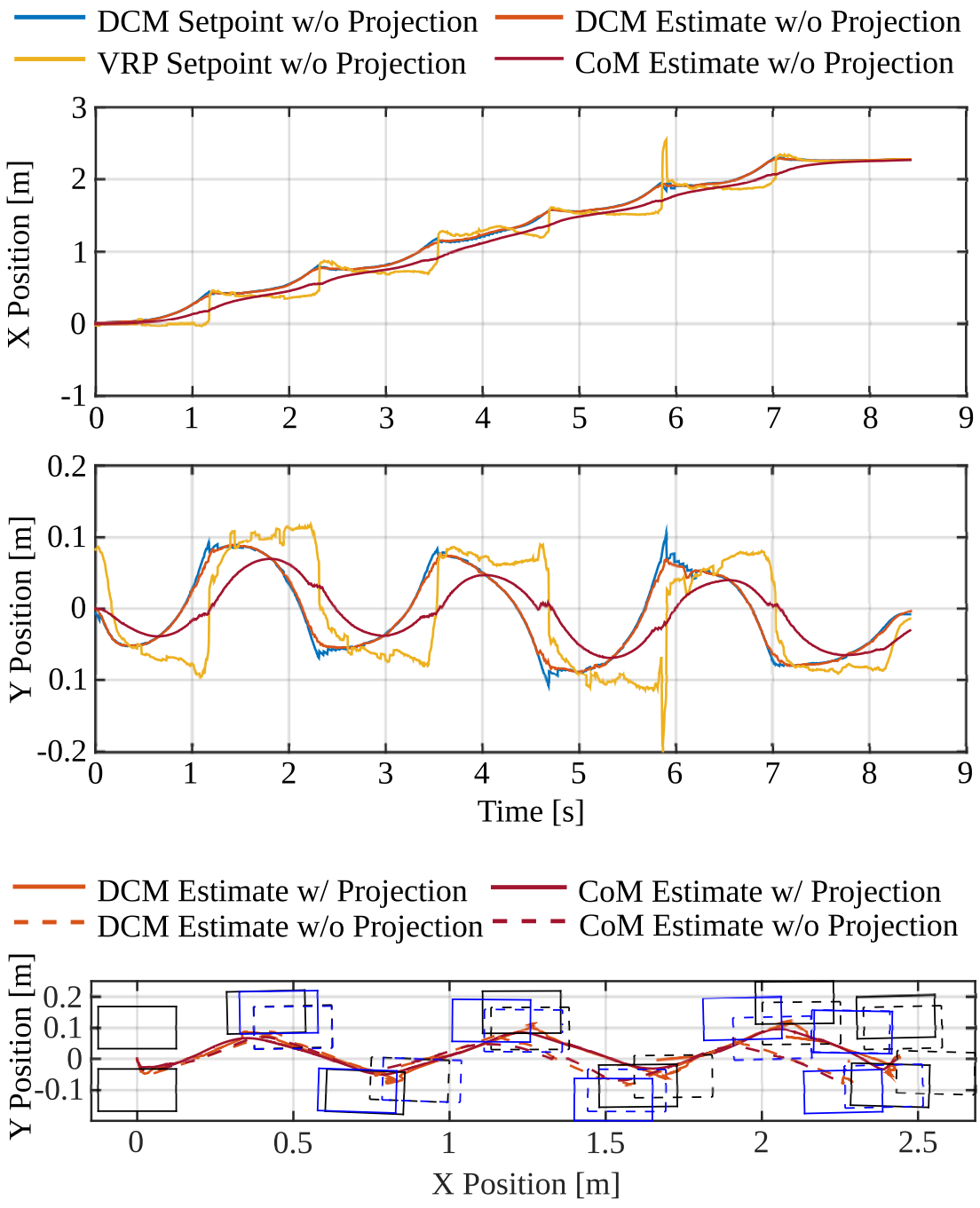


Figure 2.25: Simulation using the same six step plan with $0.5m$ steps with a $1s$ step duration, but with the DCM prediction module turned off. This results in larger penalties on the DCM acceleration, shifting the feet more. The blue feet represent the results with the DCM prediction turned off, while the black feet are the same feet as in Figure 2.13.

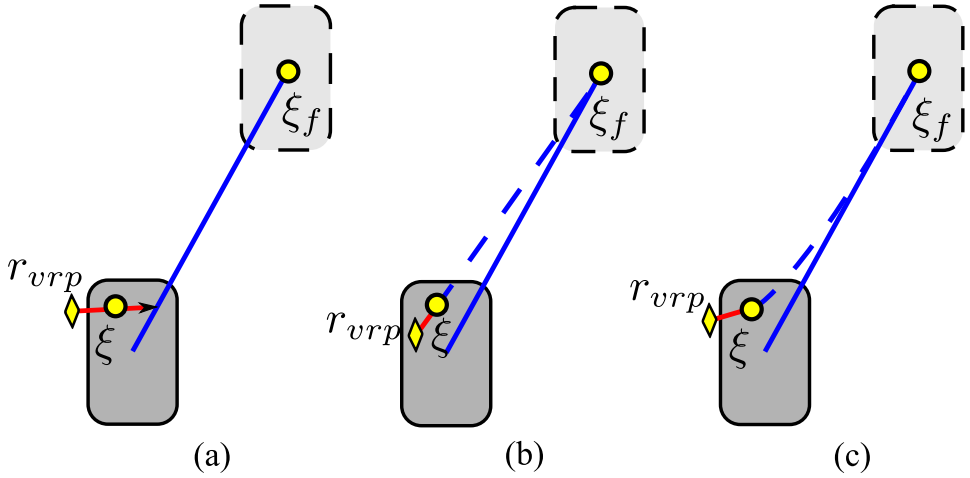


Figure 2.26: Illustration of DCM control algorithms. (a) shows a simple proportional feedback algorithm that tries to drive the DCM back to the reference trajectory, shown as the solid blue line. (b) highlights a MPC-type scheme, where the DCM trajectory is replanned from the current state to the desired final state, shown as the dashed blue line. (c) illustrates an MPC scheme that attempts to return the DCM trajectory to the nominal reference trajectory, a blend of proportional feedback and MPC control schemes.

$$\boldsymbol{\Phi}_0^v = \begin{bmatrix} \prod_{i=1}^{N-1}(A_i) & \prod_{i=2}^{N-1}(A_i) & \cdots & (A_{N-2}A_{N-1}) & A_{N-1} & 1 \end{bmatrix} \tag{2.48}$$

$$\boldsymbol{\Phi}_u^v = \begin{bmatrix} B_1 & A_1B_2 + C_1 & A_1(A_2B_3 + C_2) & \dots & \left(\prod_{i=1}^{N-3} A_i\right)(A_{N-2}C_{N-1} + B_{N-2}) & \left(\prod_{i=1}^{N-2} A_i\right)C_{N-1} \\ 0 & B_2 & A_2B_3 + C_2 & \dots & \left(\prod_{i=2}^{N-3} A_i\right)(A_{N-2}C_{N-1} + B_{N-2}) & \left(\prod_{i=2}^{N-2} A_i\right)C_{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & A_{N-2}B_{N-1} + C_{N-2} & A_{N-2}C_{N-1} \\ 0 & 0 & 0 & \dots & B_{N-1} & C_{N-1} \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \tag{2.49}$$

Chapter 3

Walking Stabilization Using Step Timing, Angular Momentum, and Location Adjustment on the Humanoid Robot, Atlas

While humans are highly capable of recovering from external disturbances and uncertainties that result in large tracking errors, humanoid robots have yet to reliably mimic this level of robustness. Essential to this is the ability to combine traditional "ankle strategy" balancing with step timing and location adjustment techniques. In doing so, the robot is able to step quickly to the necessary location to continue walking. In this work, we present both a new swing speed up algorithm to adjust the step timing, allowing the robot to set the foot down more quickly to recover from errors in the direction of the current capture point dynamics, and a new algorithm to adjust the desired footstep, expanding the base of support to utilize the center of pressure (CoP)-based ankle strategy for balance. We then utilize the desired centroidal moment pivot (CMP) to calculate the momentum rate of change for our inverse-dynamics based whole-body controller. We present simulation and experimental results using this work, and discuss performance limitations and potential improvements.

3.1 Introduction

People are very adept at recovering from large disturbances and uncertainties when walking. Shifting the Center of Pressure (CoP) within the available foothold (the "ankle strategy") is common, as is using angular momentum, by lunging the upper body (the "hip strategy") [121] or windmilling the arms [122]. Angular momentum has its limits, though, and the control authority of the ankle strategy decreases as the walking speed increases and becomes more

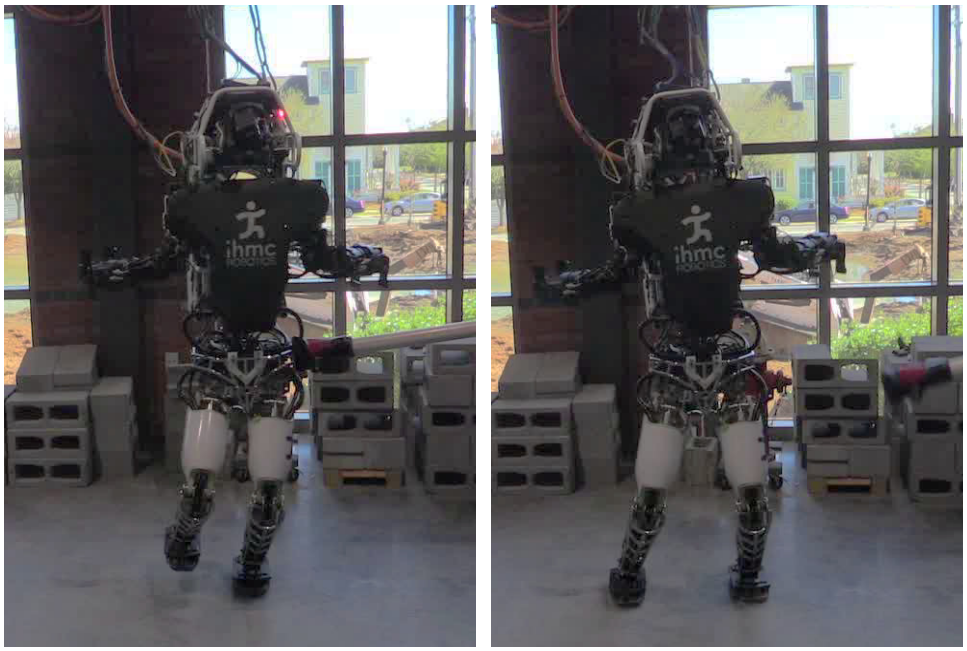


Figure 3.1: Atlas recovering from a lateral push while stepping in place.

dynamic. To handle these limitations, humans quickly adjust their step to the right location and continue walking [123].

Humanoid robots can, in theory, utilize these same approaches, but have yet to match the speed and adaptability of humans. Robots have been demonstrated to be very capable of walking using a set of desired footsteps, stably tracking desired center of mass (CoM) motions, as long as the tracking error does not become too large. This has primarily been performed by controlling either the Zero Moment Point (ZMP), Instantaneous Capture Point (ICP), or Divergent Component of Motion (DCM) with momentum based methods. The Linear Inverted Pendulum Model (LIPM) has been well utilized to generate feasible CoM motions using analytic solutions [13], preview control [22], and Differential Dynamic Program [70], among others. Both the ICP [17] and DCM [18] were introduced by splitting the LIPM dynamics into stable and unstable components, and then controlling only this unstable portion to maintain balance. The LIPM dynamics have then been tracked successfully using momentum-based whole-body control techniques with both traditional feedback controllers [70] and LQR-based methods [30]. ICP and DCM methods have also been used with whole-body controllers to effectively stabilize the walking motion [40, 124, 43]. Due to the limited size of the support polygon, however, these type of tracking controllers are ill equipped to handle large tracking errors, and have very limited effective control authority. While angular momentum has been illustrated as providing additional controllability [125], further improvements are still needed to handle the large tracking errors that may result from external disturbances and uncertainties.

To improve robustness in the face of large errors, several authors have mimicked nature and introduced step adjustment algorithms. Some works have formulated model predictive controllers (MPC) as quadratic programs to achieve this step adjustment [27, 109, 103]. In [27, 109], the step locations are optimized to reject disturbances using the ZMP dynamics while minimizing the CoM jerk to ensure smooth motions. Instead of utilizing the ZMP dynamics, the MPC in [103] is based on the DCM dynamics, but similarly optimizes footstep locations while trying to provide "nice" CoM motions. Alternatively, [115] simply uses the LIPM dynamics to determine the necessary upcoming footstep to return to the desired step plan. This is similar to the work in [126], which integrates the current DCM forward in time to calculate the necessary footstep location to return to the nominal trajectory. While highly efficient, as they are not optimizing full trajectories, neither [115] or [126] consider the combined effects of the ankle strategy with step adjustment.

Instead of adjusting the footstep location, however, the foot can also simply be set down more quickly, another common action employed by humans. However, adjusting the step timing is a challenge, as it tends to result in nonlinearities, and so it is typically viewed as fixed. [127] uses a nonlinear optimization-based pattern generator to find the optimized step positions and step timing given the current CoM state, which are then tracked using a ZMP based feedback controller. [128] augments the earlier work of [109] by allowing the step time to vary, as well, but again utilizes nonlinear optimization to do so. Instead, [129] approximates this nonlinear term as a linear one, allowing the problem to maintain its convexity and efficiency. With the exception of [129], these methods all use non-linear optimization to determine the timing adjustment. We believe that, in addition to using optimization-based approaches, the advantages of timing adjustment can be captured using only the ICP dynamics in a simple algorithm.

In this work, we present two separate timing adjustment algorithms that are highly effective when the ICP tracking error is in the direction of the desired motion, essentially speeding up the dynamic plan in the direction of this error. The first uses only the ICP dynamics to determine the amount of swing speed up. The second relies on a gradient descent approach around a quadratic program to achieve step adjustment. The linearizing approach presented in [129] is not possible in our framework due to the feedback mechanisms that we desire coupling the timing, which would require bilinear constraints. These speed up algorithms then greatly improves the effectiveness of the disturbance rejection with step adjustment, as the robot is able to quickly step to the necessary location for recovery. For step adjustment, instead of using traditional MPC techniques that optimize the entire trajectory, we instead combine the ability to utilize CoP control like in [109] with step adjustment to return to the nominal ICP plan, as in [126]. For this, we present two approaches. The first can be done by observing that the reference ICP trajectory can be formulated as a linear function of the upcoming footstep locations when using ICP planners like [42]. The second is a simpler adjustment algorithm that approximates the desired CMP as a constant, and uses this to compute how much adjustment is required to stabilize the dynamics. Then, by embedding a proportional feedback controller into a quadratic program, the reference trajectory can be

optimized by adjusting the footsteps, taking into account the CoP feedback control action. This makes for a highly efficient algorithm that can be run on robotic hardware in real-time at high frequencies.

3.2 Dynamic Planning and Control

The underlying dynamic planning algorithm utilized on Atlas is based on the ICP, and is fully described in [42]. Note that in [42], the authors utilize the DCM, but, assuming constant height, this is formulaically equivalent in x-y to the ICP. We will summarize this approach in the following paragraphs.

The ICP is a transformation of the CoM state defined as

$$\boldsymbol{\xi} = \mathbf{x} + \frac{1}{\omega_0}\dot{\mathbf{x}}, \tag{3.1}$$

where $\boldsymbol{\xi} = [\xi_x,\ \xi_y]^T$ is the ICP position, $\mathbf{x} = [x,\ y]^T$ and $\dot{\mathbf{x}} = [\dot{x},\ \dot{y}]^T$ are the CoM position and velocity, and $\omega_0 = \sqrt{g/\Delta z_{\text{com}}}$ is the natural frequency of the inverted pendulum. By reordering this, we can see that the CoM has stable first order dynamics with respect to the ICP, meaning that it will converge to the ICP over time. Through differentiation, the ICP dynamics are defined as

$$\dot{\boldsymbol{\xi}} = \omega_0 \left(\boldsymbol{\xi} - \mathbf{r}_{\text{cmp}}\right), \tag{3.2}$$

where we see that the Centroidal Moment Pivot (CMP) point [130], $\mathbf{r}_{\text{cmp}}$, controls the ICP dynamics. From Equation 4.2, the CMP is defined as

$$\mathbf{r}_{\text{cmp}} = \boldsymbol{\xi} - \frac{1}{\omega_0}\dot{\boldsymbol{\xi}}, \tag{3.3}$$

allowing it to be calculated from a given ICP trajectory.

3.2.1 Dynamic Planning

From the definition of the ICP dynamics in Equation 4.2, the linear, first order differential equation has a closed form solution

$$\boldsymbol{\xi}(t) = e^{\omega_0 t}\left(\boldsymbol{\xi}_0 - \mathbf{r}_{\text{cmp}}\right) + \mathbf{r}_{\text{cmp}}, \tag{3.4}$$

assuming $\mathbf{r}_{\text{cmp}}$ is held constant throughout t. Using this equation, we can calculate a desired ICP trajectory for walking, given a set of desired footsteps and desired CMP locations in those footsteps. To more accurately represent human-like walking, we use two CMPs per foot, one in the heel ($\mathbf{r}_{\text{cmp},H}$) and one in the toe ($\mathbf{r}_{\text{cmp},T}$), as shown in Figure 4.2 by the green

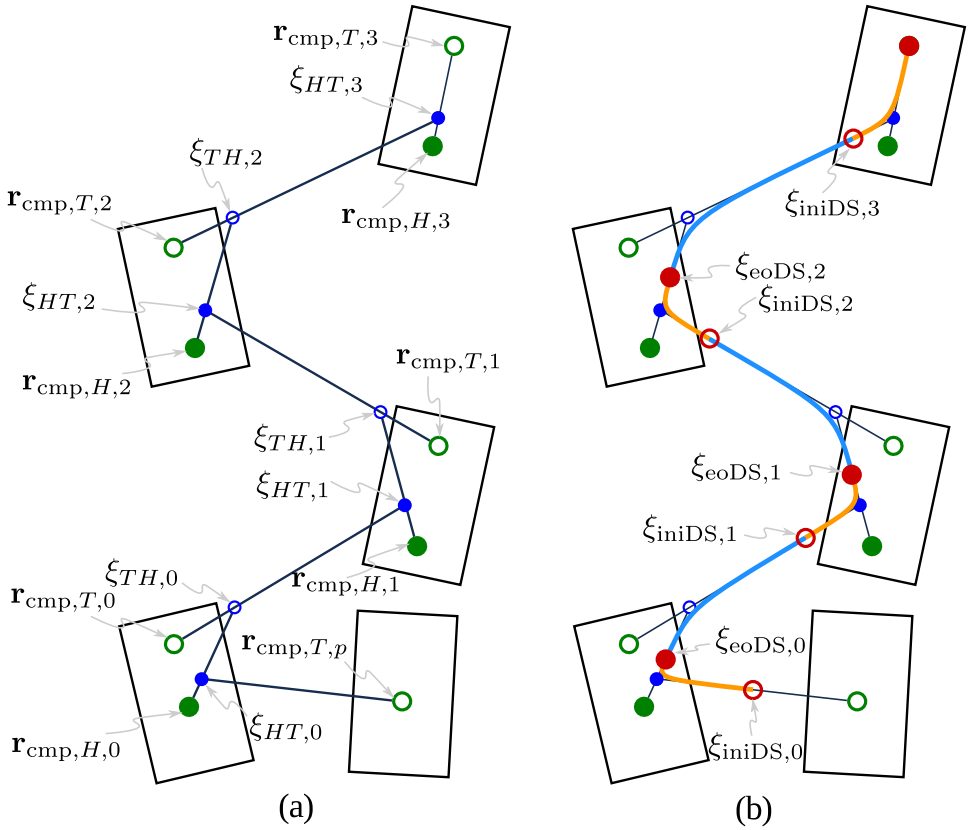


Figure 3.2: Heel-to-Toe ICP trajectory [42].

circles. This results in the reference CMP trajectory moving from the heel to the toe in the foot while stepping.

To determine the desired ICP trajectory, we can recurse backward from the final objective location. This can be done by using the solution to the ICP dynamics in Equation 4.3, and assuming a static CMP location. We can define the time spent on the toe-CMP as a fraction of the full step duration, $T_{TH} = \alpha_{TH} T$ and the corresponding time spent on the heel-CMP as $T_{HT} = (1 - \alpha_{TH})\, T$. Using this, we can calculate the ICP "corner points", $\boldsymbol{\xi}_{TH,i}$ and $\boldsymbol{\xi}_{HT,i}$. This results in the dark blue trajectories in Figure 4.2(a).

To achieve this reference trajectory, however, an instantaneous shift is required from the reference CMP locations, $\mathbf{r}_{\text{cmp},H,i}$ and $\mathbf{r}_{\text{cmp},T,i}$. Instead, we can smooth these trajectories using third order polynomial interpolation, which guarantees smoothness of the CMP trajectory [42]. The general goal during the transfer state is to shift the desired CMP from the previous toe to the upcoming heel. As such, we can define the initial ICP location at the start of double support, $\boldsymbol{\xi}_{\text{iniDS},i}$, and the ICP location at the end of double support, $\boldsymbol{\xi}_{\text{eoDS},i}$,

with respect to the corner point $\boldsymbol{\xi}_{HT,i}$ as

$$\begin{aligned} \boldsymbol{\xi}_{\text{iniDS},i} &= \mathbf{r}_{\text{cmp},T,i-1} + e^{-\omega_0 T_{\text{iniDS}}} \left(\boldsymbol{\xi}_{HT,i} - \mathbf{r}_{\text{cmp},T,i-1} \right), \\ \boldsymbol{\xi}_{\text{eoDS},i} &= \mathbf{r}_{\text{cmp},H,i} + e^{\omega_0 T_{\text{eoDS}}} \left(\boldsymbol{\xi}_{HT,i} - \mathbf{r}_{\text{cmp},H,i} \right). \end{aligned} \tag{3.5}$$

The durations to compute these boundary conditions are defined by $T_{\text{iniDS}} = \alpha_{\text{iniDS}} T_{DS}$ and $T_{\text{eoDS}} = (1 - \alpha_{\text{iniDS}}) T_{DS}$, where T_{DS} is the transfer duration. These knots are shown as the dark red circles in Figure 4.2(b). This spline can then be used to compute the ICP position and velocity as a linear function of the boundary conditions,

$$\boldsymbol{\xi}(t) = \mathbf{C}_{\xi}(t) \boldsymbol{\Xi}_{\text{bnd}}, \;\; \dot{\boldsymbol{\xi}}(t) = \mathbf{C}_{\dot{\xi}}(t) \boldsymbol{\Xi}_{\text{bnd}}. \tag{3.6}$$

Here, the matrices $\mathbf{C}_{\xi}$ and $\mathbf{C}_{\dot{\xi}}$ encode the polynomial values at time t. This results in the light blue and orange colored lines in Figure 4.2(b). The reference CMP trajectory can then be calculated using Equation 3.3.

This approach for ICP planning leads to the trajectories shown in Figure 3.3, which uses $\alpha_{TH} = \alpha_{\text{iniDS}} = 0.5$. As the walking speed is increased, the resulting plans become more dynamic. The blue cross represents the desired ICP location half-way through the swing state. As shown, as the walking speed increases, this ICP location gets further outside of the foot, representing more dynamic walking trajectories.

3.2.2 Control

In our momentum-based control framework, the desired CMP position $\mathbf{r}_{\text{cmp},d}$ is transformed to the desired rate of change of the horizontal linear momentum of the robot by

$$\dot{\mathbf{l}} = m\omega_0^2 \left(\mathbf{x} - \mathbf{r}_{\text{cmp},d} \right). \tag{3.7}$$

This becomes the momentum objective to the whole-body controller described in [124]. $\mathbf{r}_{\text{cmp},d}$ can be calculated using a simple proportional feedback law [125],

$$\mathbf{r}_{\text{cmp},d} = \boldsymbol{\xi} - \frac{1}{\omega_0} \dot{\boldsymbol{\xi}}_r + \mathbf{k}_p \left(\boldsymbol{\xi} - \boldsymbol{\xi}_r \right), \tag{3.8}$$

where $\boldsymbol{\xi}$ is the measured ICP location. Inserting the ICP dynamics from Equation 4.2 into Equation 3.8 yields

$$\mathbf{r}_{\text{cmp},d} = \mathbf{k}_{\xi} \left(\boldsymbol{\xi} - \boldsymbol{\xi}_r \right) + \mathbf{r}_{\text{cmp},r}, \tag{3.9}$$

where $\mathbf{k}_{\xi} = \mathbf{k}_p + \mathbf{1}$, showing that the controller simply adjusts the CMP proportional to the current ICP error.

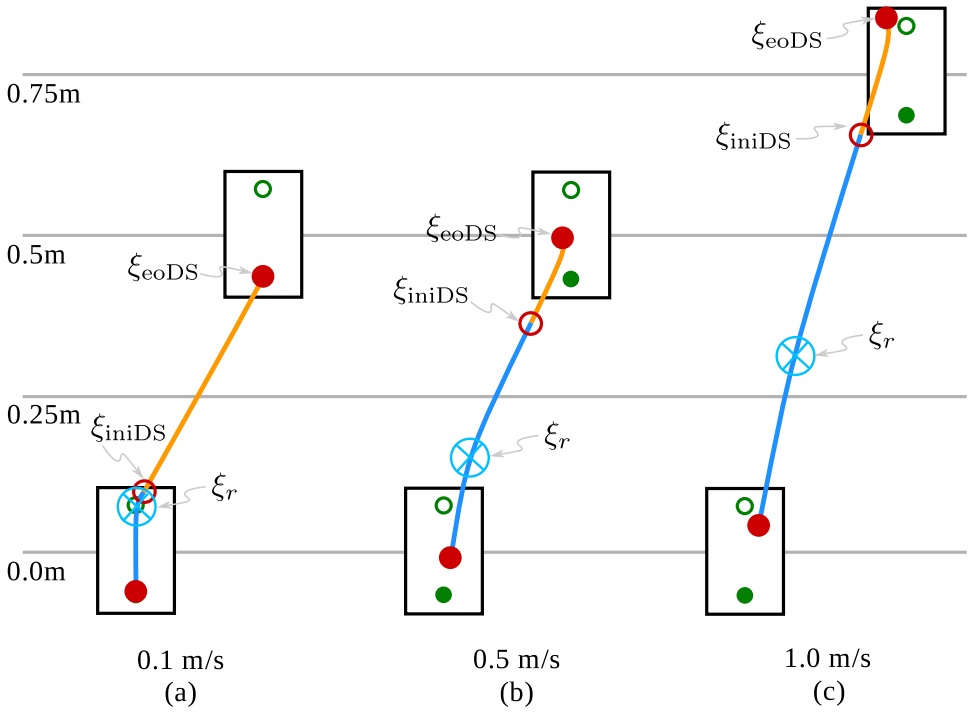


Figure 3.3: Diagram showing step plans at different walking speeds. The light blue lines represent the ICP trajectory during swing, while the orange lines are during transfer.

3.3 Dynamics Based Swing Speed Up

While in an ideal scenario, humanoid robots do not experience any tracking errors when walking, this is, unfortunately, almost never the case. Any combination of circumstances can combine to induce these errors, from joint stiction to inaccurate dynamic models to external disturbances. Most commonly, some form of proportional feedback controller, as in Equation 3.8, is employed to correct for this tracking error. This results in applying additional corrective forces to drive the ICP back to the desired path.

An alternative to providing corrective forces during swing is to adjust the timing of the step. This is a technique commonly utilized by people; when pushed, we will rapidly put our foot down to recover, in addition to or in place of adjusting the step. If the error occurs along the current ICP trajectory, this then requires no corrective forces at all, instead only setting the foot down. Additionally, when combined with step adjustment strategies, step timing can be very effective for assisting in rejecting significant ICP tracking errors. Due to the exponential relationship between the ICP dynamics and the step time, as shown in Equation 4.3, the required step adjustment to recover from tracking errors increases exponentially as the swing time increases. This means that the inverse also holds: decreasing the swing time exponentially decreases the required step adjustment.

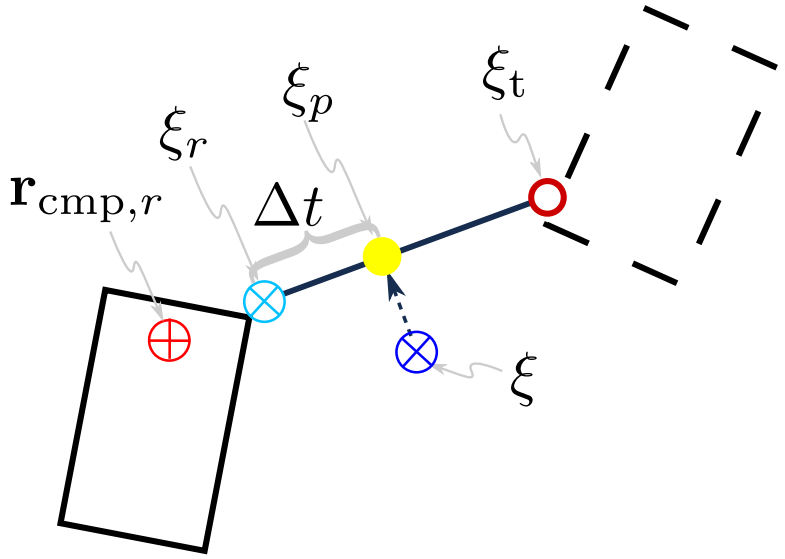


Figure 3.4: Illustration of proposed swing speed up calculation.

We would like to find a time advancement, Δt, then, such that, at $t^+ = t + \Delta t$, the reference ICP, $\boldsymbol{\xi}_p$, is as close to the estimated ICP as possible. From the definition of the ICP dynamics, this value lies on the vector $\boldsymbol{\xi}_\text{t} - \boldsymbol{\xi}_r$, where $\boldsymbol{\xi}_\text{t}$ is the final ICP location at touchdown. This is an accurate description of the ICP dynamics, assuming that the location of $\mathbf{r}_{\text{cmp},r}$ does not change during swing; a valid assumption given appropriate planning. $\boldsymbol{\xi}$ can be projected onto this vector to find $\boldsymbol{\xi}_p$ by

$$\boldsymbol{\xi}_p = \boldsymbol{\xi}_r + \frac{(\boldsymbol{\xi} - \boldsymbol{\xi}_r)^T (\boldsymbol{\xi}_\text{t} - \boldsymbol{\xi}_r)}{\|\boldsymbol{\xi}_\text{t} - \boldsymbol{\xi}_r\|}, \tag{3.10}$$

as shown in Figure 3.4.

From $\boldsymbol{\xi}_p$, we can calculate how much further ahead in time that point is using Equation 4.3, setting the projected ICP as the end condition,

$$\boldsymbol{\xi}_p = e^{\omega_0 \Delta t} (\boldsymbol{\xi}_r - \mathbf{r}_{\text{cmp},r}) + \mathbf{r}_{\text{cmp},r}. \tag{3.11}$$

From here, Δt can be solved for by

$$\Delta t = \frac{1}{\omega} \log_e \left(\frac{\boldsymbol{\xi}_p - \mathbf{r}_{\text{cmp},r}}{\boldsymbol{\xi}_r - \mathbf{r}_{\text{cmp},r}} \right). \tag{3.12}$$

The ICP plan is then advanced to the new time, t^+. To track the swing foot trajectory, however, instead of advancing the time, we calculate a speed up factor σ that will cause the

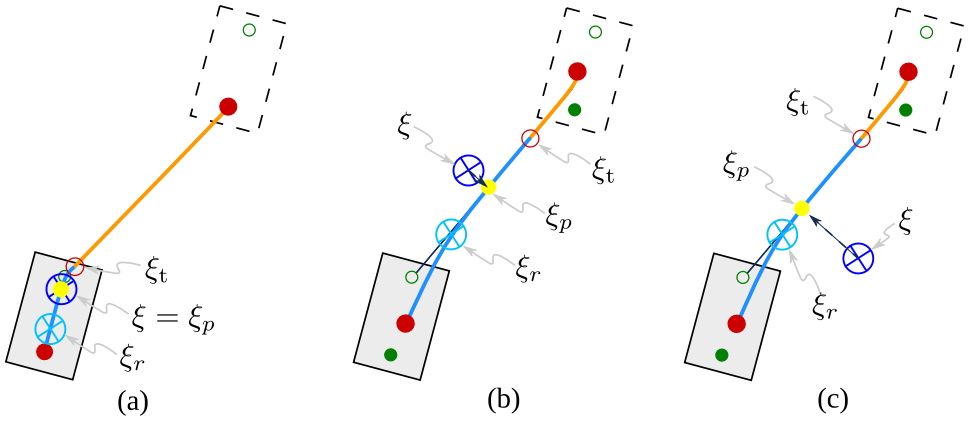


Figure 3.5: Speeding up the plan can be very effective when the error is in the direction of the dynamics, as in (a) and (b), but not when it is perpendicular to this motion, as in (c).

remaining duration to pass more quickly. σ can be calculated using Δt as

$$\sigma = \frac{T_{SS} - t}{T_{SS} - t^+} \tag{3.13}$$

where T_{SS} is the desired swing time. This approach prevents discontinuities in the desired position for the swing foot.

This control technique is very effective for compensating for errors in the direction of the desired motion, such as being pushed from behind while walking forward, as shown in Figure 3.5. If the robot is taking slower steps, as in Figure 3.5(a), some tracking error purely in the x direction is still on the ICP plan, requiring no corrective forces. If we take faster steps, as in Figure 3.5(b), significant forward error still results in relatively small tracking errors once the plan is sped up. However, this speed up approach is not very effective when the errors are perpendicular to the stepping motion (Figure 3.5(c)). Here, the tracking error is only marginally reduced by projecting the ICP onto the plan, requiring either significant corrective forces or step adjustment to compensate.

3.4 Step Adjustment

The main objective of the step adjustment algorithm is to combine a proportional feedback controller with one that can adjust the upcoming footsteps. As we showed in **??**, speeding up the swing is only effective for errors in the directions of the desired ICP dynamics. When these errors are perpendicular to the dynamics, CMP-based control must be used to try and return the ICP to the reference trajectory. The control authority granted by moving this value is limited, though, which is equivalent to saying that some tracking errors are too great to return to the nominal plan. In this case, the only remaining action is to adjust

the upcoming footsteps, allowing the footstep to be moved in the direction of the current ICP dynamics. When combined with the ability speed up the swing plan, this becomes particularly effective, allowing the robot to step quickly to the necessary location to return to the nominal walking plan after N steps.

In this section, we present two separate step adjustment strategies. The first utilizes the knowledge that the ICP plan can be formulated as a linear function of the upcoming step positions. We can then effectively invert the planner to compute the necessary step positions based on the current dynamic state. We will refer to this as the "plan inversion" step adjustment. The second is slightly less accurate, as it assumes a constant CMP location. This, however, is less restrictive, as it is plan independent, only relying on the current desired ICP and estimated ICP, as well as the desired CMP location. This allows it to utilize nonlinear plans, if desired, and encodes any plan modifications that can be made on-the-fly. We will refer to this as the "simple" step adjustment.

3.4.1 Plan Inversion Step Adjustment Recursive Dynamics

Given a footstep plan, we can define N steps to consider for adjustment. We can then define the first static ICP corner point in the plan, $\boldsymbol{\xi}_{HT,N+1}$, as $\boldsymbol{\xi}_{\mathrm{f}}$, from which the local reference value will be defined. Based on Equation 4.3, we can see that the ICP corner points are simply linear functions of $\boldsymbol{\xi}_{\mathrm{f}}$ and the N heel and toe CMP locations. This is formally defined by

$$\boldsymbol{\xi}_{\mathrm{eo}} = \gamma_{\mathrm{f}}\boldsymbol{\xi}_{\mathrm{f}} + \textstyle\sum_{i=0}^{N} \left(\gamma_{T,i}\mathbf{r}_{\mathrm{cmp},T,i} + \gamma_{H,i}\mathbf{r}_{\mathrm{cmp},H,i}\right), \tag{3.14}$$

where $\boldsymbol{\xi}_{\mathrm{eo}}$ is $\boldsymbol{\xi}_{TH,0}$ if the robot is currently in the swing state and $\boldsymbol{\xi}_{HT,0}$ if in transfer. The scalar multipliers $\gamma_{\mathrm{f}}, \gamma_{T,i}$, and $\gamma_{H,i}$ are computed in Algorithm 1. If we observe that the CMP locations can be defined relative to footstep positions by

$$\mathbf{r}_{\mathrm{cmp},T,i} = \mathbf{r}_{\mathrm{off},T,i} + \mathbf{r}_{f,i}, \quad \mathbf{r}_{\mathrm{cmp},H,i} = \mathbf{r}_{\mathrm{off},H,i} + \mathbf{r}_{f,i}. \tag{3.15}$$

Equation 3.14 can then be rewritten as a linear function of the step positions,

$$\begin{aligned} \boldsymbol{\xi}_{\mathrm{eo}} =& \gamma_{\mathrm{f}}\boldsymbol{\xi}_{\mathrm{f}} + \boldsymbol{\Xi}_{\mathrm{off}} + \gamma_{T,0}\mathbf{r}_{\mathrm{cmp},T,0} + \gamma_{H,0}\mathbf{r}_{\mathrm{cmp},H,0} \\ &+ \textstyle\sum_{i=1}^{N} \left(\gamma_{T,i} + \gamma_{H,i}\right)\mathbf{r}_{f,i}, \end{aligned} \tag{3.16}$$

where

$$\boldsymbol{\Xi}_{\mathrm{off}} = \textstyle\sum_{i=1}^{N} \left(\gamma_{T,i}\mathbf{r}_{\mathrm{off},T,i} + \gamma_{H,i}\mathbf{r}_{\mathrm{off},H,i}\right).$$

We can then define the boundary conditions for the splines in transfer and swing for Equation 3.6. Using $\boldsymbol{\xi}_{\mathrm{eo}}$ from Equation 3.16, we can define $\boldsymbol{\Xi}_{\mathrm{bnd}}$ as

$$\boldsymbol{\Xi}_{\mathrm{bnd}} = \mathbf{A}_{\mathrm{F}}\boldsymbol{\xi}_{\mathrm{eo}} + \mathbf{B}_{T,0}\mathbf{r}_{\mathrm{cmp},T,0} + \mathbf{B}_{H,0}\mathbf{r}_{\mathrm{cmp},H,0}, \tag{3.17}$$

Algorithm 1 Recursive multipliers

if Single-Support **then**
  $\boldsymbol{\xi}_{\text{eo}} = \boldsymbol{\xi}_{TH,0}$;
  $\gamma_{T,0} = 1 - e^{-\omega_0 T_{TH,0}}$;
  $\gamma_{H,0} = 0$;
  $\gamma_{\text{f}} = e^{-\omega_0 \left(T_{TH,0} + \sum_{i=1}^{N} T_i\right)}$;
  for i = 1,N **do**
    $\gamma_{T,i} = e^{-\omega_0 \left(T_{TH,0} + T_{HT,i} + \sum_{j=1}^{i-1} T_j\right)} \left(1 - e^{-\omega_0 T_{TH,i}}\right)$;
    $\gamma_{H,i} = e^{-\omega_0 \left(T_{TH,0} + \sum_{j=1}^{i-1} T_j\right)} \left(1 - e^{-\omega_0 T_{HT,i}}\right)$;
  end for
else
  $\boldsymbol{\xi}_{\text{eo}} = \boldsymbol{\xi}_{HT,0}$;
  $\gamma_{T,0} = e^{-\omega_0 T_{HT,0}} \left(1 - e^{-\omega_0 T_{TH,0}}\right)$;
  $\gamma_{H,0} = 1 - e^{-\omega_0 T_{HT,0}}$;
  $\gamma_{\text{f}} = e^{-\omega_0 \sum_{i=0}^{N} T_i}$;
  for i = 1,N **do**
    $\gamma_{T,i} = e^{-\omega_0 \left(T_{HT,i} + \sum_{j=0}^{i-1} T_j\right)} \left(1 - e^{-\omega_0 T_{TH,i}}\right)$;
    $\gamma_{H,i} = e^{-\omega_0 \sum_{j=0}^{i-1} T_j} \left(1 - e^{-\omega_0 T_{HT,i}}\right)$;
  end for
end if

where $\mathbf{A}_{\mathrm{F}}, \mathbf{B}_{T,0}$, and $\mathbf{B}_{H,0}$ are calculate the boundary conditions from the corner points using the ICP dynamics.

Combining Equation 3.16 and Equation 3.17 yields $\boldsymbol{\xi}_r$ as a linear function of the step positions,

$$\boldsymbol{\xi}_r = \boldsymbol{\Phi}_F \boldsymbol{\xi}_{\mathrm{f}} + \textstyle\sum_{i=1}^{N} \boldsymbol{\Gamma}_i \mathbf{r}_{f,i} + \boldsymbol{\Phi}_{\mathrm{cnst}}, \tag{3.18}$$

where

$$\begin{aligned}
\boldsymbol{\Phi}_F =& \gamma_{\mathrm{f}} \mathbf{C}_{\xi}(t^+) \mathbf{A}_{\mathrm{F}}, \\
\boldsymbol{\Gamma}_i =& \left(\gamma_{T,i} + \gamma_{H,i}\right) \mathbf{C}_{\xi}(t^+) \mathbf{A}_{\mathrm{F}}, \\
\boldsymbol{\Phi}_{\mathrm{cnst}} =& \mathbf{C}_{\xi}(t^+) \left(\mathbf{A}_{\mathrm{F}} \boldsymbol{\Xi}_{\mathrm{off}} + \left(\mathbf{B}_{T,0} + \gamma_{T,0} \mathbf{A}_{\mathrm{F}}\right) \mathbf{r}_{\mathrm{cmp},T,0} \right. \\
& \left. + \left(\mathbf{B}_{H,0} + \gamma_{H,0} \mathbf{A}_{\mathrm{F}}\right) \mathbf{r}_{\mathrm{cmp},H,0}\right).
\end{aligned}$$

We can reformulate this as a standard linear function,

$$\boldsymbol{\xi}_r = \mathbf{A} \mathbf{r}_{f,i} + \mathbf{b}, \tag{3.19}$$

where

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{\Gamma}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Gamma}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \dots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\Gamma}_i \end{bmatrix}, \tag{3.20}$$

and

$$\mathbf{b} = \boldsymbol{\Phi}_F \boldsymbol{\xi}_{\mathrm{f}} + \boldsymbol{\Phi}_{\mathrm{cnst}}. \tag{3.21}$$

This formulation will be used later to determine the reference ICP location.

3.4.2 Simple Step Adjustment Recursive Dynamics

An alternative to using the plan inversion approach relies on some fundamental ICP dynamics. Fundamentally, we can estimate where the ICP location will be at the end of the current state, and compare that to the desired ICP location at the end of the step. We can then use this difference to define the desired footstep adjustment. This also allows the incorporation of a simple "safety factor" that scales the desired step adjustment.

We can predict the reference ICP location at the end of single support given the current reference value using the closed form ICP dynamics as,

$$\boldsymbol{\xi}_{f,SS} = e^{\omega T_r} \boldsymbol{\xi}_r + \left(1 - e^{\omega T_r}\right) \mathbf{r}_{\mathrm{cmp},r}, \tag{3.22}$$

where T_r is the time remaining in swing. During transfer, however, there is a phase where the ICP is somewhat at the exit ICP of the current plan. We can approximate this using the same notation as [42] as $T_{DS,exit} = \alpha_{DS} T_{DS}$. Using the notation from [42], we can use

this to define the corner point, which will be $\boldsymbol{\xi}_f$,

$$\boldsymbol{\xi}_f = e^{\omega(T_r+\alpha_{DS}T_{DS})}\boldsymbol{\xi}_r + \left(1 - e^{\omega(T_r+\alpha_{DS}T_{DS})}\right)\mathbf{r}_{cmp,r}. \tag{3.23}$$

If the upcoming footstep is adjusted, this is effectively adjusting this corner point. We will refer to this adjusted point as $\boldsymbol{\xi}_f^*$, which can be defined by

$$\boldsymbol{\xi}_f^* = \boldsymbol{\xi}_f + (\mathbf{x}_f - \mathbf{x}_{f,r}). \tag{3.24}$$

Using this modified desired endpoint, we can recurse backward in time using the same procedure to find the modified reference ICP location,

$$\boldsymbol{\xi}_r^* = e^{-\omega(T_r+\alpha_{DS}T_{DS})}\boldsymbol{\xi}_f^* + \left(1 - e^{-\omega(T_r+\alpha_{DS}T_{DS})}\right)\mathbf{r}_{cmp,r}. \tag{3.25}$$

By inserting Equation 3.23 and Equation 3.24 into Equation 3.25, we obtain

$$\boldsymbol{\xi}_r^* = \boldsymbol{\xi}_r + e^{-\omega(T_r+\alpha_{DS}T_{DS})}(\mathbf{x}_f - \mathbf{x}_{f,r}). \tag{3.26}$$

Putting this into standard linear system form yields

$$\boldsymbol{\xi}_r^* = \gamma\mathbf{x}_f + \boldsymbol{\xi}_r - \gamma\mathbf{x}_{f,r}, \tag{3.27}$$

or

$$\boldsymbol{\xi}_r^* = \mathbf{A}\mathbf{x}_f + \mathbf{b}, \tag{3.28}$$

where

$$\begin{aligned} \gamma &= e^{-\omega(T_r+\alpha_{DS}T_{DS})} \\ \mathbf{A} &= \gamma \\ \mathbf{b} &= \boldsymbol{\xi}_r - \gamma\mathbf{x}_{f,r}. \end{aligned} \tag{3.29}$$

This approach is illustrated in Figure 3.6.

As the reference CMP location is not typically held constant during swing, this does introduce some error in the predicted final ICP location. This, however, only imposes some error in our value for γ. Since the CMP when walking is moving from the heel to the ball of the foot, it is moving in the direction of the ICP dynamics. This means that the estimated footstep adjustment will be slightly greater than what is strictly necessary to achieve stability. This is acceptable, as it is simply expanding the support polygon more than needed. In practice, we can modify γ by a safety factor, forcing the adjustment to be either more or less than necessary.

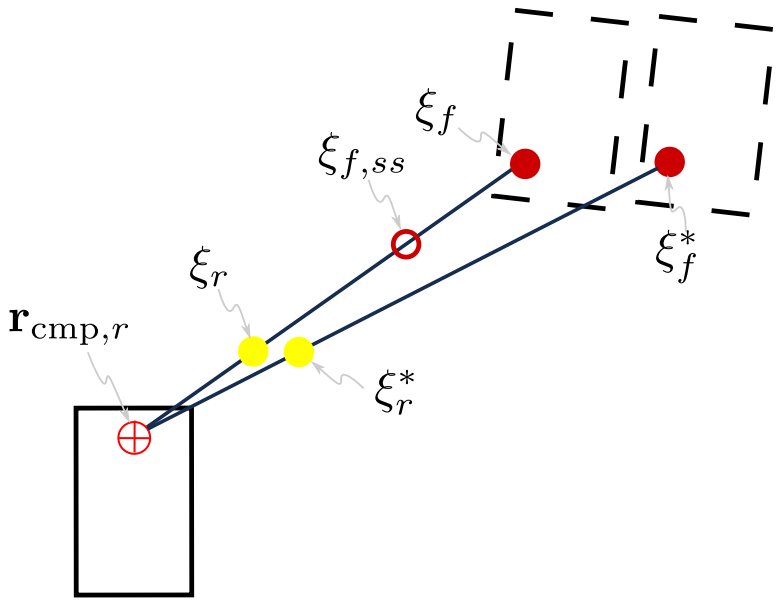


Figure 3.6: Outline of the simple ICP adjustment algorithm. As can be seen, the adjusted reference ICP location is a function of the current desired ICP location, the current reference CMP location, and the amount of footstep adjustment.

3.4.3 Feedback Controller

The standard feedback controller in Equation 3.9 can be rearranged to yield the corrective CMP action,

$$\mathbf{r}_{\text{cmp},d} - \mathbf{r}_{\text{cmp},r} = \mathbf{k}_\xi \left(\boldsymbol{\xi} - \boldsymbol{\xi}_r \right). \tag{3.30}$$

Through this definition, we do not gain any insight into the amount of angular momentum used for balance. Angular momentum has been shown to be a powerful stabilizing tool [45], but is typically not utilized, choosing instead to constrain the CMP to the base of support. Instead, we can rewrite our feedback controller as

$$\left(\boldsymbol{\kappa} + \mathbf{r}_{\text{cop},d} \right) - \mathbf{r}_{\text{cmp},r} = \mathbf{k}_\xi \left(\boldsymbol{\xi} - \boldsymbol{\xi}_r \right). \tag{3.31}$$

In this case, by noting the definition of the CMP as

$$\mathbf{r}_{cmp} = \mathbf{r}_{cop} + \frac{1}{m\left(g + \ddot{z}\right)} \left[-\tau_y,\ \tau_x,\ 0 \right]^T, \tag{3.32}$$

we see that the term $\boldsymbol{\kappa}$ encodes the amount of angular momentum. We can introduce a new variable, $\boldsymbol{\delta} = \mathbf{r}_{cop,d} - \mathbf{r}_{cmp,r}$, such that

$$\boldsymbol{\kappa} + \boldsymbol{\delta} = \mathbf{k}_\xi \left(\boldsymbol{\xi} - \boldsymbol{\xi}_r \right), \tag{3.33}$$

where $\boldsymbol{\delta} + \boldsymbol{\kappa}$ encodes the amount of corrective forces the robot exerts to try to return to the nominal plan. By inserting Equation 3.18, we can see that the feedback action is a function of the current state of the robot $\boldsymbol{\xi}$, the current time t^+, and the upcoming footsteps, $\mathbf{r}_{f,i}$.

3.4.4 Objective Function

Using the definition of the feedback controller, we can define a quadratic program (QP) that optimizes between using feedback control and footstep adjustment, which can be written as

$$\begin{aligned} &\min && J\left(\mathbf{r}_{f,i}, \boldsymbol{\delta}, \boldsymbol{\kappa}, \boldsymbol{\eta}\right) \\ &\text{subject to} && \boldsymbol{\delta} + \boldsymbol{\kappa} = \mathbf{k}_{\xi}\left(\boldsymbol{\xi} - \mathbf{A}\mathbf{r}_{f} - \mathbf{b} - \eta\right), \end{aligned} \tag{3.34}$$

where J is the total quadratic cost function. We can define J as the sum of several quadratic cost functions, each representing its own motion task,

$$J = J_{\text{foot}} + J_{\text{foot,reg}} + J_{\delta} + J_{\delta,\text{reg}} + J_{\text{ang}} + J_{\text{relax}}. \tag{3.35}$$

We can define these costs as

$$J_{\text{foot}} = \sum_{i=1}^{N} \left\|\mathbf{r}_{f,i} - \mathbf{r}_{f,r,i}\right\|_{\mathbf{Q}_{f,i}}^{2}, \tag{3.36}$$

where $\mathbf{Q}_{f,i}$ in J_{foot} is a positive definite weighting matrix that penalizes deviations of the footstep location from desired footstep location,

$$J_{\delta} = \left\|\boldsymbol{\delta}\right\|_{\mathbf{R}_{\delta}}^{2}, \tag{3.37}$$

where $\mathbf{R}_{\delta}$ in J_{δ} is a positive definite weighting matrix that penalizes deviations of the CoP from the nominal value, effectively penalizing the use of corrective forces,

$$J_{\text{ang}} = \left\|\boldsymbol{\kappa}\right\|_{\mathbf{R}_{\kappa}}^{2}, \tag{3.38}$$

where $\mathbf{R}_{\kappa}$ in J_{ang} is a positive definite weighting matrix that penalizes deviations of the feedback CMP from the feedback CoP penalizing the use of angular momentum. $\boldsymbol{\eta}$ is a slack variable introduced to the dynamics to guard against over constraining the problem, and is minimized through the quadratic cost function

$$J_{\text{relax}} = \left\|\boldsymbol{\eta}\right\|_{\mathbf{Q}_{\eta}}^{2}, \tag{3.39}$$

where $\mathbf{Q}_\eta$ is a sufficiently high weight matrix, causing η to only occur sparingly. We also add two regularization cost functions,

$$J_{\text{foot,reg}} = \sum_{i=1}^{N} \|\mathbf{r}_{f,i} - \mathbf{r}_{f,p,i}\|^2_{\mathbf{Q}_{f,p,i}}, \tag{3.40}$$

and

$$J_{\delta,\text{reg}} = \|\boldsymbol{\delta} - \boldsymbol{\delta}_p\|^2_{\mathbf{R}_{\delta,p}}, \tag{3.41}$$

where $\mathbf{Q}_{f,p,i}$ in $J_{\text{foot,reg}}$ regularizes the footstep location solution and $\mathbf{R}_{\delta,p}$ in $J_{\delta,p}$ regularizes the feedback location solution in the optimization by penalizing deviations from the previous solutions. These costs are added to improve the numerical stability of the solver.

This controller can be seen to allow the two fundamentally different types of walking to emerge. If we require that $\mathbf{r}_{f,i} = \mathbf{r}_{f,r,i}$, the robot can no longer adjust its feet, and walks purely by controlling the ICP with the CMP, as with a standard proportional feedback controller. If $\boldsymbol{\delta}$ is constrained to equal zero, no correct forces are allowed, and the robot is only allowed to balance through step adjustment, similar to walking with only point feet and a point mass.

In practice, through proper tuning, we can ensure that the robot utilizes its full control authority with the CMP before adjusting the footsteps by setting $\mathbf{Q}_{f,i}$ much greater than $\mathbf{R}$. The required footstep adjustment has an exponential relation with the tracking error, but only a linear one with $\boldsymbol{\delta}$. As such, with proper weighting, increasing $\boldsymbol{\delta}$ incurs much lower costs than adjusting the footstep. However, $\boldsymbol{\delta}$ has limits, which we impose through constraints on the QP in the following section. This leads to the robot adjusting the footsteps only when absolutely necessary.

3.4.5 Problem Constraints

Feedback Support Polygon Constraints

Based on physical constraints, the CoP is not allowed to exit the support polygon. While the CMP is, theoretically, allowed to exit the support polygon through the generation of angular momentum, in practice, this should be used sparingly. The amount of angular momentum that can be generated is limited, and it must always be "paid back" by removing it from the system. As such, we can place one of two types of constraints on the system: first, constrain the CoP to be within the support polygon, or, second, constrain the CMP to be within the support polygon. This can be done by defining a series of convex inequality constraints, either

$$\mathbf{r}_{cop,d} = \mathbf{r}_{cmp,r} + \boldsymbol{\delta} \in \mathcal{S}_{\text{support}} \rightarrow \mathbf{A}_{\text{support}} \left(\mathbf{r}_{cmp,r} + \boldsymbol{\delta}\right) \leq \mathbf{b}_{\text{support}}, \tag{3.42}$$

or, if constraining the CMP to lie within the base support, allowing no angular momentum,

$$\mathbf{r}_{cmp,d} = \mathbf{r}_{cmp,r} + \boldsymbol{\delta} + \boldsymbol{\kappa} \in \mathcal{S}_{\text{support}} \rightarrow \mathbf{A}_{\text{support}} \left(\mathbf{r}_{cmp,r} + \boldsymbol{\delta} + \boldsymbol{\kappa}\right) \leq \mathbf{b}_{\text{support}}. \tag{3.43}$$

In practice, it may be desirable to constrain the CoP to lie some distance inside the edge of the support polygon. To do so, we can shrink the support polygon slight, making the CoP constraint

$$\mathbf{r}_{cop,d} = \mathbf{r}_{cmp,r} + \boldsymbol{\delta} \in \mathcal{S}^{-}_{\text{support}} \rightarrow \mathbf{A}^{-}_{\text{support}} \left(\mathbf{r}_{cmp,r} + \boldsymbol{\delta}\right) \leq \mathbf{b}^{-}_{\text{support}}. \tag{3.44}$$

We can also place constraints on the maximum amount of distance outside the support polygon the CMP should be allowed to move. This prevents too much angular momentum rate of change from being requested. Without this, rapid movements of the torso may result, much more than can be handled safely by the robot. This means expanding the support polygon, making the CMP constraint

$$\mathbf{r}_{cmp,d} = \mathbf{r}_{cmp,r} + \boldsymbol{\delta} + \boldsymbol{\kappa} \in \mathcal{S}^{+}_{\text{support}} \rightarrow \mathbf{A}^{+}_{\text{support}} \left(\mathbf{r}_{cmp,r} + \boldsymbol{\delta} + \boldsymbol{\kappa}\right) \leq \mathbf{b}^{+}_{\text{support}}. \tag{3.45}$$

Footstep Constraints

Additional constraints can be placed on the footstep locations, as long as they are of convex form

$$\mathbf{A}_{r,i}\mathbf{r}_{f,i} \leq \mathbf{b}_{r,i}, \forall i. \tag{3.46}$$

In this work, we used this to define a simple rectangular reachability constraint for the robot. We form a simple square, with a minimum step width, maximum step width, maximum forward step, and maximum backward step. More complicated convex shapes may be desired, though. For example, the robot cannot take its longest step when also taking its widest step.

This formulation can also be used to constrain the footstep location to permissible convex regions, such as the planar regions used in the original footstep planning algorithm. For example, when going up stairs, this can prevent the desired foothold from being adjusted off the step. The development of an intelligent, higher level constraint algorithm is desirable, as it can then select the largest convex region lying inside the current capture region as the current foothold constraint. This then would the robot, for example, to switch which cinder block the robot is trying to step to if tracking degrades.

3.5 Optimization Based Swing Speed Up

As an alternative to dynamics based swing speed up, we can also utilize a gradient descent approach for determine the swing duration. Similar to the formulation in [131], we can perform gradient descent around the quadratic cost presented in Equation 3.34 to optimize

for the time. This prevents solving the complex nonlinear cost function directly, instead finding the solution in an efficient manner with gradient descent.

To perform the gradient descent, the first step is to define the time-augmented cost function,

$$J_{\text{total}}\left(\mathbf{r}_{f,i}, \boldsymbol{\delta}, \boldsymbol{\kappa}, \boldsymbol{\eta}, T\right) = J\left(\mathbf{r}_{f,i}, \boldsymbol{\delta}, \boldsymbol{\kappa}, \boldsymbol{\eta}, T\right) + J_{\text{time}}\left(T\right), \tag{3.47}$$

where J is the cost function defined previously and J_{time} is

$$\left\|T_{ss} - T_{\text{desired}}\right\|_{R_t}^2. \tag{3.48}$$

This cost function adds a quadratic cost for deviating from the desired duration. At first glance, this appears as if it may remain a convex optimization problem. However, the values contained in J do not have a linear relationship with T, instead exponential, making the costs non-linear with respect to T. There are, however, still convex.

The next step in the gradient descent approach is to approximate the cost gradient with respect to time. To do so, we can approximate this gradient by

$$\nabla_T J_{\text{total}} = \frac{\left(J_{\text{total}}\left(\mathbf{r}_{f,i}, \boldsymbol{\delta}, \boldsymbol{\kappa}, \boldsymbol{\eta}, T + \delta T\right) + J_{\text{total}}\left(\mathbf{r}_{f,i}, \boldsymbol{\delta}, \boldsymbol{\kappa}, \boldsymbol{\eta}, T\right)\right)}{\delta T}. \tag{3.49}$$

Using this gradient, we can iteratively search for the optimal solution by defining a standard update rule where we move in the direction of the gradient,

$$T_{n+1} = T_n - \alpha \nabla_T J_{\text{total},n}, \tag{3.50}$$

where α is a sufficiently small update gain.

We can continue this iterative computation until the gradient drops below some threshold. When the magnitude of the gradient $\nabla_T J_{\text{total}}$ is sufficiently small, we know that the cost is approaching either some local or global minima. The cost is at this minima when $\nabla_T J_{\text{total}}$, however, something within some small distance of the actual minima, which is sufficient for our purposes. Due to the construction of the cost, as well, we know that the cost is still convex with respect to time, so the only minimum is the global one.

To improve the reliability of the update policy, we can examine the updated cost. If, after the update in Equation 3.50, the total cost increases, we can use a different update law,

$$T_{n+2} = 0.5 T_{n+1}. \tag{3.51}$$

This reduction will be performed until the total cost is found to decrease. This is illustrated in Figure 3.7. Without this update modification, an additional QP is required to be solved at the updated value to compute the gradient. The effect of this modified update rule is to reduce the number of iterations required for the gradient descent algorithm to find a solution.

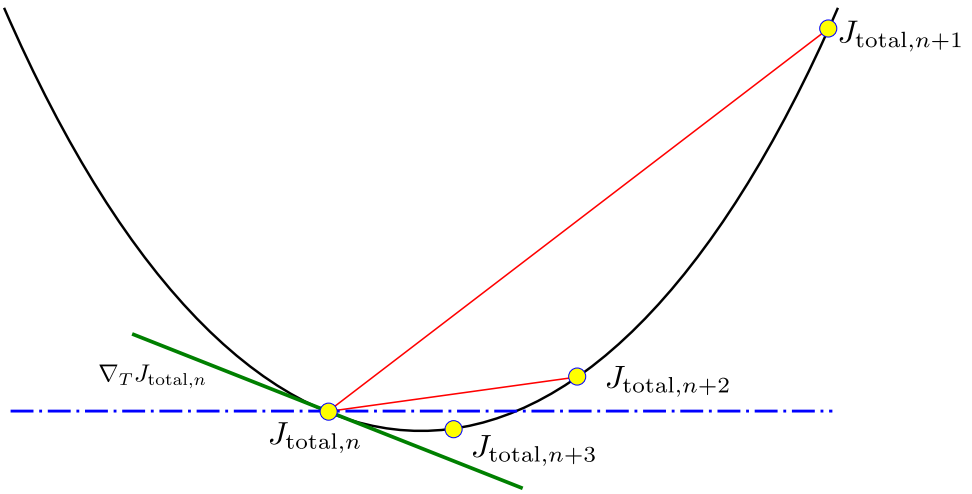


Figure 3.7: Modified update rule. If the cost is increased, the time update should be reduced.

3.6 Results

We used the above walking controller both in simulation and on the hardware platform for the Atlas robot. Using a quad-core 2.7 GHz 3rd generation i7 processor, the QP for the recursive ICP adjustment algorithm was solved using a custom active-set solver in an average $80\mu s$, while the entire algorithm took an average $220\mu s$. The QP for the simple ICP adjustment algorithm, however, was solved using the same custom solver in an average of only $60\mu s$, with the entire algorithm taking only $120\mu s$. Both of these algorithms are easily solvable in real-time at the 250Hz control-loop frequency used on the robot. The main contributor to the reduced loop time for the simple algorithm was the elimination of the slack variable, instead setting the feedback dynamics as an objective for the cost function. This removed two control variables and two Lagrange multipliers, reducing the problem from $\mathbb{R}^10$ to $\mathbb{R}^6$. The entire algorithm speed was further increased by greatly simplifying the recursive dynamics, removing the entire recursively calculated multipliers in favor of a single exponential term.

The duration required for solving the gradient descent-based timing optimization algorithm was not explicitly measured. Being an iterative solver, there are no guarantees that a solution will require the same number of iterations to find from one control tick to the next. Instead, we set a maximum duration that the solver can take, and allow it to iterate as many times within that duration as necessary. The iterations are terminated if the algorithm exceeds its deadline. It then resumes iterating on the next control tick. If the control loop rates are high enough, the problem conditions do not change too much from one tick to the next, effectively allowing the algorithm to span multiple ticks in its attempt to find a solution. This was found to be accurate enough for solutions to be found relatively quickly.

expand this

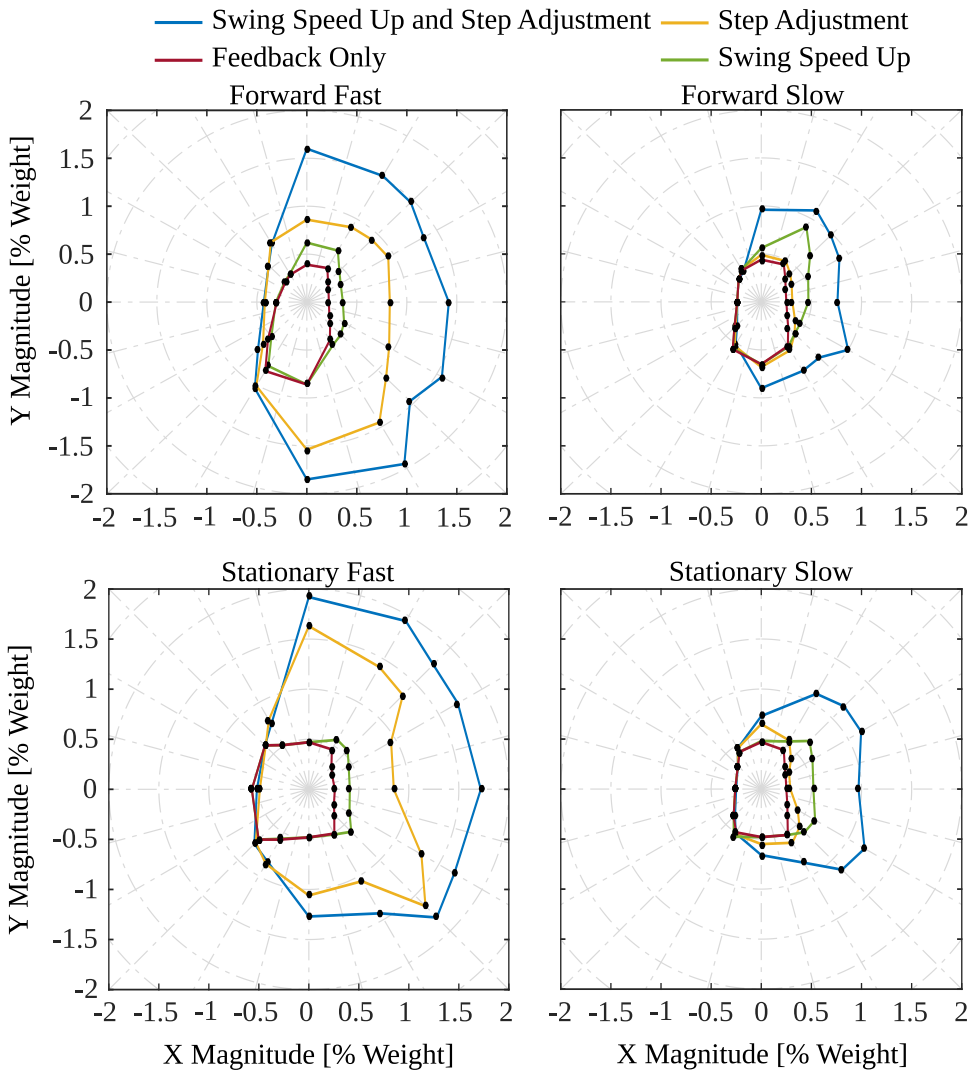


Figure 3.8: Maximum push the robot can recover from and continue walking, at different push angles and step speeds, as a function of the robot weight, using different push recovery methods. The push is applied to the center of mass for $0.1s$. Forward steps are $0.5m$ long.

To explore the effectiveness of the different ICP control mechanisms, we conducted simulations comparing the maximum external disturbance that the robot can recover from. For the recursive ICP step adjustment algorithm, we explored each of the control mechanisms: proportional feedback only, feedback with angular momentum, feedback with swing speed up, feedback with step adjustment, feedback swing step adjustment and angular momentum, and feedback with swing speed up, step adjustment, and angular momentum. The results of applying disturbances in different directions to the center of mass of the robot while it is using several different step directions are shown in Figure 3.8.

Each disturbance in all the following simulation experiments was applied to the center of

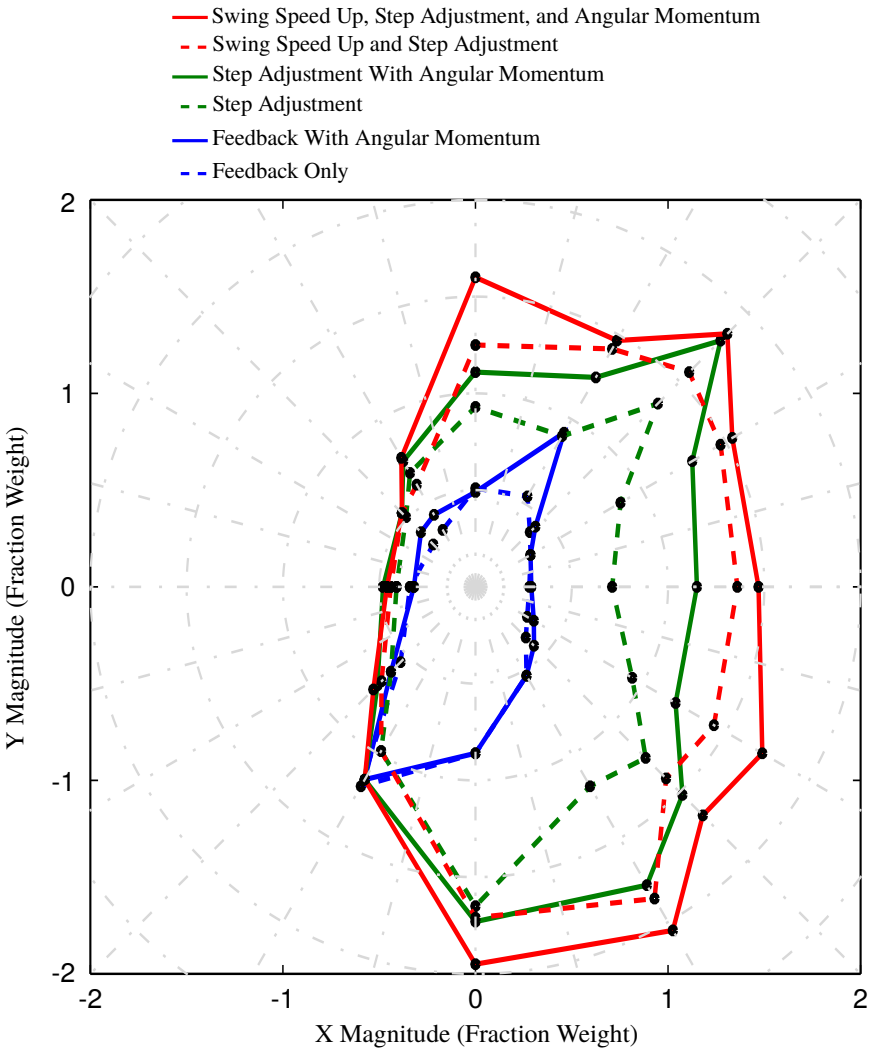


Figure 3.9: Maximum push the robot can recover from and continue walking when walking forward with $0.5m$ steps. This compares the effects of using angular momentum and not using angular momentum for stability. The push is applied to the center of mass for $0.1s$.

mass of the robot for $0.1s$ halfway through the step. The step motions included $0.5m$ forward and stationary steps, both fast ($0.95s$) and slow ($2.0s$). This minimum swing time allowed after speed up was $0.6s$.

The inclusion of some additional stabilizing mechanism to the feedback controller was found to improve disturbance rejection, while adding both speed up and step adjustment was consistently the most robust method. Speed up was generally more effective than step adjustment when walking slowly, as the corresponding required step adjustment was quite large due to the slower step speed. Exceptions to this are when tracking errors are perpendicular to the dynamics, such as being pushed forward when stepping in place. As expected, the

effectiveness of step adjustment for stabilization was dramatically increased by increasing the step speed. It is worth noting that the magnitude of recoverable disturbances using only feedback did not significantly change between the different step speeds. Using both speed up and step adjustment, the largest disturbance the robot recovered from in simulation was 1.92 times its weight, or $2937N$, when stepping quickly in place.

The effects of adding angular momentum control to the fast forward stepping motions is shown in Figure 3.9. To use angular momentum, the CMP was allowed to exit the support polygon. This provided considerable increases in the control authority of the robot, as shown by the increased area of the solid lines over the dashed lines in Figure 3.9. With very few exceptions, the control authority of feedback, step adjustment, and swing speed up with step adjustment were all increased by the addition of angular momentum in all push directions. The greatest increases were seen when pushes were applied in the backward direction. We believe that this is because it is easier for the robot to achieve ground reaction forces that generate angular momentum when the additional force is in the direction of the motion, particularly when the stance foot is behind the CoM. This, however, is somewhat a function of tuning, as well.

In Figure 3.10, a comparison between the dynamics-based swing speed up algorithm and the timing optimization algorithm is presented. As can be seen, the dynamics-based swing speed up algorithm shows considerably better improvement that the timing optimization one. As expected, the inclusion of angular momentum in both algorithms increases the magnitude of disturbances that the robot can recover from. We believe that, while the optimization-based timing adjustment algorithm does not show as high of performance, it is still a viable option. However, the dynamics-based one is evidently more reliable, and its use is preferred, based on these results. Additionally, the dynamics-based algorithm has considerably faster solve speeds, as it only requires solving one QP per control tick, as opposed to the gradient-descent based algorithm, which needs multiple solutions as it iterates every control cycle.

Similar results for the simple ICP step adjustment algorithm are presented in Figure 3.11 for the fast forward walking steps. This figure compares the results of the simple algorithm to the results of the recursive algorithm. As can be seen, the simple algorithm tends to perform comparably to the recursive algorithm, although slightly less well in several instances. In a few cases, the algorithm was terminated early, due to a few numerical instabilities. With slight further work, we expect these to be alleviated. Additionally, the simple algorithm was allowed to use much less angular momentum for stability, with the CMP required to be within 6cm of the support polygon, limiting the amount of angular momentum that could be used for stabilization. For the recursive algorithm, this bound was much larger. Additionally, the simple algorithm has several distinct advantages over the recursive algorithm, which will be elaborated on further in the next section.

The real robot was also able to successfully use all these various algorithms to adjust the step timing and locations to compensate for large tracking errors. In the first two experiments, we tested the recursive step adjustment algorithm using swing speed up and step adjustment.

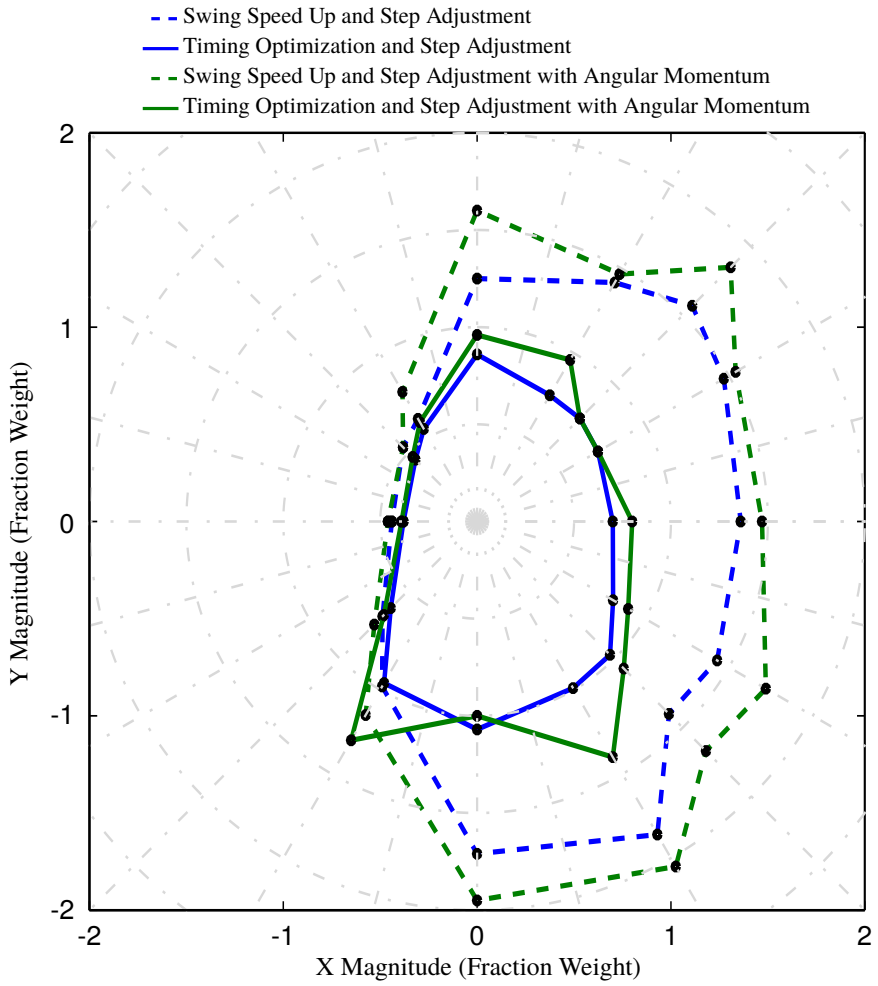


Figure 3.10: Comparison of the maximum recoverable disturbances between the dynamics-based swing speed up algorithm and the timing optimization-based speed up algorithm when the robot is walking forward. As can be seen, the dynamics-based algorithm consistently outperforms the optimization based algorithm.

The algorithm was not allowed to use angular momentum at the time. We forced tracking errors by pushing the robot while stepping. In the presented experiments, the steps durations were $2s$, with $1s$ spent in transfer and $1s$ in swing. Figure 3.12 shows the results of applying an outward push when stepping in place. As can be seen, the reference time is advanced during swing to speed up the ICP trajectory, and the foot is adjusted outward to help maintain balance. Figure 3.13 shows the results of a forward push while the robot is walking. Again, the swing state is sped up, and the step adjusted in the direction of the push. Low frequency oscillations in the ICP position occurred after heel strike due to the high speed at which the robot set the foot down, but were quickly damped out. The impact speed also resulted in additional ICP tracking errors in the direction of the stance foot, but this was

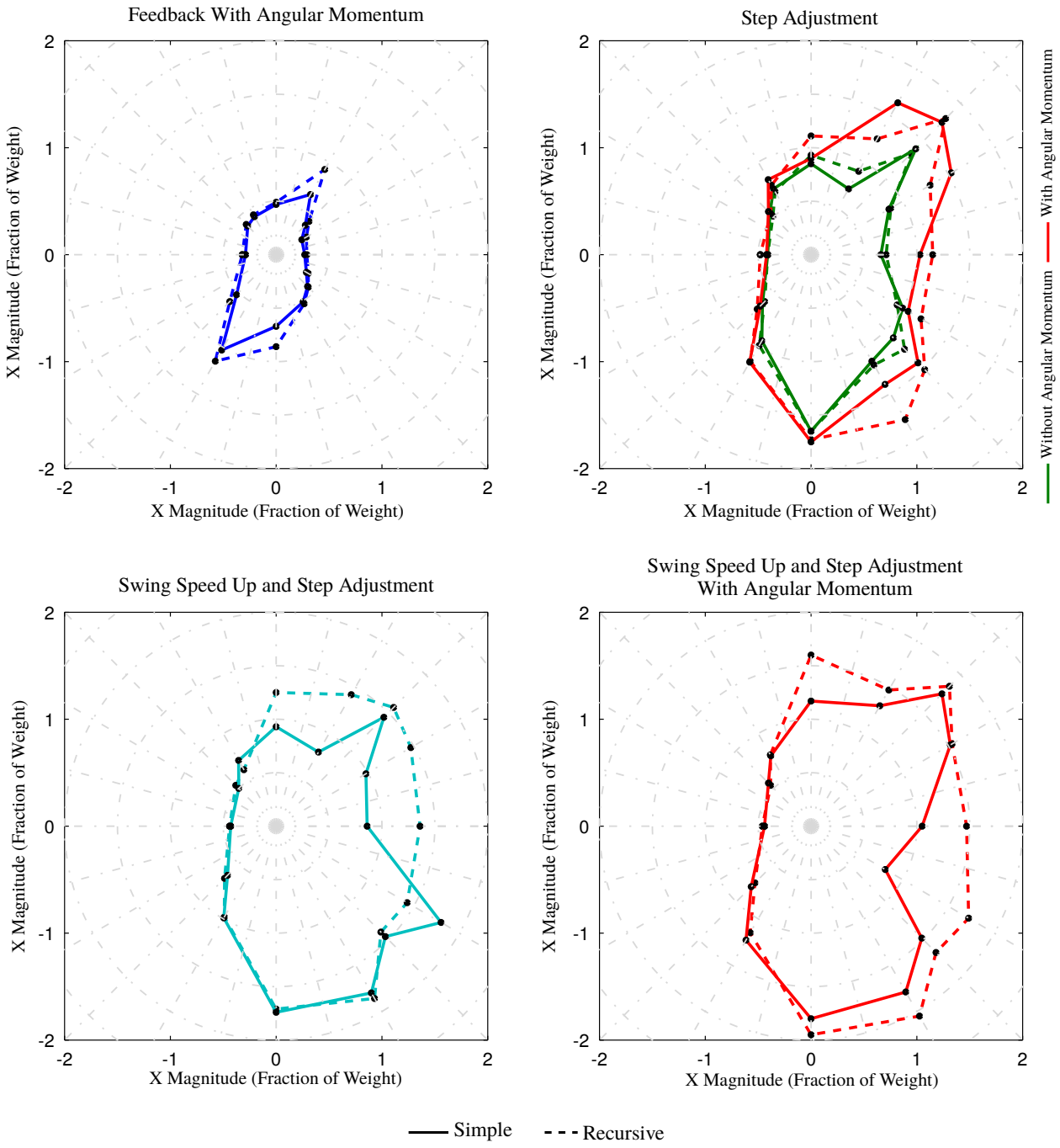


Figure 3.11: Comparison between the recoverable pushes with the recursive and simple plans when walking forward. As can be seen, the performance is comparable between the two algorithms, with the recursive algorithm typically performing slightly better. Both algorithms greatly improve the control authority of the robot.

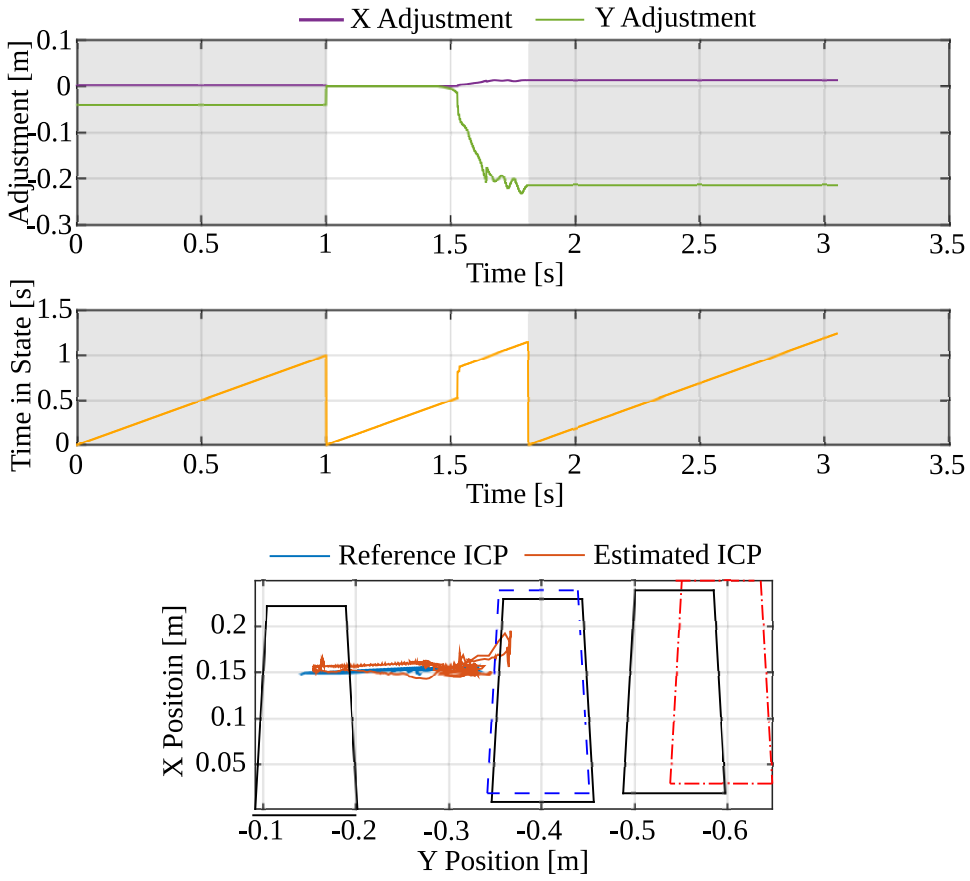


Figure 3.12: Results of applying an outward push when stepping in place. The gray background represents the transfer phase. The dashed blue foot is the reference footstep, the dashed red footstep is the reference footstep with adjustment, and the black footstep is the actual foot location.

easily corrected given the additional control authority during transfer.

Further experiments which tested the timing optimization algorithm were conducted, as well, with images of the results shown in Figure 3.14 and Figure 3.15. For these experiments, the same step timing parameters of 1.0s swing and 1.0s transfer were chosen. The robot was commanded to take very short steps, only approximately 4cm in length, to illustrate the step adjustment abilities. It was then pushed forward in the first experiment, and sideways in the other. In both cases, the robot was able to quickly set the foot down, adjusting it in the direction of the ICP error. The use of angular momentum was also allowed for recovery. In these experiments, the amount of angular momentum allowed was unbounded, resulting in the robot using a considerable (possibly unsafe) amount to help recover.

No hardware experiments using the simple ICP step adjustment algorithm are presented in this chapter. However, all results in Chapter 5 utilize that algorithm for control, so please

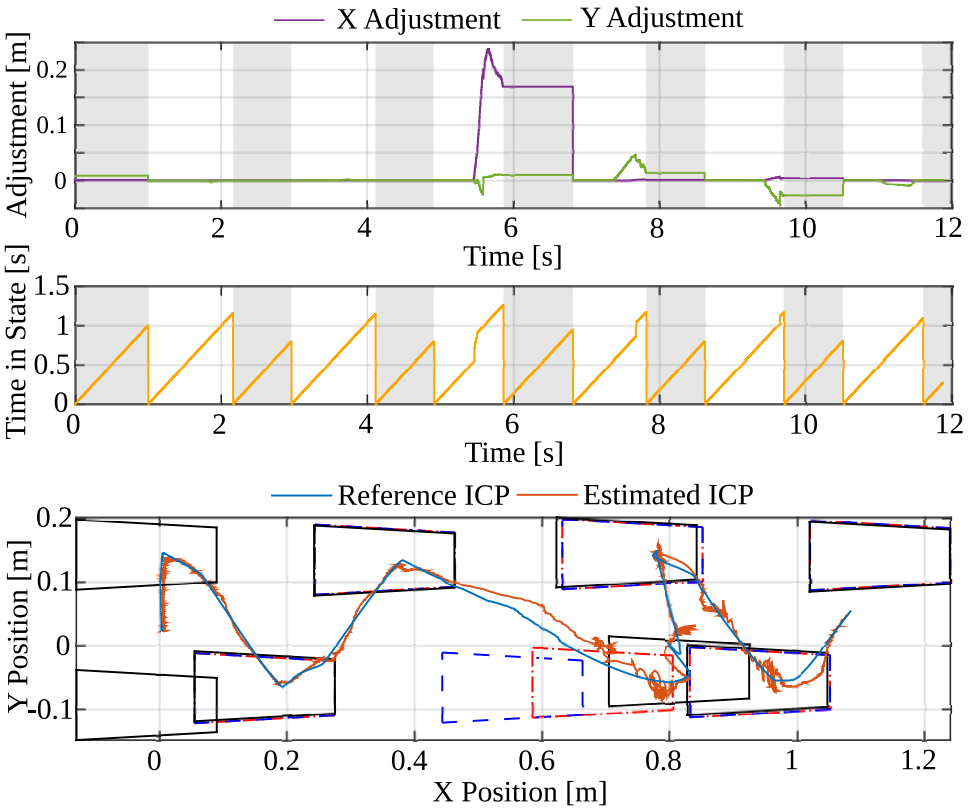


Figure 3.13: Results of applying an forward push when walking forward. The gray background represents the transfer phase. The dashed blue foot is the reference footstep, the dashed red footstep is the reference footstep with adjustment, and the black footstep is the actual foot location.

see that chapter for more details on the performance of the simple algorithm on hardware platforms.

3.7 Discussion

While the maximum recoverable disturbances in different directions provides some degree of a useful metric for comparing the different algorithms, it is not necessarily the most effective means for determining the advantages and disadvantages for each. For one, these results are heavily skewed based on the quality to which each algorithm is tuned. For example, an algorithm that has theoretically lower peak performance than another may actually perform better if the second is not tuned as well as the first. Indeed, given the number of different gains and costs that can be varied, tuning can be somewhat of a tedious process, with no guarantees on how close to the theoretical limits of the algorithm the resulting performance is.

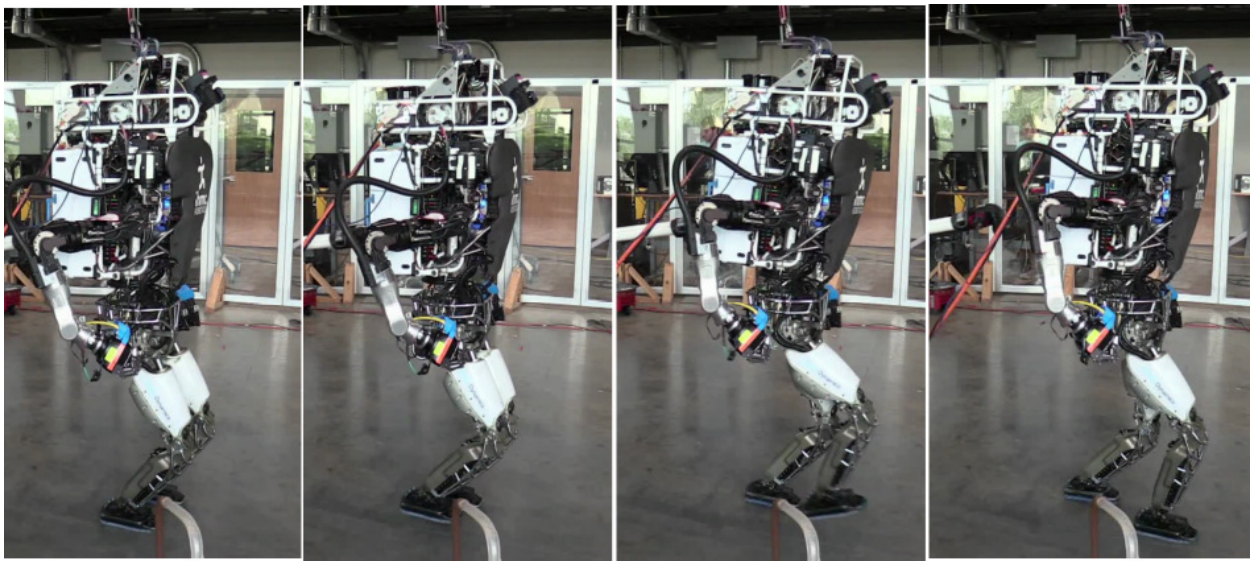


Figure 3.14: Results of applying an forward push when walking forward. The gray background represents the transfer phase. The dashed blue foot is the reference footstep, the dashed red footstep is the reference footstep with adjustment, and the black footstep is the actual foot location.

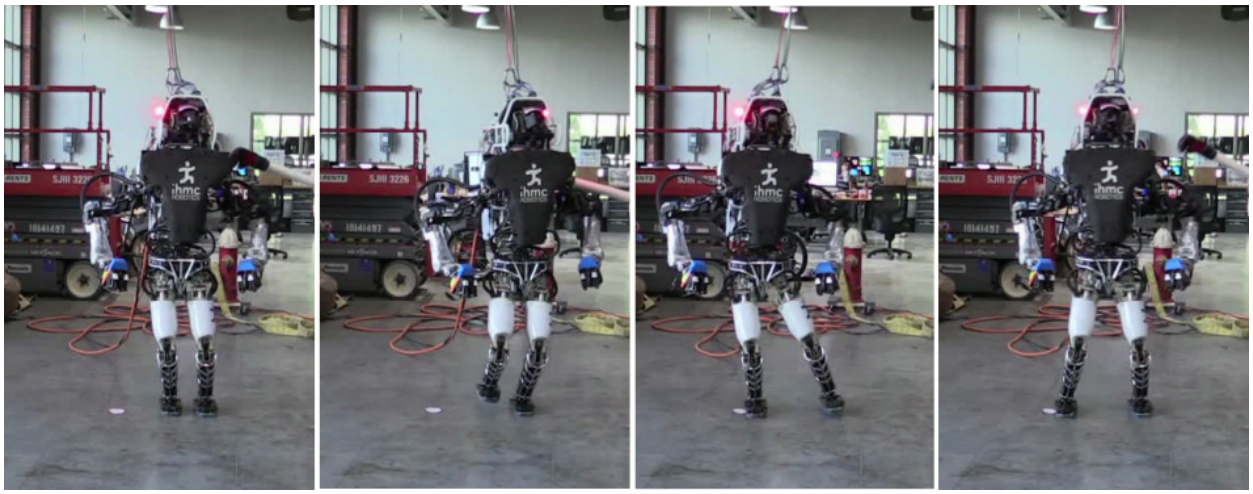


Figure 3.15: Results of applying an forward push when walking forward. The gray background represents the transfer phase. The dashed blue foot is the reference footstep, the dashed red footstep is the reference footstep with adjustment, and the black footstep is the actual foot location.

When comparing the three high-level algorithms, being recursive step adjustment, timing

optimization, and simple step adjustment, it was noticed that, on hardware, the simple step adjustment algorithm tended to have the best performance. We believe that this is primarily due to its simplicity and predictability. While the recursive algorithm showed very high performance in simulation, this performance did not transfer as well to hardware. As the simple algorithm is based only on the ICP dynamics, rather than the recursive ICP plan, it adjusts the footstep by a very predictable amount, rather than by what the encoded recursive ICP plan would predict. This translates well to making it more robust against unmodeled dynamics, as it tends to slightly over adjust what is actually strictly necessary. The algorithm design also makes it easy to tune to force additional step adjustment, further increasing the robustness to swing foot tracking errors. The simple algorithm also has the inherent advantage that it is not dependent on the ICP plan. This means that, as the ICP plan is improved, these improvements can be taken into account immediately, rather than requiring a redesign of the recursive dynamics. This then enables advanced ICP planning techniques, including nonlinear ones, that cannot be captured by the recursive ICP step adjustment algorithm.

While the presented algorithms require fairly accurate control of the CoP and CMP, the ability to adjust the step outward based on the ICP dynamics somewhat relaxes this. By expanding the support polygon, the robot's CoP control authority is less likely to become saturated by operating further from the support polygon boundary, where accuracy is lowest as well. On the Atlas robot, the CoP is controllable with an accuracy of approximately $2cm$ due to good force control in the ankle joints. However, as we are not directly measuring the CMP, there may be unquantified tracking errors caused by unmeasured deviations of the actual CMP from the actual CoP. Additionally, it was found that the robot had some difficulty in achieving the desired angular momentum rate of change. This means that, as the requested deviation of the CMP from the CoP increases, the accuracy of the CMP control correspondingly decreases. While this is detrimental, angular momentum is often the "last resort" control variable, meaning that the robot is already doing everything it can to recover. This means that, while the CMP control errors decreases the maximum amount of tracking error that can be recovered from, the performance when compensating for smaller errors does not really suffer.

The proposed algorithms only provide marginal improvements against tracking errors in the inward direction of the step. The step reachability polygon does not allow for any crossover of the steps, simply constraining them to a minimum inward position. This is partly due to the difficulties in defining a reachability region that enables crossover while maintaining convexity. By defining multiple possible reachability constraints and selecting the active one based on the current step type and tracking errors, however, crossover could be possible. Despite this, most robots do not actually have the range of motion in the hips to perform crossover steps. The increased robustness observed is primarily due to the addition of angular momentum. The other option is to set the swing foot down quickly, so that the opposite step can then be adjusted outward, in the direction of the step. However, due to the exponential relationship between the ICP dynamics and the stance foot location, even slight inward

tracking errors will grow very large very quickly if the support foot is switched, particularly with the long required swing times often found on robots. As such, this is not really a viable mechanism for recovering from inward pushes.

A variety of factors led to performance limitations of these controllers when ported from simulation to hardware. These include: Errors in the robot model. When using an inverse dynamics-based approach, model accuracy greatly affects the resulting ground reaction forces. If the controller cannot effectively achieve the CoP at the support polygon edge, it will not be able to as successfully mitigate tracking errors; Actuator speed and torque limits, which bounds how quickly the robot can step. By increasing this step speed, we expect the effectiveness of step adjustment algorithms to greatly improve, as illustrated in Figure 3.8; Sensor noise, which greatly affects the precision of the ICP calculation. Measurement uncertainty further exacerbates inaccuracies in the inverse-dynamics calculation, as well as other task-space controllers.

3.8 Conclusion

The ability to robustly recover from large tracking errors is essential to improving the capabilities of humanoid robots, and represents a critical step forward in enabling them to competently function in uncertain environments. In this work, we presented a new approach for adjusting both step timing and locations to reject external disturbances and their corresponding tracking errors. By including step timing adjustment, the required step adjustment to reject errors is exponentially decreased. Our algorithm formulates this problem in a highly efficient manner, allowing it to be solved quickly in real-time. In the future, we hope to incorporate angular momentum in the algorithm to further increase the control authority available to the robot. We also plan to integrate the step timing adjustment into the optimization algorithm, borrowing from the gradient descent approaches used by air vehicles [131]. We will additionally include environmental information to allow the step adjustment algorithm to be used effectively in dynamic and cluttered environments.

Chapter 4

Capture Point Trajectories for Reduced Knee Bend using Step Time Optimization

Traditional humanoid robot walking utilizes highly bent knees, resulting in high torques, inefficient, and unnatural motions. Even with advanced planning of center of mass height trajectories, significant amounts of knee bend can be required due to arbitrarily chosen step timing. In this work, we present a method that examines the effects of adjusting the step timing to produce plans that require a specifiable amount of knee bend. We define a quadratic program that optimizes the step timings using the gradients of the instantaneous capture point trajectories with respect to time. We execute this using a simple iterative feedback approach to account for the higher order terms that are ignored by the quadratic program. We then illustrate the effectiveness of this algorithm by comparing the walking gait of the simulated Atlas humanoid with and without the algorithm, showing that the algorithm significantly reduces the amount of knee bend during execution. We aim to later use this approach to achieve natural, efficient walking motions on humanoid robot platforms.

4.1 Introduction

Humanoid robots have been demonstrated capable of robustly walking across flat surfaces, recovering from pushes and uncertainties, and walking across varying terrain. They are also demonstrating increasingly dynamic behavior when walking, a major step forward from the slow, quasi-static walking that was previously standard. However, while there are some exceptions, most all of the gaits employed feature highly bent knees, walking with an almost "squatted" motion. This is highly unnatural, and, while it offers some control benefits, results in significant increases in power consumptions of the knee [22] over that seen in

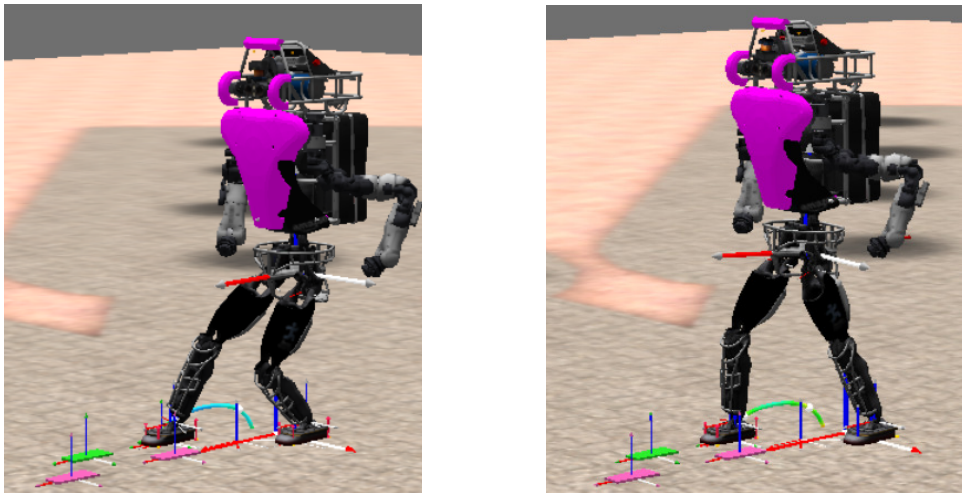


Figure 4.1: Without considering the effects of step timing, ICP plans can require significant knee-bend to execute (left). Modifying the timing, however, can result in dynamic, natural, and efficient dynamic plans (right).

humans [132]. It additionally decreases ground clearance of the robot and creates clearance issues when walking over varied terrain.

Before tackling the host of control challenges introduced by walking with straighter legs, the underlying dynamic plans used by the robot must be conducive to walking with less-bent knees. While whole-body planning schemes that consider the full rigid-body dynamics of the robot have made significant improvements in speed, they typically require too long of solve times for online implementation. Instead, a typical approach for dealing with the highly nonlinear and high dimensional problem of planning and controls for humanoids is to restrict consideration to the center of mass (CoM). These low-order dynamic representations have been often preferred for online implementation due to their speed and efficiency. The zero-moment point (ZMP) is an often utilized representation of the CoM dynamics, and is the point on the ground plane where the moment induced by the inertial and gravitational force is perpendicular to the surface [21], It has been used to generate stable walking motions that avoid tipping about the foot edges of the robot [22, 133, 25]. Alternatively, the instantaneous capture point (ICP) [17] and divergent component of motion (DCM) [18] were introduced as stable state transformations of the CoM, and have been show as a highly effective strategy for momentum-based planning and control of humanoid robots [42, 38].

Several works have attempted to address planning and control of height trajectories. In [134], the authors presented an approach for generating straightened-knee trajectories by combining model predictive control (MPC) of the ZMP to generate horizontal CoM trajectories with a spring-damper approach to generate vertical CoM trajectories. Alternatively, [112] used linear differential inclusion to incorporate these height trajectories directly into the MPC, resulting in relatively natural, cyclical motions of the CoM. In [135], the authors

also included CoM height in their ZMP-based quadratic program, but required sequential quadratic programming to solve the resulting problem's quadratic constraints. The planning and control of height trajectories as they affect the DCM was addressed in [38], which relaxed the assumption of a constant pendulum height in an attempt to increase control of the CoM height over variable terrain. This last is an example of the standard planning approach, as noted by [112], with CoM height planning typically done independently of horizontal planning, leaving planners ill-equipped to deal with kinematic limitations. This issue comes to the forefront when attempting to walk with straighter legs.

While some previous works are capable of generating kinematically feasible height trajectories [134, 112], these approaches, along with most others, typically rely on arbitrarily chosen single and double support phase durations. While kinematic feasibility of the CoM height plan is guaranteed, this has the fundamental limitation of ignoring the effects of step timing on the resulting horizontal CoM plans. This is, in turn, ignoring the kinematic constraints placed on the system by the timing and the dynamic plan, potentially requiring significant knee-bend during execution. As the walking speed of most humanoid robot platforms is still relatively slow, this becomes common, with the robot needing to crouch for the next step to be kinematically feasible, as shown on the left of Figure 4.1. As the step length increases, the required knee-bend of this slow step correspondingly increases, as well. To address this, we present in this work an approach for optimizing the swing and transfer durations to produce ICP trajectories that do not require knee-bend beyond an amount specifiable by the operator. We utilize a novel quadratic program that uses the gradient of the ICP plan w.r.t. to time to compute the required timing adjustments that satisfy the desired maximum and minimum knee-bend kinematic constraints. We believe that proper timing selection is critical to the continued progress of humanoid locomotion towards efficient, human-like gaits.

4.2 Dynamic Planning Background

An increasingly common approach for dynamic planning in humanoid robots is to utilize a stable transformation of the CoM position and velocity in the form of either the ICP, DCM, or extrapolated center of mass (XCoM) [17, 18, 40, 41, 136, 38, 36, 42]. In this work, we will refer only to the ICP, noting its equivalency to the DCM and XCoM in the x-y plane. The ICP is, as mentioned, a stable transformation of the CoM state, and is defined as

$$\boldsymbol{\xi} = \mathbf{x} + \tfrac{1}{\omega}\dot{\mathbf{x}}, \tag{4.1}$$

where $\boldsymbol{\xi}$ is the ICP position, $\mathbf{x}$ and $\dot{\mathbf{x}}$ are the CoM position and velocity, and $\omega = \sqrt{g/\Delta z_{\text{com}}}$ is the natural frequency of the inverted pendulum. By reordering this, we can see that the CoM has stable first order dynamics with respect to the ICP, meaning that it will converge

to the ICP over time. Through differentiation, the ICP dynamics are defined as

$$\dot{\boldsymbol{\xi}} = \omega \left(\boldsymbol{\xi} - \mathbf{r}_{\text{cmp}} \right), \tag{4.2}$$

where we see that the Centroidal Moment Pivot (CMP) point [130], $\mathbf{r}_{\text{cmp}}$, controls the ICP dynamics.

4.2.1 Dynamic Planning of ICP Trajectories

There are a variety of methods that have been used to generate stable ICP trajectories, such as [38], where discrete trajectories were generated numerically from specified ZMP trajectories. In this work, we use the algorithm proposed in [42], summarized in the following paragraphs.

From the definition of the ICP dynamics in Equation 4.2, the differential equation has an analytic solution

$$\boldsymbol{\xi}(t) = e^{\omega t} \left(\boldsymbol{\xi}_0 - \mathbf{r}_{\text{cmp}} \right) + \mathbf{r}_{\text{cmp}}, \tag{4.3}$$

assuming $\mathbf{r}_{\text{cmp}}$ is held constant. Using this, we can calculate a desired ICP trajectory for walking, given a set of desired footsteps and desired CMP locations in those footsteps. To more accurately represent human-like walking, we use two CMPs per foot, one in the heel ($\mathbf{r}_{\text{cmp},H}$) and one in the toe ($\mathbf{r}_{\text{cmp},T}$), as shown in Figure 4.2 by the green circles. This results in the reference CMP trajectory moving from the heel to the toe in the foot while stepping.

To determine the amount of time spent on the CMP, we can break the ICP plan into four segments, $T_{\text{iniDS}}, T_{\text{endDS}}, T_{\text{iniSS}}$, and T_{endSS}. These correspond roughly to the time in transfer shifting the weight to the upcoming support foot, the time in transfer shifting the weight forward in the foot, the time in swing shifting the weight forward in the foot, and the time spent in swing with the weight in the front of the foot, respectively. The desired ICP trajectory can calculated by recursing backward from the final objective location. This can be done by using the solution to the ICP dynamics in Equation 4.3, and assuming a static CMP location, allowing the ICP locations to be computed at the beginning and end of swing and transfer. We can smooth the ICP trajectories using third order polynomial interpolation between each of these points, which guarantees smoothness of the CMP trajectory [42], resulting in the light blue and orange colored lines in Figure 4.2.

Note that the effect of taking slower steps is highlighted in Figure 4.2. This results in much more of the ICP motion occurring during transfer. This, in turn, has great effects on the location of the CoM when walking.

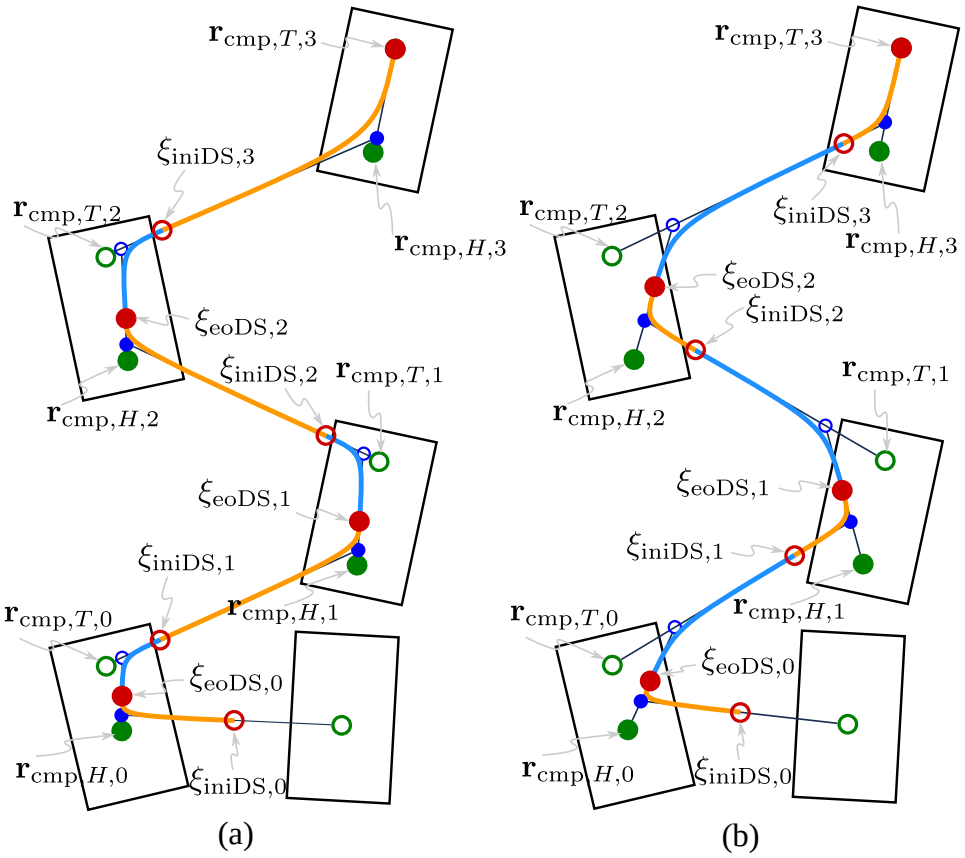


Figure 4.2: Heel-to-Toe ICP trajectory with smooth CMP trajectories for slower (left) and faster (right) steps [42].

4.2.2 Computation of CoM Trajectory Solutions

While the utilization of the ICP for dynamic planning has the major advantage of not requiring specific calculation of CoM trajectories, this is required for determination of kinematic feasibility. In particular, we want to know the location of the CoM at touchdown, as this is the farthest point from both the leading and trailing feet. By using the ICP algorithm in [42], we can obtain an analytic solution for the CoM trajectory. If the ICP trajectory is defined using a constant CMP, as in Equation 4.3, the CoM trajectory can be defined by integrating the CoM dynamics

$$\dot{\mathbf{x}}(t) = \omega \left(e^{\omega t} \left(\boldsymbol{\xi}_0 - \mathbf{r}_{\mathrm{cmp}} \right) + \mathbf{r}_{\mathrm{cmp}} - \mathbf{x}(t) \right), \tag{4.4}$$

which has the solution

$$\mathbf{x}(t) = \tfrac{1}{2} e^{\omega t} \left(\boldsymbol{\xi}_0 - \mathbf{r}_{\mathrm{cmp}} \right) - \tfrac{1}{2} e^{-\omega t} \left(\boldsymbol{\xi}_0 + \mathbf{r}_{\mathrm{cmp}} - 2\mathbf{x}_0 \right) + \mathbf{r}_{\mathrm{cmp}}. \tag{4.5}$$

Alternatively, if the ICP trajectory is defined using a cubic polynomial, the CoM dynamics become

$$\dot{\mathbf{x}}(t) = \omega \left(\mathbf{c}_0 + \mathbf{c}_1 t + \mathbf{c}_2 t^2 + \mathbf{c}_3 t^3 - \mathbf{x}(t) \right), \tag{4.6}$$

which has the solution

$$\begin{aligned} \mathbf{x}(t) =& \mathbf{c}_3 t^3 + \left(\mathbf{c}_2 - \tfrac{3}{\omega}\mathbf{c}_3 \right) t^2 + \left(\mathbf{c}_1 - \tfrac{2}{\omega}\mathbf{c}_2 + \tfrac{6}{\omega^2}\mathbf{c}_3 \right) t \\ &+ \left(\mathbf{c}_0 - \tfrac{1}{\omega}\mathbf{c}_1 + \tfrac{2}{\omega^2}\mathbf{c}_2 - \tfrac{6}{\omega^3}\mathbf{c}_3 \right) \\ &+ e^{-\omega t} \left(\mathbf{x}_0 - \mathbf{c}_0 + \tfrac{1}{\omega}\mathbf{c}_1 - \tfrac{2}{\omega^2}\mathbf{c}_2 + \tfrac{6}{\omega^3}\mathbf{c}_3 \right). \end{aligned} \tag{4.7}$$

Using these equations, we can then calculate where the CoM will be located at touchdown.

4.3 Center of Mass Adjustment Calculation

Before we can calculate the desired timing adjustments in the ICP plan, we must determine *how* we want to adjust the plan itself. To do this, we can determine the requirements placed on the CoM location that satisfy our kinematic constraints placed on the system; namely that of maximum and minimum leg extension. We can use these constraints to define the "feasibility region" for the CoM as the area where the CoM can be located so that both legs do not bend more or less than desired.

To compute the three dimensional feasibility region, we can observe that placing limits on the amount of knee bend at both full retraction and full extension acts to constrain the location of the robot's hips. We can calculate these areas exactly by defining a series of spheres whose radius is a function of the retracted (when it is most bent) and extended (when it is least bent) leg length. By defining a maximum and minimum amount of leg bend, $\theta_{\max}$ and $\theta_{\min}$, respectively, we can calculate the corresponding bent and extended leg length, l_{min} and l_{max}. We can then use this to define a sphere centered at the ankle, $B_{\min}$ that the hip joint must be located outside of, shown by the dark green spheres in Figure 4.3. In doing so, we then know that as long as, at touchdown, both hip joints are located outside of these spheres, the ICP plan will not require either knee to bend past $\theta_{\max}$. Additionally, a second set of spheres can be defined, $B_{\max}$, shown by the lighter green spheres in Figure 4.3, to constrain the hip locations to be achievable at full leg extension.

Using these concentric set of spheres, we can compute the 3D feasibility region, $\mathcal{F}_{3D} \subset \mathbb{R}^3$, to constrain the CoM location by saying the hip locations must be within $B_{\max}$ and outside of $B_{\min}$, or

$$\mathbf{r}_{\text{hip}} \in \mathcal{F}_{3D} \equiv \{ B_{\max} - \ B_{\min} \} \subset \mathbb{R}^3, \tag{4.8}$$

which is illustrated in Figure 4.3. Here, the ankle locations are the blue balls, while the hip locations of the robot are the red balls. As can be seen, if both hip- are located inside the sphere of the maximum leg length and outside the sphere of the minimum leg length at touchdown, the plan is both kinematically achievable and will not require bending either

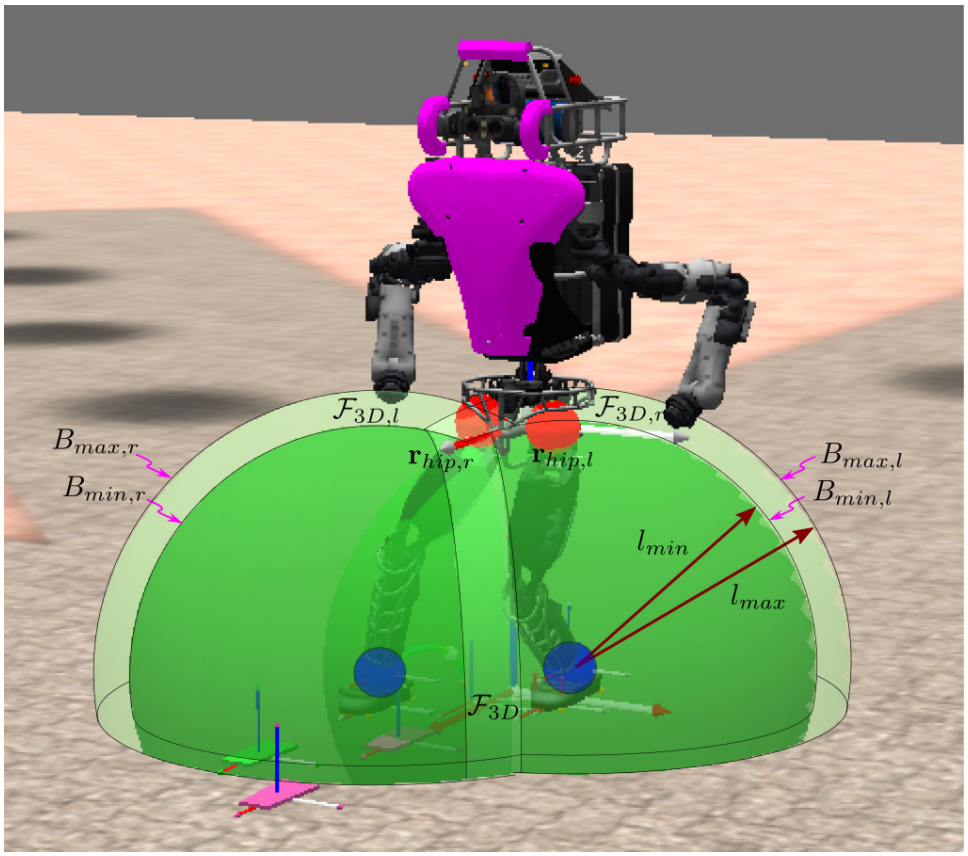


Figure 4.3: Illustration of the kinematic constraints we are using to calculate the desired CoM adjustments. The light green spheres represent the area in which the hips (the red dots) must be located to be reachable with the legs at maximum extension. The dark green spheres represent the area outside of which the hips must be located to guarantee bending less than some specifiable amount.

knee past $\theta_{\max}$.

From the 3D feasible space, we can compute the 2D feasibility region, $\mathcal{F}_{2D} \subset \mathbb{R}^2$, in the x-y plane where the CoM must be located, shown as the shaded green region in Figure 4.4. To do so, we assume that the distance from the CoM to each hip can be treated as a constant, a fairly common [134] and accurate assumption due to the mass distribution on most humanoid robots. This allows us to offset the sphere origins from the ankle locations, defining the blue circles in Figure 4.4 as $\mathbf{s}_l$ and $\mathbf{s}_r$, enabling us to directly consider the CoM rather than the hip locations as in Figure 4.3. $\mathcal{F}_{2D}$ then becomes much simpler to calculate, being bounded on the sides by straight edges perpendicular to the vector from one ankle to the other when walking with constant height,

$$\overrightarrow{\mathbf{F}}_{\min} \perp \overrightarrow{\mathbf{s}_{l,r}}, \qquad \overrightarrow{\mathbf{F}}_{\max} \perp \overrightarrow{\mathbf{s}_{l,r}}, \tag{4.9}$$

and by arcs centered on $\overrightarrow{\mathbf{s}_{l,r}}$ when changing height,

$$\widehat{\mathbf{A}}_{\min}\left(\overrightarrow{\mathbf{F}}_{\min} \cap \overrightarrow{\mathbf{s}_{l,r}}\right), \quad \widehat{\mathbf{A}}_{\max}\left(\overrightarrow{\mathbf{F}}_{\max} \cap \overrightarrow{\mathbf{s}_{l,r}}\right). \tag{4.10}$$

This follows directly from projecting $\mathcal{F}_{3D}$ in Equation 4.8 onto the x-y plane.

The objective then becomes to compute the distance required to project the CoM into the feasibility region,

$$\Delta x_{\parallel} = d(\mathbf{x}, \mathcal{F}_{2D}). \tag{4.11}$$

We can simplify this adjustment as being along the vector $\overrightarrow{\mathbf{s}_{l,r}}$, the red line in Figure 4.4. From the definition of the bounds of the set in Equation 4.9 and Equation 4.10, this is equivalent to

$$\Delta x_{\parallel} = \begin{cases} d(\mathbf{x}, \overrightarrow{\mathbf{F}}_{\max}), & x > \overrightarrow{\mathbf{F}}_{\max}, \\ 0, & \overrightarrow{\mathbf{F}}_{\min} \leq \mathbf{x} \leq \overrightarrow{\mathbf{F}}_{\max}, \\ d(\mathbf{x}, \overrightarrow{\mathbf{F}}_{\min}), & x < \overrightarrow{\mathbf{F}}_{\min}. \end{cases} \tag{4.12}$$

Note that this yields a conservative estimate of the required adjustment, as

$$d(\mathbf{x}, \overrightarrow{\mathbf{F}}) \geq d(x, \widehat{\mathbf{A}}). \tag{4.13}$$

This then provides the desired CoM adjustment of ICP plan to achieve the desired kinematic motions.

4.4 Timing Optimization Algorithm

Now that we can compute the desired CoM adjustment, we must develop a tool for modifying the ICP plan to achieve this adjustment. The CoM trajectory is a function of the CMP position and step timing. As the CMP position is defined according to the step plan, it is more or less fixed, requiring the durations used in the ICP plan to be modified to achieve the necessary CoM adjustment. The objective CoM adjustment can be found by computing the CoM location at touchdown, $\mathbf{x}_f$, at the beginning of each step through integrating the dynamic plan. Using the planning algorithm in section 4.2, this can be done with the analytic solutions in Equation 4.5 and Equation 4.7, although it can be found using numerical integration for any ICP plan, as well. Then, using $\mathbf{x}_f$, we can compute the objective parallel adjustment $\Delta x_{\parallel}$ using the approach outlined in section 4.3. The ICP plan however, is a function of several timing elements, meaning the timing adjustments cannot be explicitly solved for from only the desired CoM adjustment, as the problem is under-constrained. Instead, we can cast the problem in an optimization framework, defining additional cost objectives to calculate an optimal set of timing adjustments. We will do so in the following section by defining a quadratic program (QP) with an iterative feedback loop.

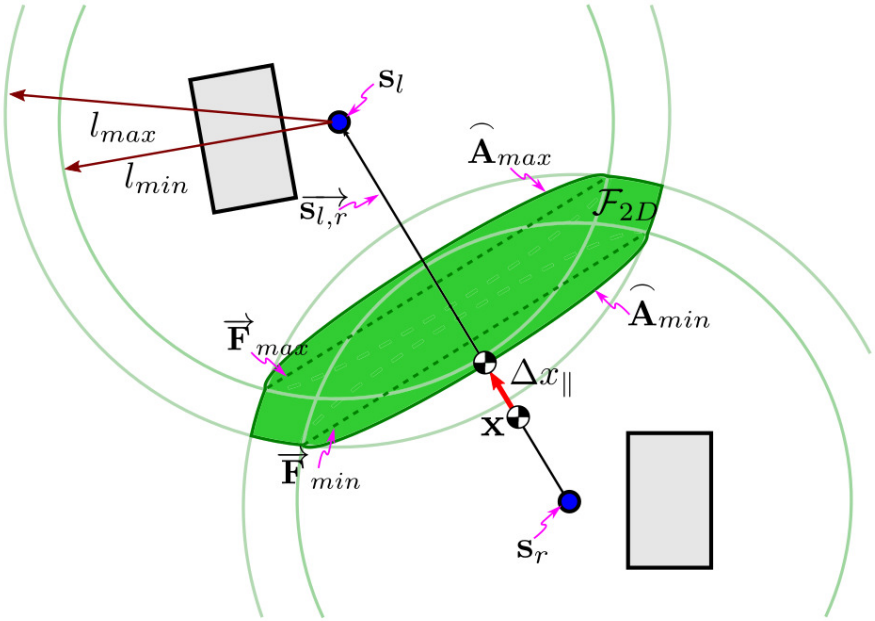


Figure 4.4: Illustration of the approach for computing the desired CoM adjustment. The region where the CoM is both kinematically feasible and satisfies the max knee bend is represented by the shaded green area. The desired CoM adjustment is the distance from CoM to the feasibility region.

From the definition of the ICP plan in section 4.2, we can define the step timing variables to be used in the adjustment calculation as

$$\begin{aligned}\mathbf{T} = \big[\ & T_{\text{iniDS},0} \quad T_{\text{endDS},0} \quad T_{\text{iniSS},0} \quad \dots \\ & T_{\text{endSS},0} \quad T_{\text{iniDS},1} \quad T_{\text{endDS},1}\ \big]^T,\end{aligned} \tag{4.14}$$

where the subscript number indicates the associate step, with 0 being current. While these variables are specific to the ICP planning approach presented in [42], variables for other ICP planning approaches can be equivalently defined. Due to the definition of the ICP dynamics, however, the ICP and resulting CoM plans are highly nonlinear with respect to time, where the CoM location at touchdown can be defined as some nonlinear function,

$$\mathbf{x}_f = \mathbf{f}(\mathbf{T}). \tag{4.15}$$

A method of determining the optimal durations, $\mathbf{T}^*$ is required. While nonlinear optimization is possible, we prefer to use a QP due to its reliability and efficiency. To do so, we can approximate the gradients of $\mathbf{f}(\cdot)$ with respect to $\mathbf{T}$,

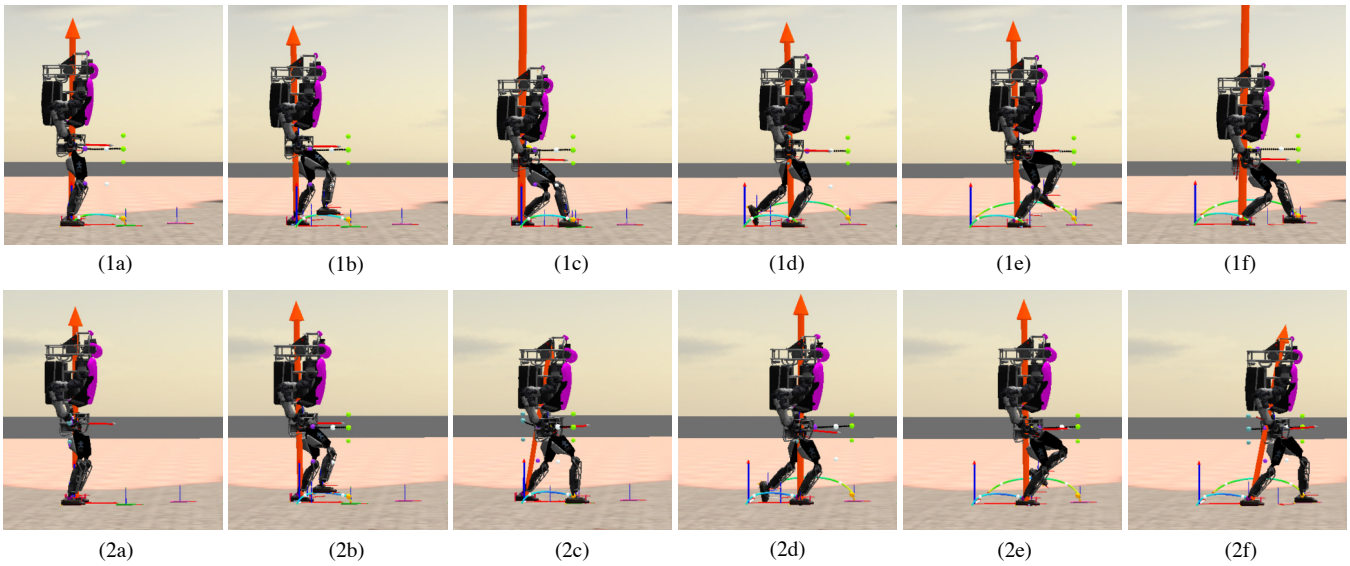


Figure 4.5: Comparison between walking without the time optimization module active (top) and with the time optimization module active (bottom). The desired steps are 0.6m long, using a 2.5s swing and 2.5s transfer. In this case, the maximum desired knee bend is 0.4 rads.

$$\nabla_{\mathbf{T}}\mathbf{f} = \begin{bmatrix} \nabla\mathbf{f}_{T_{\text{iniDS},0}} & \nabla\mathbf{f}_{T_{\text{endDS},0}} & \nabla\mathbf{f}_{T_{\text{iniSS},0}} & \cdots \\ \nabla\mathbf{f}_{T_{\text{endSS},0}} & \nabla\mathbf{f}_{T_{\text{iniDS},1}} & \nabla\mathbf{f}_{T_{\text{endDS},1}} & \end{bmatrix}, \tag{4.16}$$

by slightly perturbing the function $\mathbf{f}(\cdot)$ such that

$$\delta\mathbf{x}_f = \nabla_{\mathbf{T}}\mathbf{f}\dot{\delta}\mathbf{T} + \text{H.O.T.}. \tag{4.17}$$

The gradient function in Equation 4.16 can be broken into components parallel and perpendicular to the desired CoM adjustment, $\nabla f_{\parallel}(\mathbf{T})$ and $\nabla f_{\perp}(\mathbf{T})$, respectively. This then allows us to define a QP using $\nabla_{\mathbf{T}}\mathbf{f}(\mathbf{T})$.

We can describe the desired CoM adjustment as a quadratic objective function of the step durations,

$$J_{\parallel} = \left\|\Delta x_{\parallel} - \nabla f_{\parallel}\Delta\mathbf{T}\right\|^2_{Q_{\parallel}}. \tag{4.18}$$

We prefer to define this as an objective instead of an equality constraint, as this formulation prevents the problem from being over-constrained if limits are placed on $\Delta\mathbf{T}$. We can then define additional quadratic objective function, such as the minimization of perpendicular CoM adjustments,

$$J_{\perp} = \|\nabla f_{\perp}\Delta\mathbf{T}\|^2_{Q_{\perp}}, \tag{4.19}$$

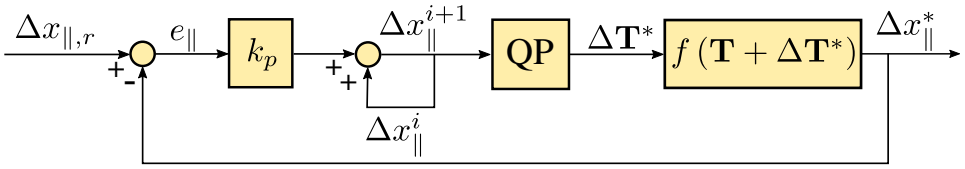


Figure 4.6: Block diagram of the feedback loop for calculating the timing adjustment. The loop is performed until the desired adjustment is achieved, with a feedback based on the achieved adjustment in each iteration.

and the minimization of the total timing adjustment,

$$J_T = \|\Delta \mathbf{T}\|^2_{\mathbf{R}_T}\,. \tag{4.20}$$

Lastly, we can define a cost function to minimize the timing symmetry between the duration of the beginning of each walking phase, $T_{\text{ini},i}$, and the end of each walking phase, $T_{\text{end},i}$,

$$J_\alpha = \|\Delta \mathbf{T}_{\text{ini}} - \Delta \mathbf{T}_{\text{end}},\|^2_{R_\alpha}\,, \tag{4.21}$$

which yields ICP plans with more desirable CMP trajectories when using the planning approach from [42]. The total cost function is then defined as

$$J = J_\parallel + J_\perp + J_T + J_\alpha, \tag{4.22}$$

with a corresponding optimal solution

$$\Delta \mathbf{T}^* = \text{argmin}\ J(\Delta x_\parallel, \Delta \mathbf{T}). \tag{4.23}$$

However, as this algorithm ignores the higher order terms in Equation 4.17, the resulting CoM adjustment from $\Delta \mathbf{T}^*$ are not likely to be equivalent to the desired adjustment, $x_\parallel \neq f_\parallel(\mathbf{T} + \Delta \mathbf{T}^*)$. This means that the QP is not likely to find an adjustment satisfying the kinematic constraints on the first iteration. To compensate for this, we can define an iterative feedback-based loop

$$\Delta x_\parallel^{i+1} = \Delta x_\parallel^i + k_p e_\parallel, \tag{4.24}$$

where $e_\parallel = \Delta x_{\parallel,r} - \Delta x_\parallel^*$ is the error between the achieved adjustment using the QP solution and the original desired CoM adjustment, $\Delta x_\parallel^i$ is the desired adjustment used in the current solver iteration, and $\Delta x_\parallel^{i+1}$ is the desired adjustment for the next iteration. This feedback loop is repeated until the error $e_\parallel$ is less than a fixed amount, $|e_\parallel| < \epsilon$, at which the loop is terminated. At the end of each iteration, the objective adjustment for next iteration, $\Delta x_\parallel^{i+1}$, is computed. The loop can also be terminated once a specified maximum number of iterations is exceeded.

4.5 Results

The algorithm was implemented in the IHMC Open Source Software system, using the Simulation Construction Set. To solve the QP, a custom implementation of the Goldfarb-Idnani solver was used. Balance is controlled when simulating the Atlas humanoid using the momentum-based whole-body controller outlined in [73].

A simulation of walking using 5.0s, 0.6m steps with and without using this algorithm is shown in Figure 4.5. As can be seen in the top row, using the fixed provided timing without adjustment requires significant bending of the knee for the step to be reached. The bottom row illustrates walking using the presented algorithm. The step durations are modified to require only 0.4rad of knee-bend, producing a much more natural, dynamic gait. The necessary timing adjustments are shown in Figure 4.7, and are discussed in detail later.

The resulting timing adjustments calculated using the algorithm for three different step lengths to achieve different degrees of knee bend are shown in Figure 4.7. Here, we compare the timing adjustment results for relatively slow steps, a 2.5s swing duration and 2.5s transfer duration when allowing a differing degree of knee-bend. For short (0.2m) steps, the motion results in approximately 0.45rad of bend at the knee using the original, slow 5.0s step duration. If we specify that we would like the legs to be straighter, i.e. less knee-bend, the upcoming transfer duration is decreased until the knees are only bent by the specified amount, while the other durations are unchanged. The optimization prefers to use the upcoming transfer duration, as it is the most effective means to adjust the CoM position, as illustrated in Figure 4.2.

For medium length (0.4m) steps, as shown in the middle plot of Figure 4.7, the step requires significantly more knee-bend at the original 5s duration than the short step length, needing approximately 0.8rad of bend. The optimization again primarily utilizes the upcoming transfer duration. As we placed constraints on the minimum durations for the different transfer segments, however, the current swing duration is also somewhat adjusted in the most extreme cases, as this offers slight modifications of the CoM positions.

For longer (0.6m) steps, as shown in the bottom plot of Figure 4.7, the user specified step duration of 5s requires a large amount of knee-bend, greater than 1.2rad. By trying to minimize knee bend, the upcoming transfer duration is made as small as possible. The current swing duration is also increased as the maximum knee bend is decreased in an attempt to achieve the desired amount of CoM adjustment.

It is worth noting that, for all step lengths, the current transfer duration has virtually no effect on the final CoM position when using the ICP planner outlined in [42]. Additionally, in practical implementation, allowing the current swing duration to increase provides relatively little benefit, with almost all effective adjustments coming from the upcoming transfer duration.

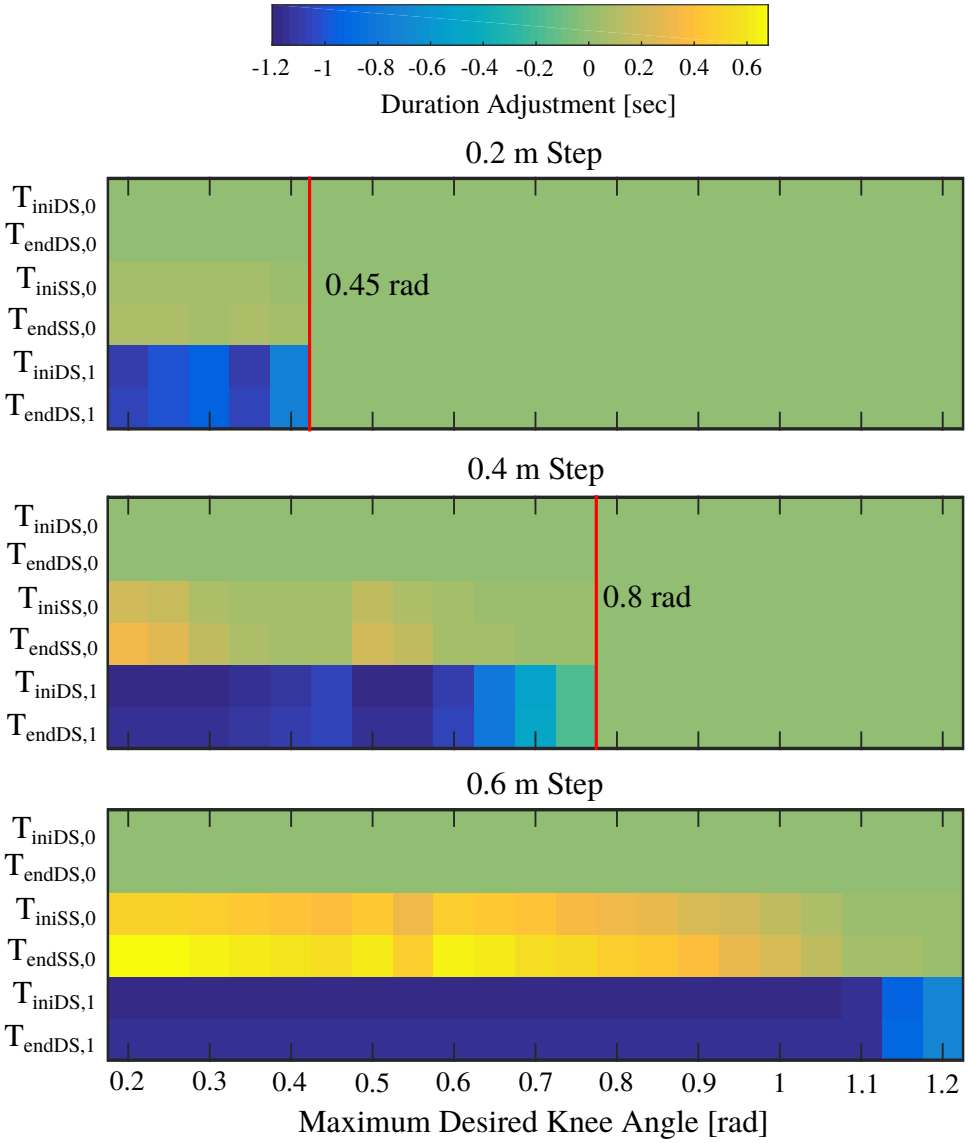


Figure 4.7: Timing adjustments while taking 0.2m, 0.4m, and 0.6m steps.

4.6 Conclusion

Using the presented approach, we were able to modify existing ICP plans to produce walking gaits that require minimal knee bend. Traditional walking approaches utilize highly bent knees, which has been somewhat addressed in other works by attempting to straighten the legs to their maximum feasible length while walking. This, however, overlooks the kinematic constraints that fixed step timing places on the CoM height and corresponding knee angles. This work represents, to our knowledge, the first study of the effects of the step duration in the resulting dynamic plans and their corresponding CoM heights.

We presented an approach that utilizes a quadratic program optimization scheme to compute

the necessary step timing adjustments to produce plans that require a specified amount of knee bend, while remaining kinematically feasible. This algorithm is then illustrated to successfully adjust the ICP plan timing to reduce the required knee bend to the desired amount. As robots move towards walking with straighter legs and more natural, efficient gaits, the effects of step timing will become increasingly important, with this work representing a critical first step towards achieving these motions.

In future work, we plan to utilize this algorithm along with advanced CoM height control techniques to enable humanoid robots to efficiently walk with straight legs using momentum-based whole-body control frameworks. We also plan to extend this work to account for kinematic effects of toe-off and pelvis orientation changes on reachability.

Chapter 5

Straight-Leg Walking Through Underconstrained Whole-Body Control

We present an approach for achieving a natural, efficient gait on bipedal robots using straightened legs and toe-off. Our algorithm avoids complex height planning by allowing a whole-body controller to determine the straightest possible leg configuration at run-time. The controller solutions are biased towards a straight leg configuration by projecting leg joint angle objectives into the null-space of the other quadratic program motion objectives. To allow the legs to remain straight throughout the gait, toe-off was utilized to increase the kinematic reachability of the legs. The toe-off motion is achieved through underconstraining the foot position, allowing it to emerge naturally. We applied this approach of under-specifying the motion objectives to the Atlas humanoid, allowing it to walk over a variety of terrain. We present both experimental and simulation results and discuss performance limitations and potential improvements.

With few exceptions, nearly all bipedal robot walking gaits utilize highly bent knees, walking with an almost "squatted" motion. Not only is this highly unnatural, it results in significant increase in power consumption at the knee [22] compared to humans [132]. Counter to this, passive-dynamic walkers, rely on using only the natural dynamics of walking and have the fundamental characteristic of walking with straight legs [14]. This offers energetic benefits, requiring far less torque, and allowing the swing leg to behave like a double pendulum. Walking with straighter legs also increases the overall ground clearance of the robot, allowing it to step over larger objects and avoid collisions, as well as decreasing the required range of motion and actuator velocities for the same walking speed. A method for utilizing the natural dynamics of walking to achieve this straight legged gait is needed to allow these benefits to be realized by actively powered robots.

From a control perspective, walking with straight legs poses significant challenges. First, by

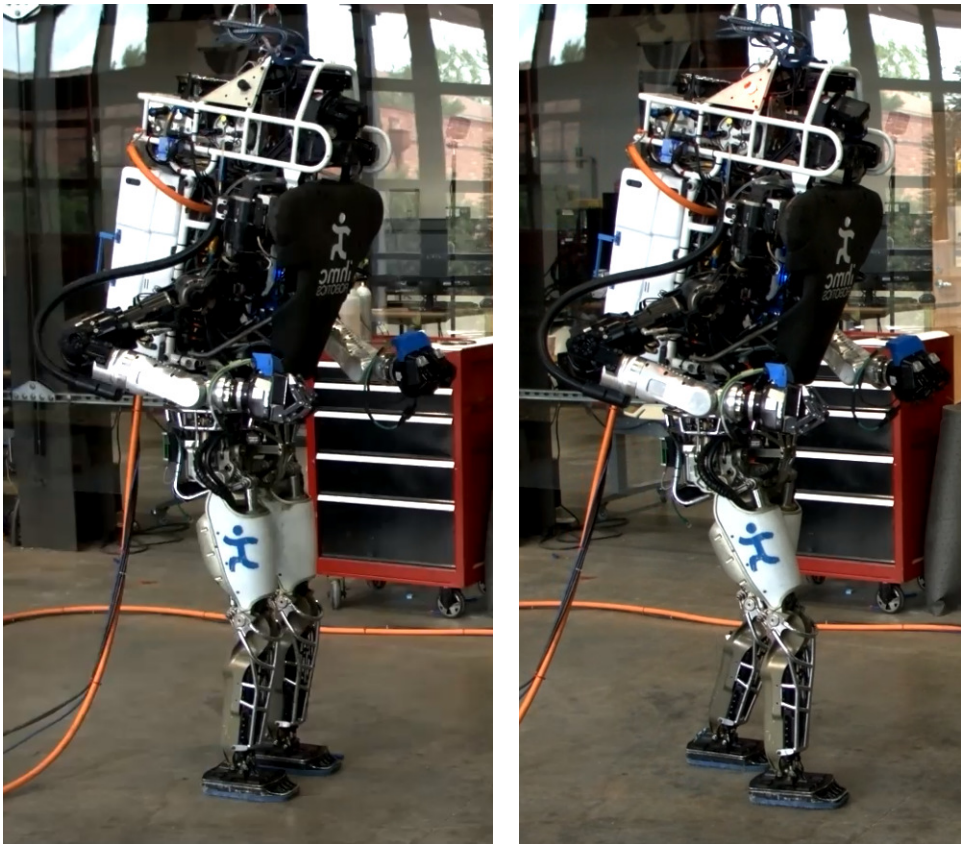


Figure 5.1: Images of the Atlas robot standing and stepping with straight legs.

allowing height variations, some models such as the Linear Inverted Pendulum (LIP) lose accuracy, as they assume that the center of mass (CoM) is moving through a plane at constant height [13]. Second, when the legs of the robot are completely straightened, the ground reaction forces are primarily dependent on the gravity vector and ankle torques, with the control authority from the knee effectively removed. Last, the straightened leg introduces a singularity in the Jacobian used by inverse-kinematics and inverse-dynamics based approaches. While there are mechanisms to provide greater control authority, such as step adjustment and angular momentum, and handle Jacobian singularities, the greatest challenge comes from the now-nonlinear dynamics. These can be solved linearly using a predefined height trajectory, and then controlled using standard zero-moment point (ZMP) [137, 30, 116] or divergent component of motion (DCM) [126, 38] approaches. Also, the height control can be decoupled from the planar control, relying on the robustness of the planar controller. To walk with straight legs, though, both approaches require complex approaches for computing an appropriate height trajectory.

Several works have attempted to address this complicated planning problem. The general objective becomes to design trajectories that result in the robot's legs being as straight as possible without violating the kinematics or the centroidal dynamics. In [134], the authors presented an approach for generating straightened-knee trajectories by modeling the legs as spring dampers to compute the CoM height after generating horizontal trajectories using

model predictive control (MPC) of the ZMP. Alternatively, [112] used linear differential inclusion to incorporate these height trajectories directly into the MPC, resulting in relatively natural, cyclic motions of the CoM. In [135], the authors also included CoM height in their ZMP-based quadratic program, but required sequential quadratic programming to solve the resulting problem's quadratic constraints. In all cases, after planning, the desired CoM trajectories can be tracked in a momentum-based control framework, as proposed in [138].

Instead of using complex CoM height planning to walk with straight legs, we propose a new approach that relies the whole-body control framework's, ability to generate torques given a desired set of motion tasks [73]. We propose directly address the objective of straightening the legs by biasing the whole-body controller solutions towards those that straighten the legs as much as possible, rather than through setting a desired CoM height. As noted in [139], the whole-body controller can resolve the kinematic and dynamic constraints on the system at runtime. To accomplish this, we propose underconstraining the whole-body controller by not specifying a desired force in the vertical direction. Instead, we can project a desired straightening leg motion into the null-space of the other objectives. The robot then will attempt to make its legs as straight as possible for any desired motion, as opposed to trying to achieved a specified CoM height trajectory. This becomes more similar to the passive-dynamic walking approach, allowing the dynamics of the system to determine the robot height rather than a preplanned trajectory.

Straight leg walking requires several additional elements for proper execution. Toe-off is an integral part of natural walking, it increases the number of reachable footholds that do not require significant knee bend by increasing the length of the stance leg. Toe-off is typically achieved with whole-body controllers by prescribing some toe-off motion in the foot [71, 47]. Instead, we propose to underconstrain the foot motion, as with our approach for straight leg walking. By under-specifying the desired motions of the foot, the toe-off motion becomes an emergent behavior of the gait. As the transition to toe-off is state dependent, we define conditions for when the robot should use a toe-off motion. Additionally, when walking with straight legs, the desired step position is often near the edge of the leg's workspace, with terrain height uncertainties creating significant challenges. In this work, we propose several mechanisms to compensate for this, as well.

In our approach, we utilize a step adjustment strategy based on that presented in [104] to increase the control authority of the robot when walking with straight legs. We also guarantee the feasibility of the dynamic trajectories for walking with straight legs using the approach presented in [105]. This is executed using the instantaneous capture point (ICP) planner in a whole-body controller framework, as presented in **??**. The approach for achieving straight leg walking is described in section 5.3, with the toe-off approach outlined in section 5.5. Finally, we present both physical and simulation experiments performed both with the Atlas robot in section 5.6.

5.1 Walking Control

To allow the balance control problem to be tractable for real-time control, reduced order models are typically employed. The LIP treats the robot as a point mass at the end of a pendulum, whose base is the ZMP [13]. The ICP was introduced as an extension of the LIP, and is a state transformation of the CoM defined as

$$\boldsymbol{\xi} = \mathbf{x} + \frac{1}{\omega}\dot{\mathbf{x}}, \tag{5.1}$$

where $\mathbf{x}$ is the center of mass position and $\omega = \sqrt{g/z_{\text{CoM}}}$ is the inverted pendulum natural frequency. The ICP dynamics are then defined as

$$\dot{\boldsymbol{\xi}} = \omega\left(\boldsymbol{\xi} - \mathbf{r}_{\text{CMP}}\right), \tag{5.2}$$

where $\mathbf{r}_{\text{CMP}}$ is the Centroidal Moment Pivot [130], which is used to control the ICP and encodes the system's ground reaction forces and angular momentum rate. In this work, we use the ICP planning approach presented in [42].

As it is based on the LIP, the ICP typically assumes that that the CoM moves in a constant height above the ground. However, ω can be treated as a design variable, rather than strictly defined by the LIP length, as noted in [37]. The enhanced centroidal moment pivot (eCMP) is located at the intersection of the CMP line with the ICP control plane, which is located $z = \omega^2/g$ distance below the CoM [37]. The eCMP in this case is used to control the ICP dynamics, rather than the CMP. The CMP is then typically located at the intersection of the CMP line and the ground plane, as shown in Figure 5.2.

As discussed in [139], by allowing the CoM height to vary, the CMP deviates from the nominal eCMP location. From the perspective of the ICP, height variations from the nominal height $z_{\text{nom}} = \omega^2/g$ will cause the CMP to deviate from the eCMP, with a relationship defined by

$$\frac{z_{\text{CoM}}}{z_{\text{nom}}}\left(\mathbf{x} - \mathbf{r}_{\text{eCMP}}\right) = \mathbf{x} - \mathbf{r}_{\text{CMP}}. \tag{5.3}$$

Using this relationship, we can perform a simple analysis to determine the maximum amount of deviation to be expected when walking using a simple inverted pendulum model with instantaneous exchanges in support, as shown in Figure 5.3. The maximum CMP error occurs when the CoM is the greatest distance from the desired eCMP, at the point of double support exchange. For this example, if the stride length is l, then this distance is $0.5l$. Evaluating Equation 5.3, assuming that the nominal CoM height is the pendulum length, the CMP error is

$$\mathbf{r}_{\text{CMP}} - \mathbf{r}_{\text{eCMP}} = 0.5l\left(1 - \frac{\sqrt{z_{\text{nom}}^2 - 0.25l^2}}{z_{\text{nom}}}\right). \tag{5.4}$$

For approximately human parameters of a pendulum height of 1m and a stride length of

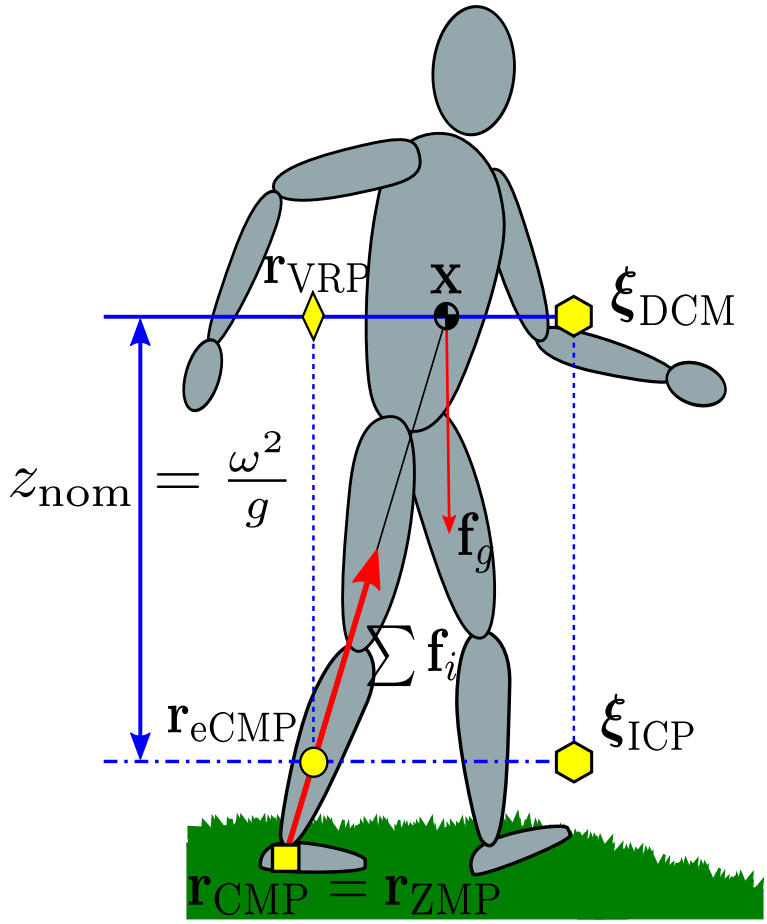


Figure 5.2: Diagram showing the relationship between important ground reaction points. The ICP encodes the CoM position and velocity, and is controlled by the eCMP. The CMP is lies at the intersection between of the line passing through the CoM and the eCMP and the ground. For a more detailed definition of the DCM and VRP, see [37]

0.75m, this yields a CMP deviation of 2.7cm from the nominal value. While this is not ideal, a standard ICP feedback controller is easily robust enough to reject the resulting tracking errors with proper planning.

5.2 Whole-Body Control

Momentum-based control frameworks have been quite effective in implementing compliant, force based control. Whole-body control represents one of the most common formulations, and gained significant prominence during the DARPA Robotics Challenge. Almost all in the DRC Finals utilize a quadratic program (QP) to optimize a cost function to resolve multiple motion tasks at every controller time step [115, 47, 73, 72]. The QP formulation found in

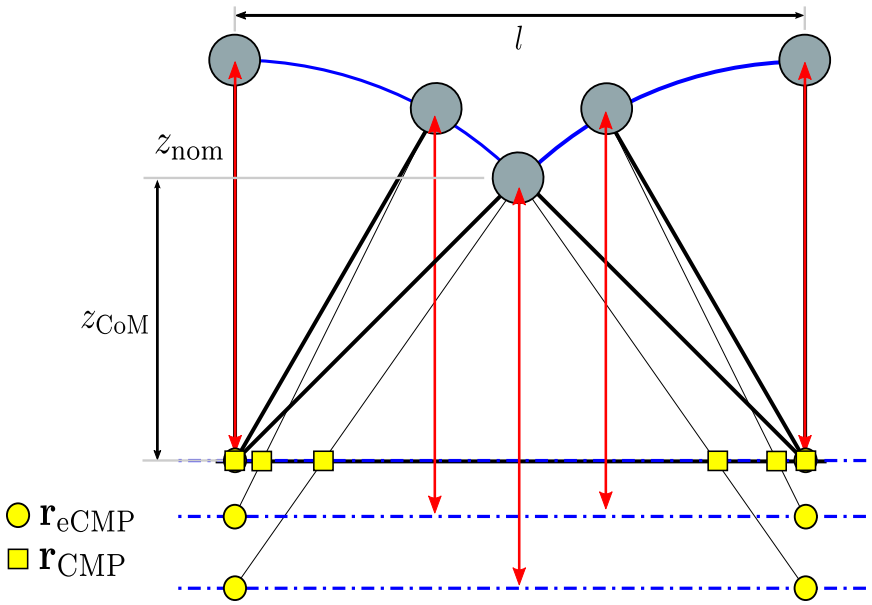


Figure 5.3: Diagram showing the CMP error as the pendulum height changes while walking.

this work is introduced in [73], and is as follows:

$$\begin{array}{ll} \min_{\dot{\mathbf{v}}_d, \boldsymbol{\rho}} & J_{\dot{\mathbf{h}}_d} + J_{\mathbf{J}} + J_{\boldsymbol{\rho}} + J_{\dot{\mathbf{v}}_d} \\ \text{s.t.} & \mathbf{A}\dot{\mathbf{v}}_d + \dot{\mathbf{A}}\mathbf{v} = \mathbf{W}_g + \mathbf{B}_{\text{CoM}}\boldsymbol{\rho} + \sum_i \mathbf{W}_{\text{ext},i}, \\ & \boldsymbol{\rho}_{\min} \leq \boldsymbol{\rho}, \\ & \dot{\mathbf{v}}_{\min} \leq \dot{\mathbf{v}}_d \leq \dot{\mathbf{v}}_{\max}. \end{array} \tag{5.5}$$

The terms of the objective function are defined as

$$\begin{array}{ll} \text{Momentum Objective:} & J_{\dot{\mathbf{h}}_d} = \left\| \mathbf{P}_{\dot{\mathbf{h}}} \left(\mathbf{A}\dot{\mathbf{v}}_d - \mathbf{b} \right) \right\|^2, \\ \text{Motion Objective:} & J_{\mathbf{J}} = \left\| \mathbf{P}_{\mathbf{J}} \left(\mathbf{J}\dot{\mathbf{v}}_d - \mathbf{p} \right) \right\|^2, \\ \text{Contact Force Cost:} & J_{\boldsymbol{\rho}} = \left\| \mathbf{P}_{\boldsymbol{\rho}}\boldsymbol{\rho} \right\|^2, \\ \text{Joint Acceleration Cost:} & J_{\dot{\mathbf{v}}_d} = \left\| \mathbf{P}_{\dot{\mathbf{v}}_d}\dot{\mathbf{v}}_d \right\|^2, \end{array}$$

where:

- $\dot{\mathbf{v}}_d$ are the desired generalized joint accelerations and $\boldsymbol{\rho}$ consists of the generalized contact forces [73].
- $\mathbf{A}$ is the centroidal momentum matrix and the convective term $\mathbf{b} := \dot{\mathbf{h}}_d - \dot{\mathbf{A}}\mathbf{v}$, where $\dot{\mathbf{h}}_d$ is the desired rate of change of momentum and $\mathbf{v}$ is the joint velocities.
- $\mathbf{J} = \begin{bmatrix} \mathbf{J}_1^T & \dots & \mathbf{J}_k^T \end{bmatrix}^T$ and $\mathbf{p} = \begin{bmatrix} \mathbf{p}_1^T & \dots & \mathbf{p}_k^T \end{bmatrix}^T$ are the concatenated Jacobian matrices and respective objective vectors for each of the k desired motions.
- $\mathbf{B}_{\text{CoM}}$ is the Jacobian matrix from the generalized contact force frame to the centroidal frame.

- $\mathbf{W}_{\text{ext},i}$ are the i external wrenches acting on the robot.
- $\mathbf{W}_g$ is the gravitational wrench.
- $\boldsymbol{\rho}_{\min} \geq 0$ is the lower bound on $\boldsymbol{\rho}$, used to enforce contact unilaterality.
- $\dot{\mathbf{v}}_{\min}$ and $\dot{\mathbf{v}}_{\max}$ are bounds on the joint accelerations, used to enforce joint angle limits.
- $\mathbf{Q}_{\dot{\mathbf{h}}}, \mathbf{Q}_{\mathbf{J}}, \mathbf{Q}_{\boldsymbol{\rho}}$, and $\mathbf{Q}_{\dot{\mathbf{v}}_d}$ are positive definite cost function weighting matrices, where $\mathbf{Q}_{(\cdot)} = \mathbf{P}_{(\cdot)}^T \mathbf{P}_{(\cdot)}$.

In this work, we choose to incorporate the linear part of the momentum objective $\dot{\mathbf{h}}_d$ by applying a selection matrix to the momentum objective,

$$J_{\dot{\mathbf{h}}_d} = \left\| \mathbf{S} \left(\mathbf{A}\dot{\mathbf{v}}_d - \mathbf{b} \right) \right\|^2_{\mathbf{Q}_{\dot{\mathbf{h}}}}, \tag{5.6}$$

where

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.7}$$

selecting only the linear momentum rate of change, $\dot{\mathbf{l}}$, from $\dot{\mathbf{h}} = \begin{bmatrix} \dot{\mathbf{k}}^T & \dot{\mathbf{l}}^T \end{bmatrix}^T$. This leaves the angular momentum rate of change $\dot{\mathbf{k}}$ unconstrained, free to be determined by the optimization. This allows angular momentum to be generated by natural motions, such as the swing foot, without requiring it be canceled via other motions.

5.3 Straight Leg Walking Control

Instead of controlling the CoM height directly, we propose controlling it inside the task null-space using desired leg configurations. The general objective of advanced CoM height planning is to keep the legs as near straight as possible to avoid higher torques at the knee. Instead of trying to straighten the legs through complicated planning, we instead bias the whole-body controller solution towards a desired configuration. To do so, we modify the selection matrix in Equation 5.7 to use only the x-y momentum rate of change,

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{5.8}$$

leaving the vertical force of the robot unconstrained. This works well within the instantaneous capture point (ICP) based framework used in [73], which required a separate controller on either the CoM or pelvis height for the vertical momentum rate of change. As typical

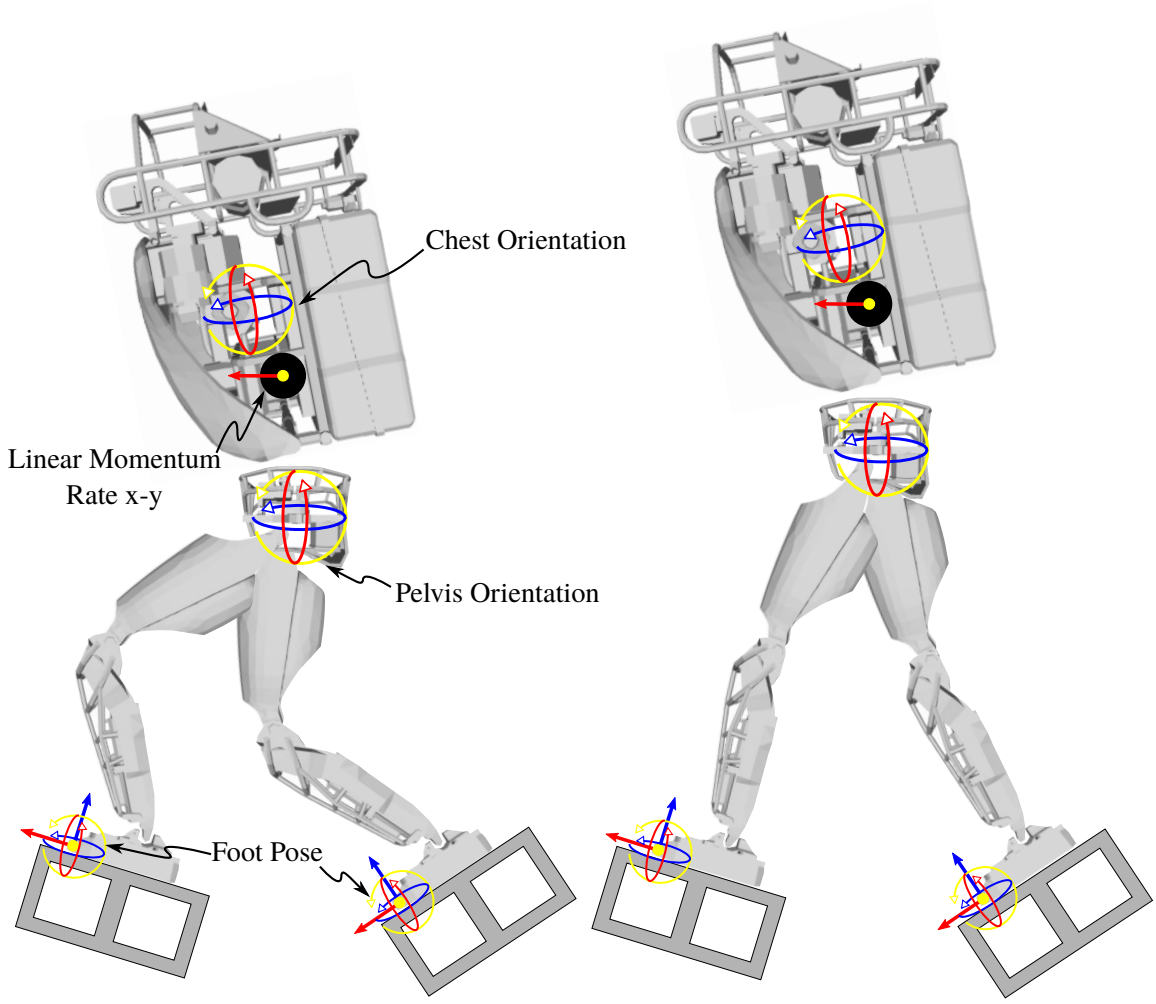


Figure 5.4: When the vertical momentum rate of change is not specified, there are many potential solution motions contained in the overall task Jacobian. This shows two possible ones, one with bent legs, one with with straight.

motion inputs for $\mathbf{p}$ consist of tasks such as pelvis angular acceleration, chest angular acceleration, arm joint acceleration, and swing foot linear and angular acceleration, the resulting QP has a large null-space, with many potential solutions satisfying these objectives, as illustrated in Figure 5.4.

To control the leg configuration without violating any of the higher level motion tasks, we can project desired joint angles, which we will refer to as the *privileged configuration*, into the null-space of the QP. The privileged configuration, $\mathbf{q}_p$, can be then used to compute the privileged joint accelerations using a simple feedback control law

$$\dot{\mathbf{v}}_p = \mathbf{k}_p \left(\mathbf{q}_p - \mathbf{q} \right) + \mathbf{k}_d \mathbf{v}, \tag{5.9}$$

where $\mathbf{q}$ is the current joint position. These privileged accelerations can then be projected into the null-space of the QP as an additional quadratic motion objective

$$J_p = \left\| \left(\mathbf{I} - \mathbf{J}_{\text{task}}^{+} \mathbf{J}_{\text{task}} \right) \dot{\mathbf{v}}_p \right\|_{\mathbf{Q}_p}^2 , \tag{5.10}$$

where $\mathbf{J}_{\text{task}} = \begin{bmatrix} (\mathbf{SA})^T & \mathbf{J}^T \end{bmatrix}^T$ is the total task Jacobian that maps all the desired inputs to the QP to joint acceleration space, and $(\cdot)^+$ is the pseudo-inverse operator.

This approach of applying privileged configurations to bias QP solutions has been well used to help with singularity escape. The effect is that, when a joint is at a singularity, there is a non-empty null-space, allowing the privileged configuration to provide an acceleration in the correct direction. Instead of using it as a mechanism to help with singularity escape, however, we use it to bias the general robot behavior, encouraging the solutions to always use as straight of knees in the support leg as possible.

5.3.1 Leg Configuration Selection

To effectively control the leg configuration, we introduce a state machine that sets the privileged configuration for each leg, shown in Figure 5.5. This is used to allow the leg to bend at the desired phases in the gait cycle. On touchdown, the new support transitions to the "Straighten" state, where the privileged configuration transitions to the straight configuration over a period of defined time. At this point, the leg is automatically transitioned into the "Straight" state. Once the support leg is partway through the swing phase, the configuration is changed to "Collapsed", allowing the leg to slightly bend. This allows the swing foot to track by moving the workspace and assists with toe-off. The trailing leg then remains in the collapsed state during transfer. At the beginning of swing, the leg is switched to the "Bent" state which helps pick the foot up off the ground and escape the knee singularity. Partway during the swing phase, it is then switched to the "Extend" state, which extends the leg outward in preparation for touchdown. On touchdown, the leg is then transitioned back to the "Straighten" state.

5.4 Swing Foot Control

Controlling the swing foot when walking with straight legs introduces several challenges. In this section, we introduce our approach for dealing with some of the challenges.

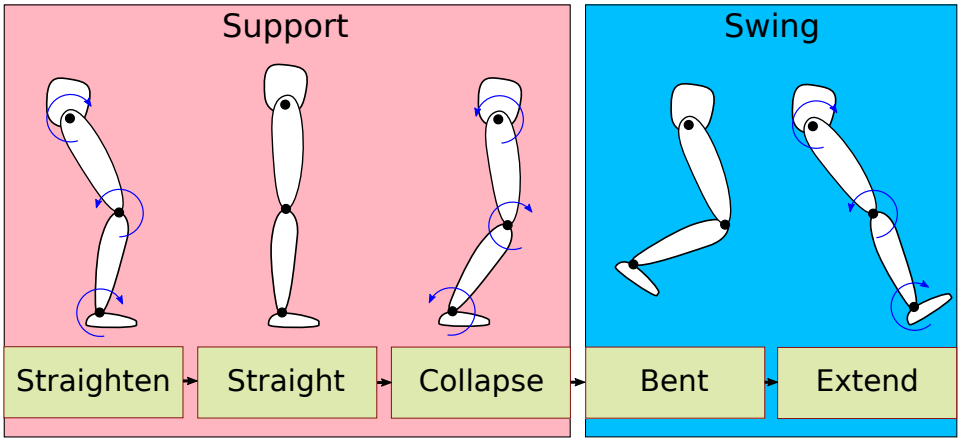


Figure 5.5: State machine for the desired leg configuration angle, which straightens and collapses the leg based on the phase of the walking gait.

5.4.1 Foot Lift Off

Picking the swing foot up off the ground does not pose significant challenges in typical control frameworks. However, when the CoM height is left uncontrolled, there become two possible solutions to achieve this motion, one of which is highly undesirable. The first, standard approach is to use the leg joints to bend the swing leg upward, lifting the foot off the ground. The second, new solution is to leave the leg joint positions fixed while raising the CoM position. This is highly undesirable, as it has the effect of requiring large ground reaction forces to raise the entire mass of the robot to simply pick up the foot.

To encourage the whole-body controller to use the swing leg joints over lifting the center of mass to pick the foot up, we can apply a scaling factor to the task Jacobian. For example, if we examine the objective for the swing foot, given by

$$\mathbf{J}\dot{\mathbf{v}} - \mathbf{b}, \tag{5.11}$$

the Jacobian is for a six degree of freedom floating base attached to a six degree of freedom leg, making the Jacobian $\mathbf{J} \in \mathbb{R}^{6\times 12}$. What we would like is for the solution to prefer using the leg joints over the floating-base joints. To do this, we can scale the effect of the floating-base joints in the Jacobian,

$$\mathbf{J}_{\text{scaled}}\dot{\mathbf{v}} - \mathbf{b}, \tag{5.12}$$

where

$$\mathbf{J}_{\text{scaled}} = \mathbf{J} \begin{bmatrix} \alpha\mathbf{I}_{6\times 6} & \mathbf{0}_{6\times 6} \\ \mathbf{0}_{6\times 6} & \mathbf{I}_{6\times 6} \end{bmatrix}, \tag{5.13}$$

with α being a scaling factor between 0 and 1. In our experiments, a scaling factor of $\alpha = 0.1$ is applied at the beginning of swing, and is smoothly transitioned to 1 during swing. This has the effect of forcing the optimization to prefer using the leg joints to pick the foot up.

5.4.2 Improved Foot Touchdown

When walking, it is generally assumed that the swing foot will strike the ground at the correct time. While the upcoming foothold is almost always in the swing leg's workspace when the knees are heavily bent, this may not be the case when the stance leg is straight. If the ground plane is lower than the predicted footstep, the foot will not have come into contact when planned, and will most likely be at the edge of the workspace This result in an exponentially increasing ICP error with respect to time. To maintain stable walking, then, the foot must be set down as quickly as possible after its expected touchdown to make the gait robust to height uncertainties. This is similar to starting to shift weight onto the swing leg, whether it's in contact or not, trusting that it will hit the ground.

A standard approach for handling this is to command a constant downward velocity at the end of the swing foot trajectory. In addition, we linearly increase the weight on the swing foot spatial accelerations in the whole-body controller with respect to the time past the expected swing touchdown. This increases the importance of continuing the downward motion of the foot. We also decrease the weight on the pelvis orientation with respect to this exceeded time, allowing the robot to rotate the pelvis to achieve the desired swing foot motion, similar to how people rotate their pelvis to increase their leg's workspace.

5.5 Toe-Off

Toe-off is an essential part of human-like bipedal walking. Besides the metabolic savings it provides through capitalizing on the leg's muscle-tendon for energy storage, it is kinematically beneficial, allowing the trailing leg to be extended and the leading leg to bend less. Toe-off also helps address range of motion limitations, such as when stepping down. However, when to toe-off is unclear, as decreasing the support polygon size limits the control authority of the robot, as shown in Figure 5.6. Additionally, how to achieve toe-off poses many challenges.

5.5.1 Toe-Off Criteria

To determine when to transition to toe-off, we establish a set of criteria that must be satisfied. The support polygon in toe-off is defined using a single toe-off point in the foot, as shown in Figure 5.6. If moving to toe-off results in either the desired or current ICP being outside the support polygon, then we no longer have full controllability of the ICP. This is equivalent to saying that the robot must be 1-step capturable during swing and 0-step capturable during transfer after toe-off. The desired and current ICP must also be within a certain distance of the foothold. Due to torque limits and other uncertainties, the desired CMP may not be achieved. By increasing the distance of the ICP from the CMP, the effects of the inaccuracies are decreased, as shown in Figure 5.7, and are more likely to move in the direction of the

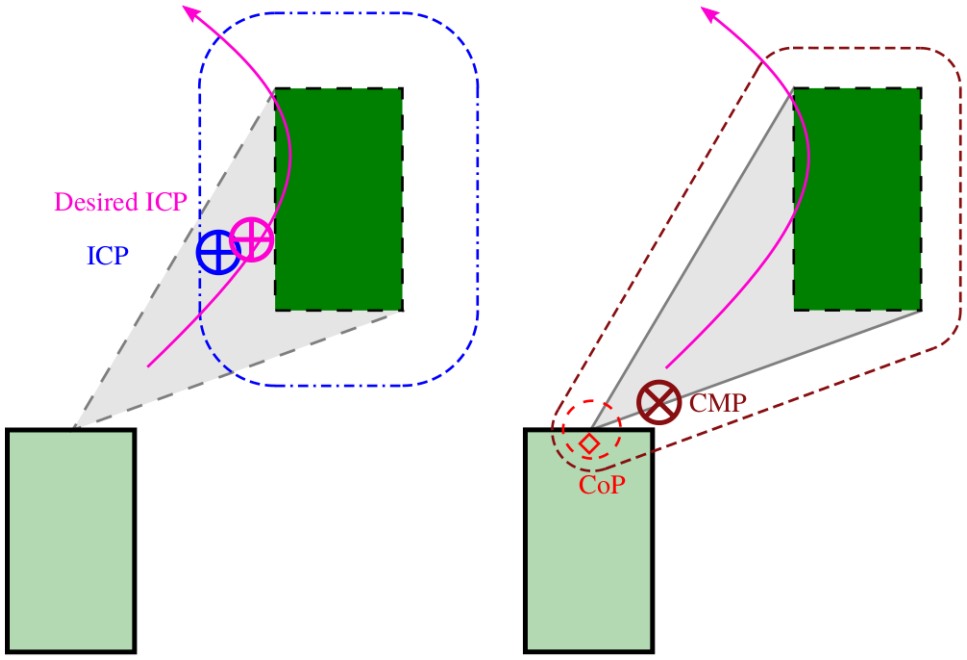


Figure 5.6: Criteria checks for toe-off. To start toe-off, both the desired ICP and estimated ICP locations must be inside the predicted support polygon (gray), as well as within a certain proximity of the desired foothold (the dotted blue line). The trailing foot CoP must be less than a certain distance from the toe-off point (the dotted red line), and the desired CMP location must be close to the predicted support polygon, the dark dotted red line.

foothold.

Additional toe-off criteria are related to the ground reaction forces. As changing the support polygon changes the constraints on the center of pressure (CoP), we require that the CoP in the support/trailing foot be within a certain proximity of the toe-off polygon, minimizing the change on toe-off. To minimize the amount of angular momentum, the distance from the centroidal momentum pivot (CMP) to the support polygon must also be minimized. These requirements are outlined in Figure 5.6.

5.5.2 Toe-Off Control

Unlike other approaches [71, 47], to control the foot during toe-off, we provide no direct control of the toe-off motion, that is, the pitching of the foot. Instead we leave this degree of freedom free and unspecified. This allows the controller to use the foot pitch to achieve other motion tasks, making toe-off motion an emergent behavior. For the single foot contact point in toe-off, we use the standard friction cone constraints, as shown in Figure 5.8. We found that using a single point resulted in better control than a contact line. To perform

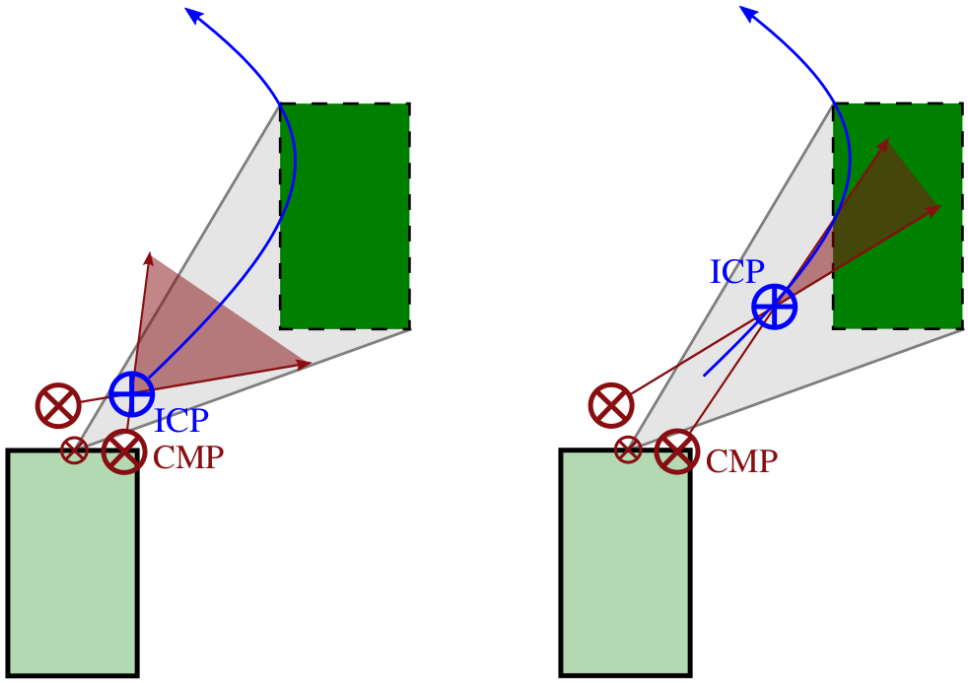


Figure 5.7: Figure showing the effects of ICP location on ICP control for proximity to upcoming foot. The closer the ICP location is to the stance foot, the more susceptible the ICP dynamics are to CMP location errors, as shown on the left. If the ICP is closer to the desired foothold, CMP errors affect the direction of the ICP dynamics less, going more in the direction of the foothold.

this control, we can break it into two, separate parts: position and orientation.

During toe-off, we do not want the toe-off contact point of the foot to slip. This can be achieved using a simple feedback controller to find linear accelerations to hold the point constant in the world frame. An orientation controller in the sole-frame of the foot can be used to maintain a constant roll and yaw of the foot throughout the toe-off motion. This leaves the pitch of the foot unconstrained and open in the null-space. Pitching of the foot about the toe-off point then occurs naturally during the toe-off state, as shown in Figure 5.8.

5.6 Results and Discussion

We implemented the proposed method on the Atlas robot, and conducted both simulations and physical experiments. Using the combination of step adjustment from [104], step time optimization from [105], and the presented strategies for straight leg walking and toe-off control, the robot was able to walk over both flat ground and varying terrain with straight legs. For the physical experiments presented here, a swing time of 1.0s and transfer time of

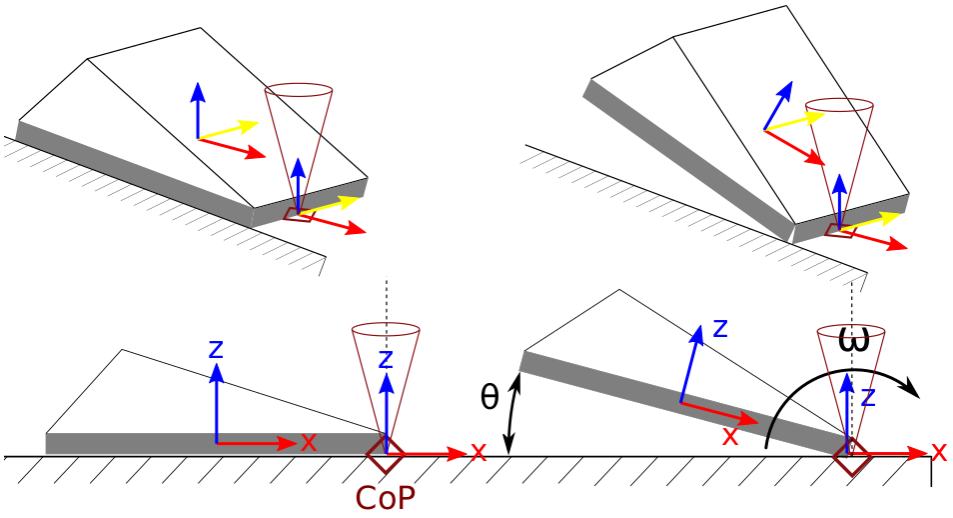


Figure 5.8: During toe-off, a single contact point is enabled on the front of the trailing foot. The pitch of the foot is then left uncontrolled, while the contact point is held constant in the world.

0.35s was used, while simulation used 0.60s and 0.25s, respectively. The speeds on hardware are limited by the pump curve of the hydraulic pump, leading to step durations nearly twice that of humans, limiting how dynamic the walking gait is. This, in turn, results in the robot requiring slightly more bent knees than otherwise possible.

Tests showing the ability for the controller to reject external disturbances when the robot is standing are shown in Figure 5.9. The robot is easily able to recover from pushes applied backward to the chest and sideways at the pelvis. The control authority in the transverse direction is much higher than the sagittal due to the wider support polygon. The CMP is quickly moved to allow the estimated ICP to return to the desired ICP. Some steady state error is present, due to unmodeled dynamics in the robot's actuators, such as stiction.

Figure 5.11 shows the resulting ICP and CoM trajectories for Atlas taking 0.35m steps over flat ground, for a walking speed of 0.26m/s. As can be seen, the robot tracks the desired ICP trajectory, albeit it with some error, resulting in a sinusoid-like CoM motion in the x-y plane, similar to that of human walking. By commanding the legs to straighten, the CoM height follows a path similar to that of the inverted pendulum. Some error occurs when the leading leg is allowed to straighten and overshoots at the beginning of swing. Unlike the true inverted pendulum, which has sharp changes in height as seen in Figure 5.3, the robot's CoM smoothly changes height during the transfer phase, shown in gray in Figure 5.11. Toe-off, which is indicated by the vertical blue line, is required for this exchange to occur during transfer. Without toe-off, the CoM height could not increase until swing, as it allows the trailing leg to extend further, while the leading leg compresses. By allowing toe-off and height variations, the resulting CoM height trajectory then resembles that observed in humans [140].

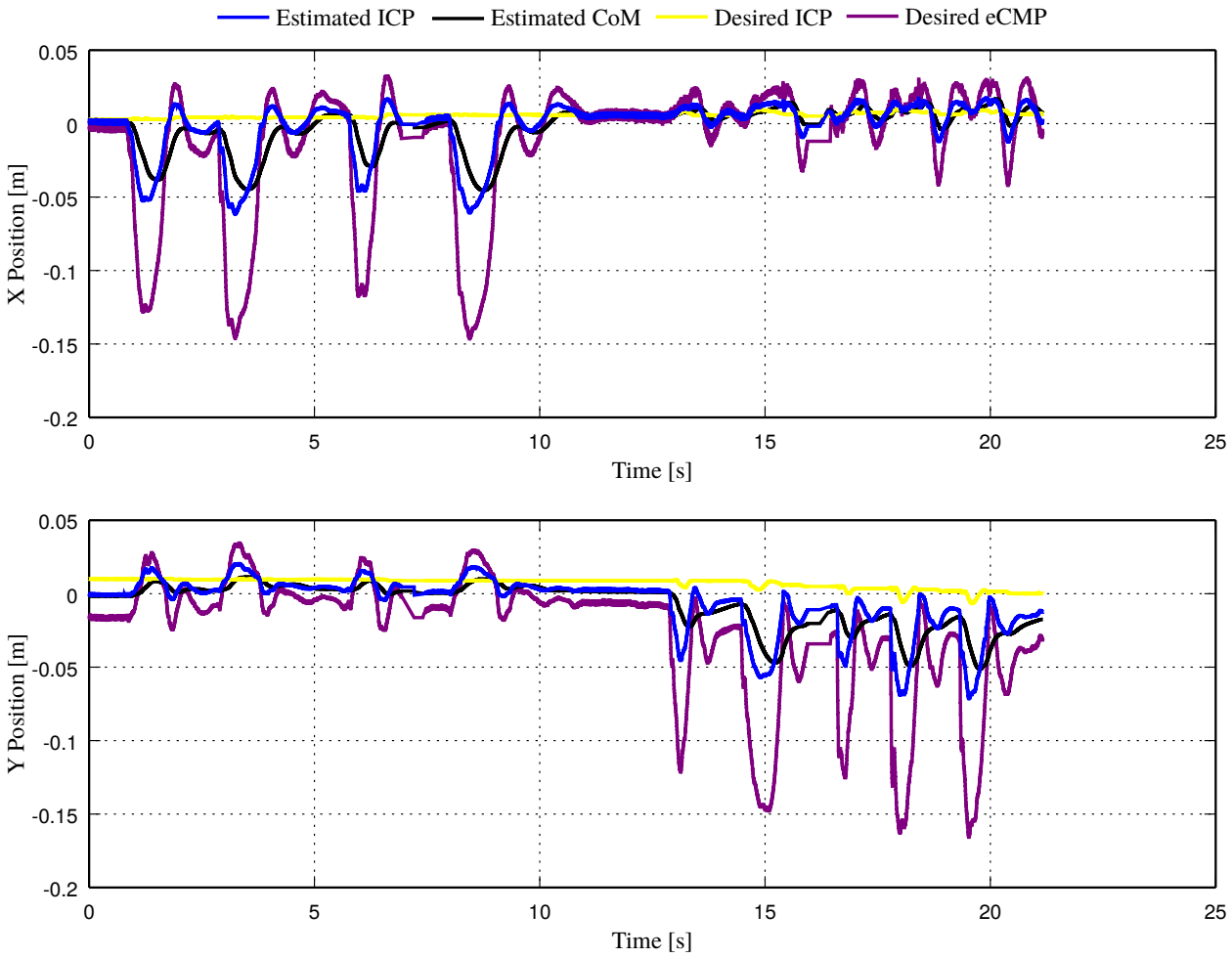


Figure 5.9: Data collected from Atlas when the robot is standing and external disturbances are applied. The first four pushes are applied pushing the robot backward on the chest. The last five pushes are applied to the left side of the pelvis.

We also tested the ability of the step adjustment algorithm to function when walking with straight legs. Figure 5.13 and Figure 5.14 show the effects of applying external forces to the robot when stepping in place in the lateral and forward directions, respectively. Figure 5.15 is another experiment conducted where a side push is applied to the robot while it is taking short steps forward. The robot is able to quickly adjust the desired footstep location, allowing it to smoothly recover from the large tracking error. For the forward walking one, the robot then smoothly continues walking after adjusting the upcoming plan. It is worth noting that after adjusting the foothold a significant amount of toe-off is used to move to the next step. The ability of the robot to handle external disturbances without falling is both a testament to the robustness of the proposed straight leg walking algorithm, and the step adjustment algorithms presented in previous chapters.

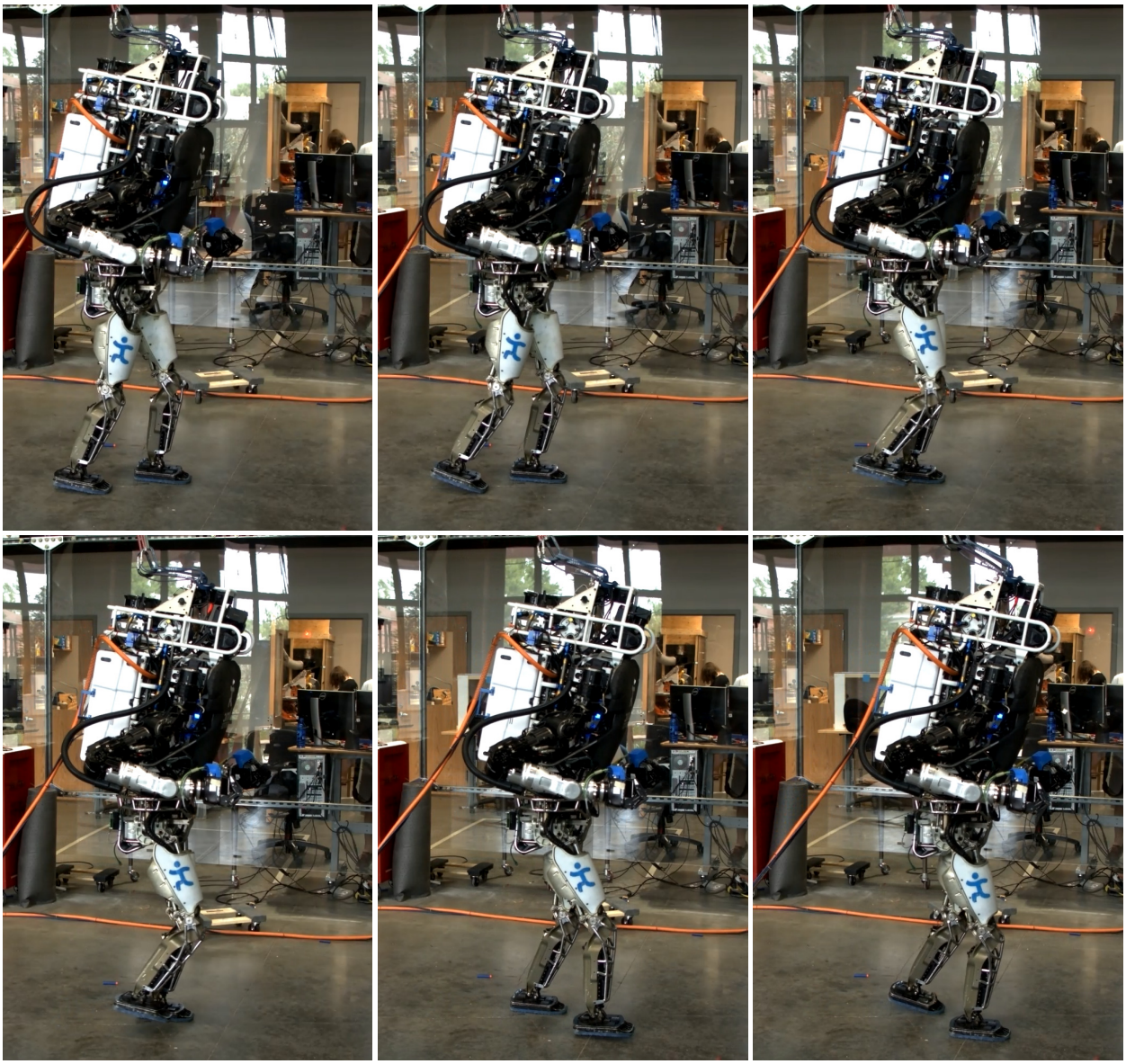


Figure 5.10: Atlas walking over flat ground using the proposed method for indirectly controlling the CoM height and toe-off.

Using the proposed approach, the robot was capable of walking over terrain of varying height. A simulation of Atlas walking up and down a set of stairs is shown in Figure 5.12, with each step 19.685cm (the maximum height allowed by the American's with Disabilities Act) high. While ascending the stairs posed little challenge for the controller, descending was slightly harder, as the desired swing foot position was often at the edge of the leg's workspace. To appropriately get the foot down to the next step, the robot had to collapse the stance leg

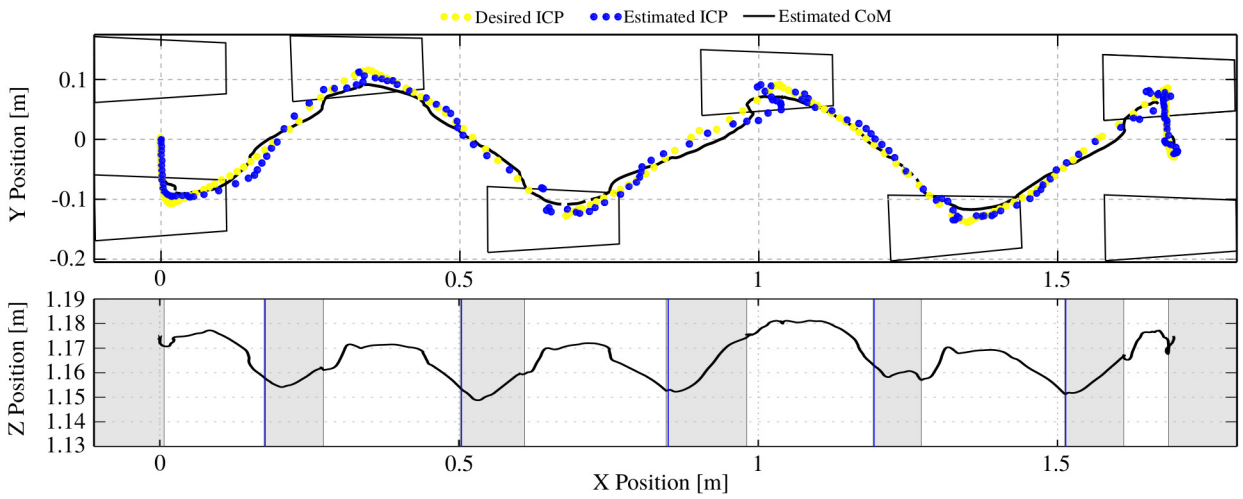


Figure 5.11: Data of Atlas walking over flat ground using the proposed method for indirectly controlling the CoM height and toe-off. The ICP is able to be tracked fairly well using the ICP control framework in [104]. The CoM height trajectory follows a path similar to that of the inverted pendulum, with velocity direction changes occurring in the transfer phase, shown in gray. Toe-off greatly improved this motion, and is indicated by the blue vertical line.

quickly, essentially dropping onto the upcoming step, similar to human. This was greatly assisted by our proposed approach in subsection 5.4.2, which incentivised the controller to collapse the stance leg and rotate the pelvis to track the desired swing foot position. The cyclic height trajectories emerged naturally, with the robot raising the CoM during transfer as much as possible, and then stepping up.

The proposed algorithm also was shown to work well on a variety of terrain during hardware experiments. As shown in Figure 5.16, Atlas was able to walk over cinder blocks of differing heights. We plan to further test the algorithm over other terrains and stairs to improve tuning, control, and transition criteria.

While the robot has not demonstrated the same level of balance as when walking with bent knees, this is expected, as the control authority is diminished through greatly reducing the effective actuation at the knee. Additionally, our balance algorithms require adequate control of the CoP and CMP locations, and accurate measurements of both the ICP position and velocity. While we are normally able to control the CoP to approximately 2cm accuracy, the control accuracy is reduced as the legs are straightened. As such, we set the desired knee angle to 0.3rad, which we found to be the minimum angle at which we had adequate CoP control for walking. The CoP inaccuracies start to impose limitations as the desired CMP is moved near the edge of the support polygon. This can then cause tipping about the edge of the foot, which further exacerbates state estimation problems faced in the ICP

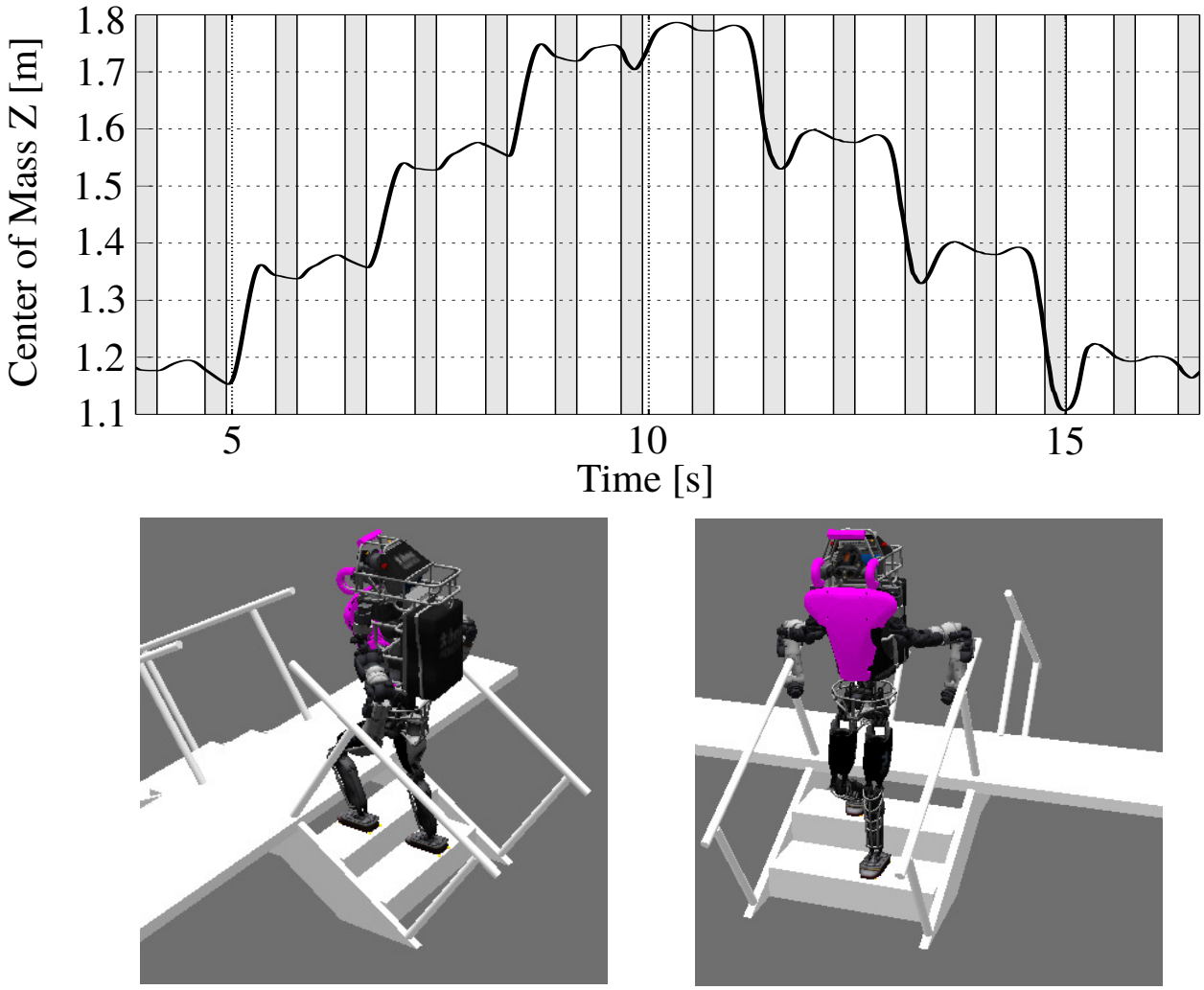


Figure 5.12: Plot of the robot ascending and descending stairs with straight legs in simulation. The stair height is 19.685 cm, the maximum allowable stair height in the US.

calculation. We plan on exploring improving this by limiting the CoP location to lie a certain distance away from the edge of the support polygon, and relying on angular momentum to achieve CMP locations near or past these bounds. As actuator torque and velocity limits are improved, the allowable angular momentum rate and stepping speed will increase, providing much more control authority when stepping. Additional instabilities when walking occur due to stiction in the ankle actuators. The rapid switching from high-impedance control at the ankles during swing to low-impedance control during support has stability issues when dealing with the higher impulse at heel-strike when walking with straight legs. This then requires significant damping in the leg control for the walking to remain stable, which will be improved with better force control

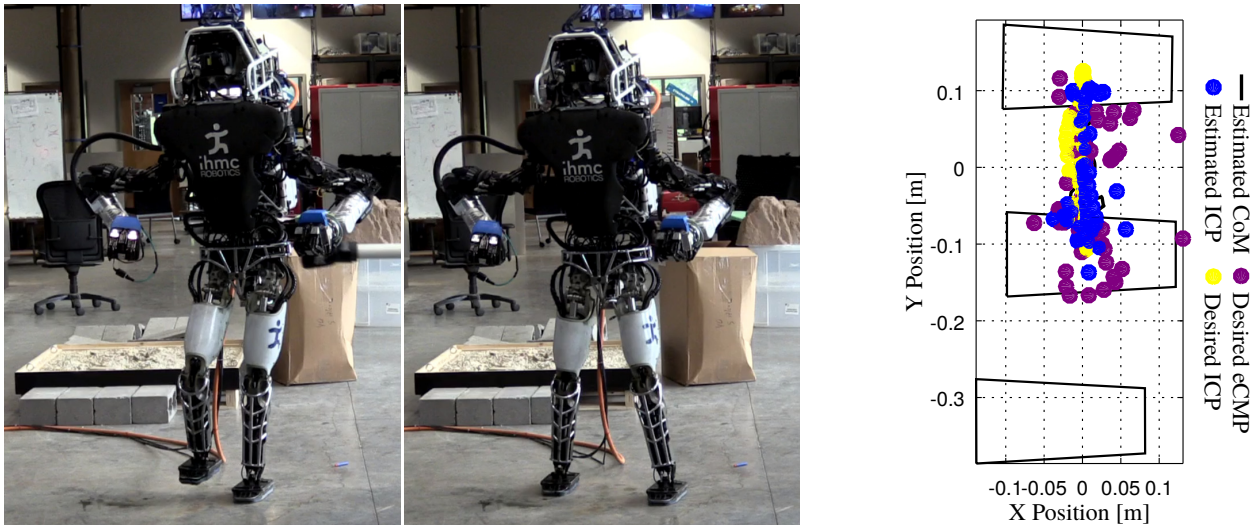


Figure 5.13: Images and data collected from Atlas when the robot is stepping in place and an external disturbance is applied sideways.

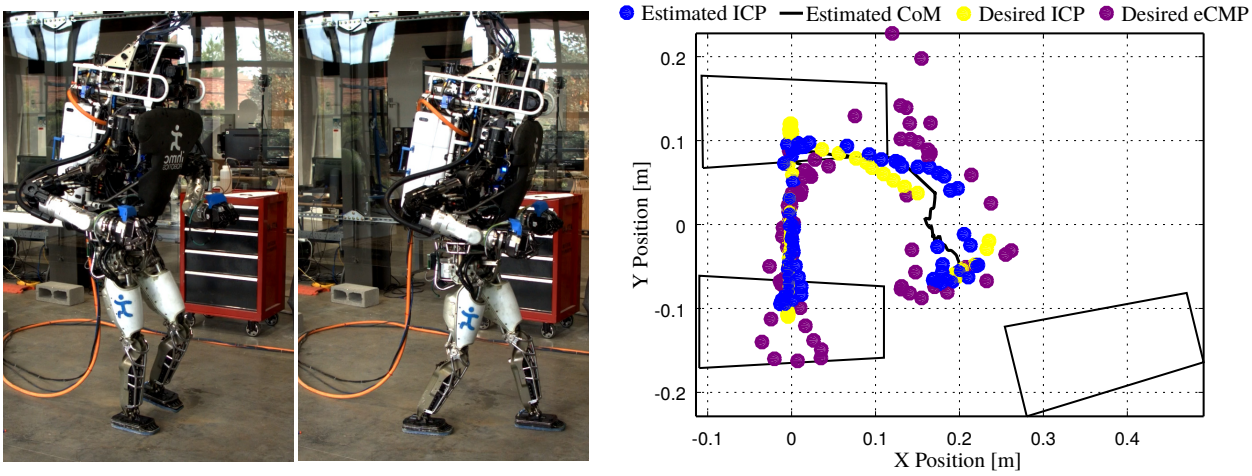


Figure 5.14: Images and data collected from Atlas when the robot is stepping in place and an external disturbance is applied forward.

5.7 Conclusion

Using the presented approach, the Atlas humanoid was capable of walking with straight legs over a variety of terrain both in simulation and on hardware. This was made possible by a novel approach that under-specified the desired motion tasks for the robot. This includes indirectly controlling the center of mass height by biasing the whole-body controller towards

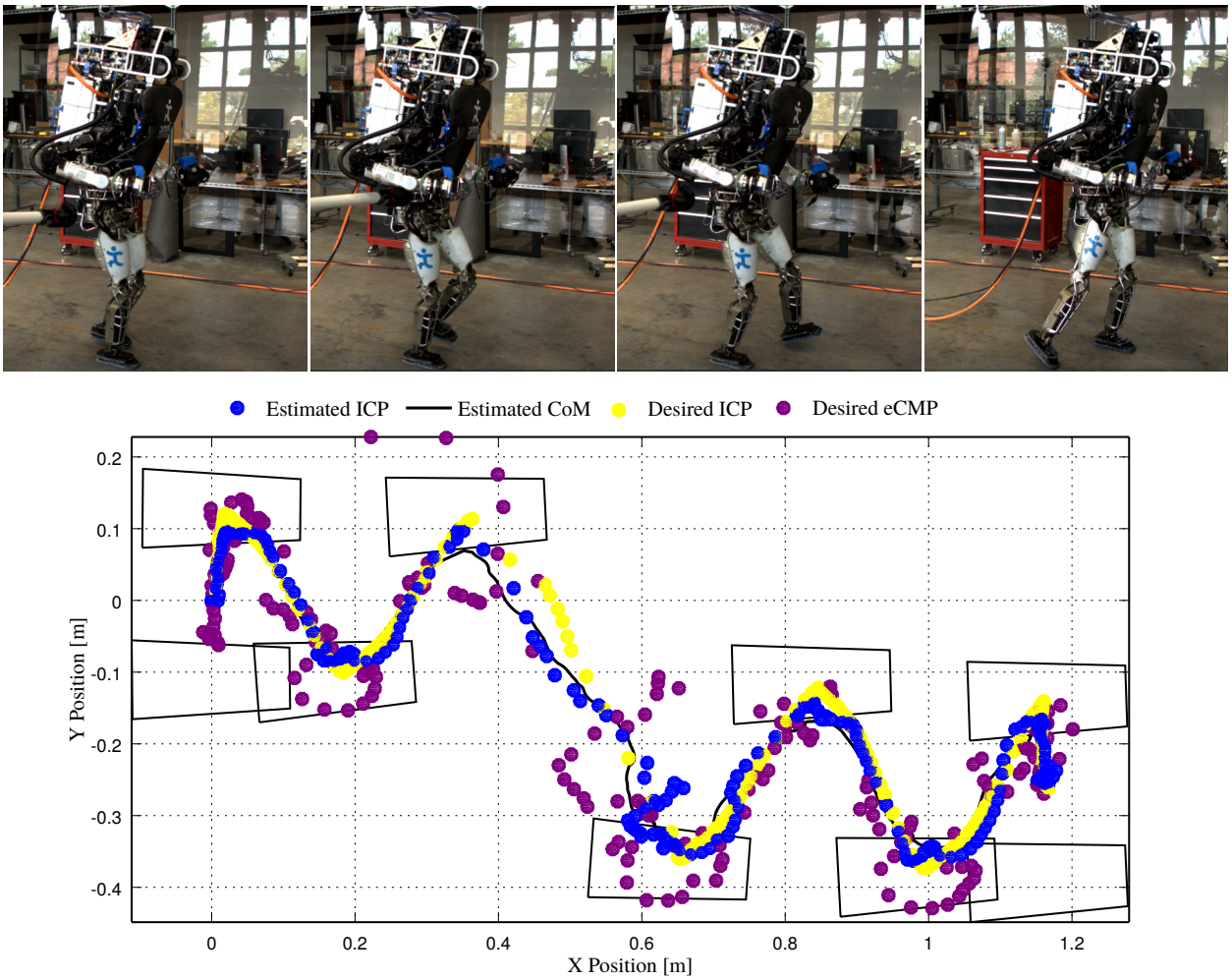


Figure 5.15: Images and data collected from Atlas when the robot is stepping in place and an external disturbance is applied forward.

solutions that use straight legs, and allowing the whole-body controller to perform toe-off motions when necessary through leaving foot pitch uncontrolled. We believe this has several important implications. First, more efficient, natural-looking walking gaits do not require complicated planning, and can be achieved by leaving the height uncontrolled, allowing the whole-body controller to determine when and when not to use straight legs. Second, many motions of the robot may be over-controlled, with natural and intuitive behaviors emerging by allowing the whole-body controller to determine what is necessary to achieve the basic motion tasks, such as desired forces and maintaining foot contact when stepping.

In the future, we hope to further the robustness of this approach to height uncertainties, enabling the robot to traverse similar terrains as humans. This includes incorporation of our work in [125], and improving step adjustment strategies to work on rough terrain using

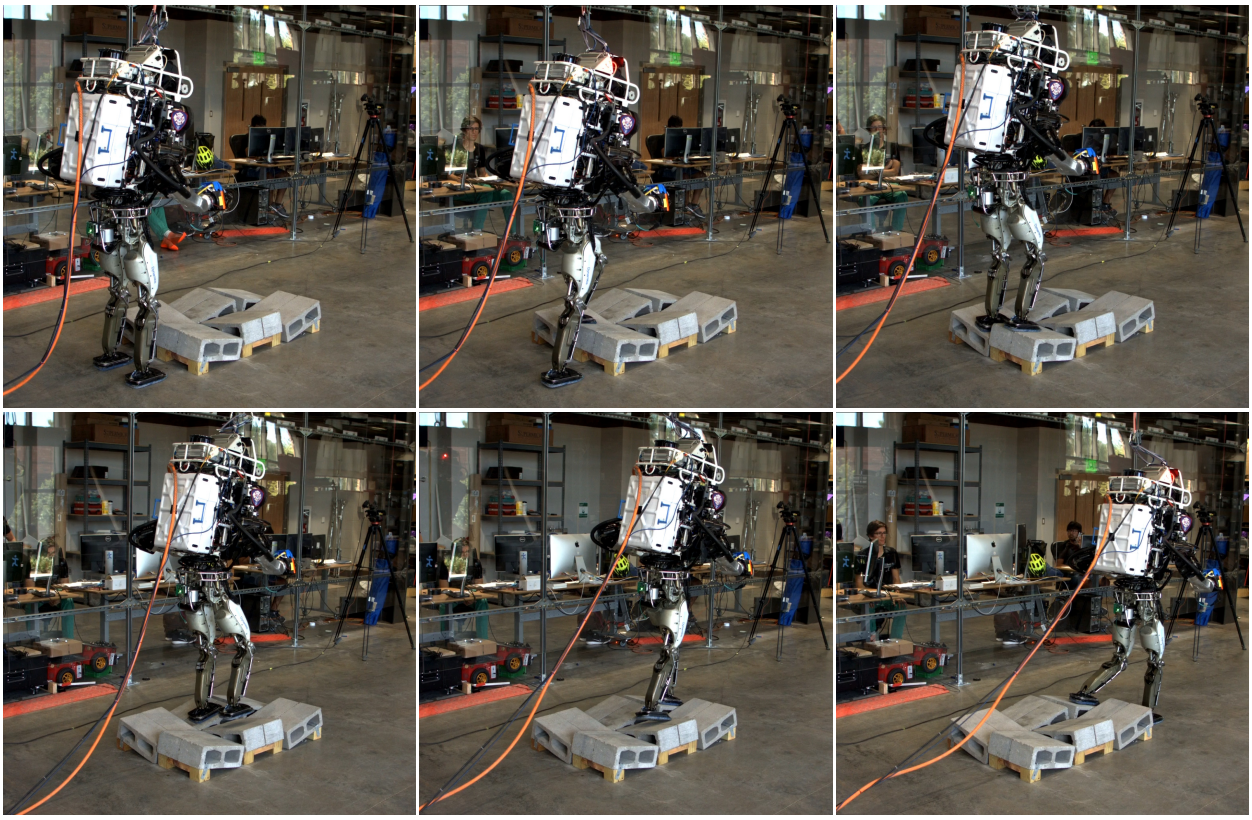


Figure 5.16: Atlas walking over cinder blocks with an indirectly controlled CoM height

convex polygon constraints. We also plan on exploring new strategies for controlling the stance leg configuration, increasing the robustness of the controller to terrain uncertainties.

Chapter 6

Conclusion

Bipedal robots offer a uniquely flexible platform capable of navigating complex, human-centric environments. This makes them ideally suited for a variety of missions, including disaster response and relief, emergency scenarios, or exoskeleton systems for individuals with disabilities. This, however, requires significant advances in humanoid locomotion and control, as they are still slow, unnatural, inefficient, and relatively unstable. The objective of the work featured in this dissertation was to advance the state of the art in bipedal locomotion planning and control for use on bipedal walking robots, with the aim was of increasing the robustness and efficiency. For legged robot's potential to be realized, significant advances beyond the current state-of-the-art to increase robustness and efficiency are necessary, moving towards the natural speed and grace exhibited in human walking. We presented a series of control improvements to enable reliable, robust, natural bipedal locomotion that was validated on a variety of bipedal robots using both hardware and simulation experiments.

6.1 Summary of Work

A huge part of reliable walking involves maximizing the robot's control authority. Our first effort towards this was the development of a model predictive controller to both control the ground reaction forces and perform step adjustment for walking stabilization, which was presented in Chapter 2. This approach included step rotation in the optimization and leveraged the capabilities of the time-varying divergent component of motion for navigating rough terrain, resulting an mixed-integer quadratic program formulation. On evaluating the effects of typical model predictive controllers used in walking, we determined that a different approach for step adjustment and dynamic control would likely be more appropriate. As an alternative, we designed several optimization-based control schemes that embedded a proportional feedback controller into a quadratic program that could use ground reaction feedback (the "ankle strategy"), angular momentum (the "hip strategy"), swing foot speed

up, and step adjustment. We presented and evaluated these approaches in Chapter 3. They were shown to be highly effective, particularly the simplest strategy, greatly improving the robot's tolerance of external disturbances. They allowed the robot to effectively shift its weight, pitch its torso, and adjust its feet to retain balance, while considering environmental constraints.

To enable the robot to walk with straightened legs, we first developed a strategy to check whether or not the dynamic plans used were kinematically and dynamically feasible to execute using straight legs, presented in Chapter 4. Without this, the dynamic plans could require bending the legs significant during execution, as the effects of timing on dynamic plans are typically ignored. This algorithm modifies the step timings to insure the plan can be executed without bending the legs beyond certain angle, while leaving the desired footsteps unmodified. We then presented a novel approach for indirectly controlling the center of mass height through the leg angles to achieve walking with straight legs in Chapter 5. This avoided complicated height planning techniques that are both computationally expensive and often not general enough to consider variable terrain by effectively biasing the solution of the whole-body controller towards using straighter legs. To incorporate the toe-off motion that is essential to both natural and straight leg walking, we also presented a strategy for toe-off control that allows it to be an emergent behavior of the whole-body controller.

The full approach was demonstrated through a series of simulation and experimental results. The straight leg walking approach in Chapter 5 uses the work in Chapter 4 and Chapter 3, and was capable of walking over a variety of terrains, including a cinder block field, stairs, and flat ground, as well as recovering from external disturbances. We discussed many of the practical considerations and limitations required when porting simulation-based controller development to hardware platforms. Using the presented approaches, we demonstrated a important concept: using whole-body control frameworks, not every desired motion need be directly commanded. Many of these motions, such as toe-off, may simply be emergent behaviors that result by attempting to satisfy other objectives, such as desired reaction forces. We also showed that optimization is a very powerful tool for walking control, able to determine both stabilizing inputs and joint torques.

6.2 Discussion and Future Research Areas

While bipedal walking performance has made significant strides forward, considerable progress is still needed for robots to rival the mobility of humans. A healthy human can reliably walk over an extreme variety of terrains, from stone fields to ice, while rarely falling. As illustrated in the 2015 DARPA Robotics Challenge, however, robots are not nearly this capable yet, with robots taking nearly an hour to complete a set of tasks that would take a human less than ten minutes. Indeed, all but one team that a walking robot fell at least once throughout the competition. Exoskeletons, as well, are still far from enabling those with spinal cord injuries the ability to walk as adeptly as an able-bodied individual. Exoskeleton capabilities

were thoroughly demonstrated in the 2016 Cybathlon, where pilots took upwards of ten minutes to complete a course designed to model every day environments that able-bodied individuals completed in one. Not only that, but most pilots at the time of completion were left completely exhausted.

One potential area for performance increases comes in the domain of dynamic planning. While hybrid-zero dynamics based planners have improved significantly, they are still typically far too slow to generate general solutions for walking over a variety of terrain in real-time. As such, reduced order models are still necessary. These plans are typically constrained to the realm of 1-step capturability. While this is not explicitly required by the planner, it often implicitly occurs based on the timing parameters and slow walking speeds of the robots. Additionally, ICP plans currently have no clear extension to three-dimensional planning, relying on their robustness to the errors that changing heights produce. This must be accounted for at some level as robot's abilities to walk over rough terrains are improved. These changing height terrains also pose an interesting problem when consider step timing. Some works have shown that steps are either accelerated or decelerated at different phases of the gait for control going over varying height terrain, but this is obtained through highly time-consuming machine-learning and nonlinear optimization techniques. Some approach for automatically determining the swing and transfer phases that considers terrain information is very much needed for bipedal walking platforms to function autonomously. Otherwise, they will be using the same timings for walking on flat terrain as rough terrain, which will result in performance degradation in one or the other.

While our work represents marked improvements in the robot's control authority, there is still very much room for further enhancements. When a human starts to lose balance, we often will reach out with our hands and brace against a wall or other terrain objects. From a control standpoint, however, this logic is very complicated. Some binary decision, whether or not to brace against the wall, must be introduced, making the walking system even more hybrid than before. Additionally, this decision must be made enough in advance such that there is enough remaining time for the robot to move its hand to the new handhold. Other approaches have explored using the CoM height as a stabilizing mechanism, allowing the CoM to be lowered to help balance. While this problem is tractable in planar motion with a fixed pendulum base, it becomes highly nonlinear for 3D motions that consider a moving center of pressure. Appropriate approaches for combining this strategy with other stabilizing mechanisms is necessary for it to be usable on hardware. Additionally, step adjustment strategies typically utilize the linear inverted pendulum model, which has the intrinsic assumption that the CoM height does not vary. Expanding this slightly to allow the consideration of impulses on foot touchdown can help expand the control authority further. These impulses have the effect of greatly slowing the inverted pendulum down, requiring considerably shorter steps to come to a rest when recovering from tracking errors.

Even with improved control authority with step adjustment, these strategies are still relatively limited when traversing complicated terrain. When going up stairs, the amount of footstep adjustment that can be used and remain on the same step is very limited. A clear

mechanism for when to transition to a different constraint surface is very much needed for these strategies to be fully useful in the real world. This could be done by projecting the planar regions that constitute the environment into the ICP control plane, and then using the planar region that has the largest intersecting area with the capture region as the constraint set for the upcoming footstep.

Whole-body controllers have been shown to be very good for resolving multiple motion tasks, but have the inherent limitation of only considering the optimal solution for the single time-step in question, with no consideration for global optimality over time. This has the unfortunate side effect of choosing actions that only result in the minimal cost for this slight in time, but may require extremely costly motions later. To date, this has primarily been addressed at the planning level. Significant effort is spent on the motion planners so that the whole-body controller produces "good" motions throughout the trajectory. Approaches that are capable of looking several ticks ahead in time, however, have the potential to offer significant performance improvements, particularly if they are capable of considering the switching constraint set such as from contact changes.

These algorithms also offer significant potential for the application to exoskeletons. The use of powered ankle flexion in exoskeletons will also enable the exploration of extending humanoid balance strategies to exoskeletons. As the underlying rigid-body dynamics both systems are the same, similar control strategies to those utilized by humanoids should be possible. The adaptation of humanoid balance strategies to exoskeleton platforms is a non-trivial task, though. The inclusion of a human first and foremost introduces significant uncertainties to the rigid-body dynamics. While some assumptions about the inertias can be made, rigid body approximations are considerably less accurate for the compliant bodies that make up humans than they are the rigid links of a robot. Any efforts by the pilot to balance pilot cannot be ignored, either. This entails the crutches imposing unknown reaction forces on the environment, and the pilot's upper body generating significant inertial effects when moving. Instead of seeking to provide the perfect control input for balance, as is required for fully-autonomous balance on robots, exoskeletons could be used to provide balance assistance. As long as the force vectors for balance are in the right direction, they are offloading some of the required effort for balance from the pilot to the robot. As such, heuristic based controllers, such as a virtual model controller for balancing using a very rough estimate of the ICP should be capable of providing considerable improvements in the pilot's performance, decreasing his fatigue while increasing his overall stability.

Bibliography

[1] Lie Yukou. *Liezi*.

[2] Ichiro Kato. "Development of WABOT 1". In: *Biomechanism* 2 (1973), pp. 173–214.

[3] Stephanie M. McPherson. *How Battle-Tested Robots Are Helping Out at Fukushima*. 2011. URL: http://www.popularmechanics.com/military/a6656/how-battle-tested-robots-are-helping-out-at-fukushima-5586925/.

[4] Douglas E. Garland et al. "Osteoporosis after spinal cord injury". In: *Journal of Orthopaedic Research* 10.3 (1992), pp. 371–378.

[5] Stefan Goemaere et al. "Bone mineral status in paraplegic patients who do or do not perform standing". In: *Osteoporosis International* 4 (1994), pp. 138–143.

[6] Jenny Wall. "Preventing pressure sores among wheelchair users". In: *Professional Nurse* 15.5 (2000), pp. 321–324.

[7] Mary A. Regan et al. "A systematic review of therapeutic interventions for pressure ulcers following spinal cord injury". In: *Archives of Physical Medicine and Rehabilitation* 90.2 (2009), pp. 213–231.

[8] Michael J. Castro et al. "Influence of complete spinal cord injury on skeletal muscle within 6 mo of injury". In: *Journal of Applied Physiology* 86.1 (1999), pp. 350–358.

[9] Suzanne L. Groah and Indira S. Lanig. "Neuromusculoskeletal syndromes in wheelchair athletes". In: *Seminars in Neurology* 20.2 (2000), pp. 201–208.

[10] Marie Alm, Helena Saraste, and Cecilia Norrbrink. "Shoulder pain in persons with thoracic spinal cord injury: prevalence and characteristics". In: *Journal of Rehabilitation Medicine* 40.4 (2008), pp. 277–283.

[11] Monica A Daley and Andrew A Biewener. "Running over rough terrain reveals limb control for intrinsic stability". In: *Proceedings of the National Academy of Sciences* 103.42 (2006), pp. 15681–15686.

[12] Miomir Vukobratovic, D Hristic, and Z Stojiljkovic. "Development of active anthropomorphic exoskeletons". In: *Medical and Biological Engineering* 12.1 (1974), pp. 66–80.

[13] Shuuji Kajita et al. "The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 1. 2001, pp. 239–246.

[14] Tad McGeer. "Passive walking with knees". In: *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*. IEEE. 1990, pp. 1640–1645.

[15] Ambarish Goswami, Benoit Thuilot, and Bernard Espiau. *Compass-like biped robot part I: Stability and bifurcation of passive gaits*. Tech. rep. INRIA, 1996.

[16] Vanessa F Hsu Chen. "Passive dynamic walking with knees: A point foot model". PhD thesis. Citeseer, 2007.

[17] Jerry Pratt et al. "Capture Point: A Step toward Humanoid Push Recovery". In: *6th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Dec. 2006, pp. 200–207.

[18] Toru Takenaka, Takashi Matsumoto, and Takahide Yoshiike. "Real time motion generation and control for biped robot -1st report: Walking gait pattern generation". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Oct. 2009, pp. 1084–1091.

[19] Hartmut Geyer, Andre Seyfarth, and Reinhard Blickhan. "Compliant leg behaviour explains basic dynamics of walking and running". In: *Proc. Biol Sci.* Nov. 2006.

[20] Miomir Vukobratovic and Branislav Borovac. "Zero-Moment Point - Thirty Five Years of Its Life". In: 1.1 (2004), pp. 157–173.

[21] Philippe Sardain and Guy Bessonnet. "Forces acting on a biped robot. Center of Pressure-Zero Moment Point". In: *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 34.5 (Sept. 2004), pp. 630–637. ISSN: 1083-4427.

[22] Shuuji Kajita et al. "Biped walking pattern generation by using preview control of zero-moment point". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2. 2003, pp. 1620–1626.

[23] Satoshi Kagami et al. "A Fast Dynamically Equilibrated Walking Trajectory Generation Method of Humanoid Robot". In: *Autonomous Robots* 12.1 (2002), pp. 71–82. ISSN: 1573-7527. DOI: 10.1023/A:1013210909840. URL: http://dx.doi.org/10.1023/A:1013210909840.

[24] Koichi Nishiwaki and Satoshi Kagami. "Strategies for adjusting the ZMP reference trajectory for maintaining balance in humanoid walking". In: *2010 IEEE International Conference on Robotics and Automation*. May 2010, pp. 4230–4236. DOI: 10.1109/ROBOT.2010.5510002.

[25] Pierre-Brice Wieber. "Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations". In: *6th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Genova, Italy, Dec. 2006, pp. 137–142.

[26] Dimitar Dimitrov et al. "On the implementation of model predictive control for on-line walking pattern generation". In: *Robotics and Automation (ICRA), IEEE International Conference on*. May 2008, pp. 2685–2690.

[27] Holger Diedam et al. "Online walking gait generation with adaptive foot positioning through Linear Model Predictive control". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sept. 2008, pp. 1121–1126.

[28] Jerry E Pratt and Russ Tedrake. "Velocity-Based Stability Margins for Fast Bipedal Walking". In: *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*. Ed. by Moritz Diehl and Katja Mombaur. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 299–324. ISBN: 978-3-540-36119-0. DOI: 10.1007/978-3-540-36119-0_14. URL: http://dx.doi.org/10.1007/978-3-540-36119-0%7B%5C_%7D14.

[29] Russ Tedrake et al. "A closed-form solution for real-time ZMP gait generation and feedback stabilization". In: *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. Nov. 2015, pp. 936–940.

[30] Scott Kuindersma, Frank Permenter, and Russ Tedrake. "An efficiently solvable quadratic program for stabilizing dynamic locomotion". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 2589–2594.

[31] Siyuan Feng et al. "3D Walking Based on Online Optimization". In: *Humanoid Robots (Humanoids), 13th IEEE-RAS International Conference on*. Oct. 2013.

[32] Steven H. Collins, Martijn Wisse, and Andy Ruina. "A three-dimensional passive-dynamic walking robot with two legs and knees". In: *The International Journal of Robotics Research* 20.7 (2001), pp. 607–615.

[33] Steven H. Collins et al. "Efficient bipedal robots based on passive-dynamic walkers". In: *Science* 307.5712 (2005), pp. 1082–1085.

[34] Arthur D. Kuo. "Stabilization of lateral motion in passive dynamic walking". In: *The International journal of robotics research* 18.9 (1999), pp. 917–930.

[35] Feng Tan, Chenglong Fu, and Ken Chen. "Biped blind walking on changing slope with reflex control system". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 1709–1714.

[36] AL Hof, MGJ Gazendam, and WE Sinke. "The condition for dynamic stability". In: *Journal of Biomechanics* 38.1 (2005), pp. 1–8.

[37] Johannes Englsberger, Christian Ott, and Alin Albu-Schaffer. "Three-dimensional bipedal walking control using Divergent Component of Motion". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Nov. 2013, pp. 2600–2607.

[38] Michael A. Hopkins, Dennis W. Hong, and Alexander Leonessa. "Humanoid Locomotion on Uneven Terrain Using the Time-Varying Divergent Component of Motion". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. Nov. 2014, pp. 266–272.

[39] Johannes Englsberger et al. "Bipedal walking control based on Capture Point dynamics". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Sept. 2011, pp. 4420–4427.

[40] Jerry E. Pratt et al. "Capturability-based analysis and control of legged locomotion, part 2: Application to M2V2, a lower body humanoid". In: *The International Journal of Robotics Research* 31.10 (2012), pp. 1117–1133.

[41] Mitsuharu Morisawa et al. "Balance control based on Capture Point error compensation for biped walking on uneven terrain". In: *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Nov. 2012, pp. 734–740.

[42] Johannes Englsberger et al. "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 (IROS)*. Sept. 2014, pp. 4022–4029.

[43] Michael A. Hopkins, Dennis W. Hong, and Alexander Leonessa. "Compliant Locomotion Using Whole-Body Control and Divergent Component of Motion Tracking". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. May 2015, pp. 5726–5733.

[44] Michael A. Hopkins et al. "Design of a compliant bipedal walking controller for the DARPA Robotics Challenge". In: *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. Nov. 2015, pp. 831–837.

[45] Twan Koolen et al. "Capturability-based Analysis and Control of Legged Locomotion, Part 1: Theory and Application to Three Simple Gait Models". In: *The International Journal of Robotics Research* 31.9 (Aug. 2012), pp. 1094–1113.

[46] Manuel Krause et al. "Stabilization of the Capture Point Dynamics for Bipedal Walking based on Model Predictive Control". In: *IFAC Symposium on Robot Control (SYROCO)*. Vol. 10. 1. 2012, pp. 165–171.

[47] Coleman Knabe et al. "Designing for Compliance: ESCHER, Team VALOR's Compliant Biped". In: *Journal of Field Robotics* (2016).

[48] Matthew Johnson et al. "Team IHMC's Lessons Learned from the DARPA Robotics Challenge Trials". In: *Journal of Field Robotics* 32.2 (2015), pp. 192–208. ISSN: 14746670. DOI: 10.1002/rob.

[49] Michael A. Hopkins et al. "Embedded joint-space control of a series elastic humanoid". In: *Intelligent Robot and Systems (IROS), 2015 IEEE/RSJ International Conference on*. Sept. 2015, pp. 3358–3365.

[50] Jerry Pratt, Peter Dilworth, and Gill Pratt. "Virtual model control of a bipedal walking robot". In: *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*. Vol. 1. IEEE. 1997, pp. 193–198.

[51] Jerry Pratt et al. "Virtual model control: An intuitive approach for bipedal locomotion". In: *The International Journal of Robotics Research* 20.2 (2001), pp. 129–143.

[52] Jianjuen J Hu et al. "Adaptive virtual model control of a bipedal walking robot". In: *Intelligence and Systems, 1998. Proceedings., IEEE International Joint Symposia on*. IEEE. 1998, pp. 245–251.

[53] Jianjuen J Hu et al. "Virtual model based adaptive dynamic control of a biped walking robot". In: *International Journal on Artificial Intelligence Tools* 8.03 (1999), pp. 337–348.

[54] Stelian Coros, Philippe Beaudoin, and Michiel Van de Panne. "Generalized biped walking control". In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 4. ACM. 2010, p. 130.

[55] Sang-Ho Hyon and Gordon Cheng. "Disturbance rejection for biped humanoids". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE. 2007, pp. 2668–2675.

[56] Sang-Ho Hyon, Rieko Osu, and Yohei Otaka. "Integration of multi-level postural balancing on humanoid robots". In: *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE. 2009, pp. 1549–1556.

[57] Luis Sentis and Oussama Khatib. "Control of Free-Floating Humanoid Robots Through Task Prioritization". In: *Robotics and Automation (ICRA), IEEE International Conference on*. 2005.

[58] Luis Sentis and Oussama Khatib. "A whole-body control framework for humanoids operating in human environments". In: *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*. May. Ieee, 2006, pp. 2641–2648. ISBN: 0-7803-9505-0. DOI: 10.1109/ROBOT.2006.1642100. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1642100.

[59] Jaeheung Park and Oussama Khatib. "Contact consistent control framework for humanoid robots". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE. 2006, pp. 1963–1969.

[60] Oussama Khatib, Luis Sentis, and Jae-Heung Park. "A unified framework for whole-body humanoid robot control with multiple constraints and contacts". In: *European Robotics Symposium 2008* (2008), pp. 303–312. URL: http://www.springerlink.com/index/164t085051013365.pdf.

[61] Martin de Lasa and Aaron Hertzmann. "Prioritized optimization for task-space control". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Oct. 2009, pp. 5755–5762.

[62] Michael Mistry, Jonas Buchli, and Stefan Schaal. "Inverse dynamics control of floating base systems using orthogonal decomposition." In: *ICRA*. Citeseer. 2010, pp. 3406–3412.

[63] Benjamin J. Stephens and Christopher G. Atkeson. "Dynamic Balance Force Control for compliant humanoid robots". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Oct. 2010, pp. 1248–1255.

[64] Sung-Hee Lee and Ambarish Goswami. "Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Oct. 2010, pp. 3157–3162.

[65] Sung-Hee Lee and Ambarish Goswami. "A momentum-based balance controller for humanoid robots on non-level and non-stationary ground". In: *Autonomous Robots* 33.4 (2012), pp. 399–414.

[66] Ludovic Righetti and Stefan Schaal. "Quadratic programming for inverse dynamics with optimal distribution of contact forces". In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)* (Nov. 2012), pp. 538–543. DOI: 10.1109/HUMANOIDS.2012.6651572. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6651572.

[67] Ludovic Righetti et al. "Optimal distribution of contact forces with inverse-dynamics control". In: *The International Journal of Robotics Research* 32.3 (2013), pp. 280–298.

[68] Patrick M. Wensing and David E. Orin. "Generation of dynamic humanoid behaviors through task-space control with conic optimization". In: *Robotics and Automation (ICRA), IEEE International Conference on*. May 2013, pp. 3103–3109.

[69] Alexander Herzog et al. "Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics". In: *IEEE International Conference on Intelligent Robots and Systems* (2014), pp. 981–988. ISSN: 21530866. DOI: 10.1109/IROS.2014.6942678. arXiv: 1305.2042.

[70] Siyuan Feng et al. "Optimization Based Full Body Control for the Atlas Robot". In: *Humanoid Robots (Humanoids), 14th IEEE-RAS International Conference on*. Nov. 2014.

[71] Siyuan Feng et al. "Optimization-based Fully Body Control for the DARPA Robotics Challenge". In: *Journal of Field Robotics* 32.2 (2015), pp. 293–312. ISSN: 14746670. DOI: 10.1002/rob.

[72] Scott Kuindersma et al. "Optimization-Based Locomotion Planning, Estimation and Control Design for Atlas". In: *Autonomous Robots* 40.3 (2016), pp. 429–455.

[73] Twan Koolen et al. "Summary of team IHMC's virtual robotics challenge entry". In: *Humanoid Robots (Humanoids), 13th IEEE-RAS International Conference on*. Oct. 2013.

[74] Coleman Knabe et al. "Design of a Series Elastic Humanoid for the DARPA Robotics Challenge". In: *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. Nov. 2015, pp. 738–743.

[75] Thomas J Roberts and Emanuel Azizi. "Flexible mechanisms: the diverse roles of biological springs in vertebrate movement". In: *The Journal of experimental biology* 214.3 (2011), pp. 353–361.

[76] Samuel K Au, Jeff Weber, and Hugh Herr. "Powered Ankle–Foot Prosthesis Improves Walking Metabolic Economy". In: *Robotics, IEEE Transactions on* 25.1 (2009), pp. 51–66.

[77] Jacob Reher et al. "Realizing dynamic and efficient bipedal locomotion on the humanoid robot DURUS". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016.

[78] Anirban Mazumdar et al. "Using parallel stiffness to achieve improved locomotive efficiency with the Sandia STEPPR robot". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. May 2015, pp. 835–841. DOI: 10.1109/ICRA.2015.7139275.

[79] Gill A. Pratt and Matthew M. Williamson. "Series elastic actuators". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on. 'Human Robot Interaction and Cooperative Robots'*. Vol. 1. Aug. 1995, pp. 399–406.

[80] David W. Robinson et al. "Series elastic actuator development for a biomimetic walking robot". In: *Advanced Intelligent Mechatronics, IEEE/ASME International Conference on*. 1999, pp. 561–568.

[81] Kyoungchul Kong, Joonbum Bae, and Masayoshi Tomizuka. "Control of rotary series elastic actuator for ideal force-mode actuation in human–robot interaction applications". In: *IEEE/ASME transactions on mechatronics* 14.1 (2009), pp. 105–118.

[82] Nicholas Paine, Sehoon Oh, and Luis Sentis. "Design and control considerations for high-performance series elastic actuators". In: *IEEE/ASME Transactions on Mechatronics* 19.3 (2014), pp. 1080–1091.

[83] Daniel Joseph Paluska. "Design of a Humanoid Biped for Walking Research". MA thesis. Massachusetts Institute of Technology, Sept. 2000.

[84] Jerry Pratt and Ben Krupp. "Design of a bipedal walking robot". In: *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. Vol. 6962. Apr. 2008.

[85] M Slovich et al. "Building HUME: A Bipedal Robot for Human-Centered Hyper-Agility". In: *Dynamic Walking Meeting*. 2012.

[86] Donghyun Kim et al. "Stabilizing series-elastic point-foot bipeds using whole-body operational space control". In: *IEEE Transactions on Robotics* (2016).

[87] Nicolaus A Radford et al. "Valkyrie: NASA's first bipedal humanoid robot". In: *Journal of Field Robotics* 32.3 (2015), pp. 397–419.

[88] Derek Lahr et al. "Development of a Parallely Actuated Humanoid, SAFFiR". In: *ASME International Design Engineering Technical Conference*. 2013.

[89] Bryce Lee et al. "Design of a Human-Like Range of Motion Hip Joint for Humanoid Robots". In: *ASME International Design Engineering Technical Conference*. 2014.

[90] Coleman Knabe, Bryce Lee, and Dennis Hong. "An inverted straight line mechanism for augmenting joint range of motion in a humanoid robot". In: *ASME International Design Engineering Technical Conference*. 2014.

[91] Brian Lattimer et al. "Humanoid Firefighting Robot for Structure Fires". In: *14th International Conference and Exhibition on Fire Science and Engineering (INTERFLAM)*. 2016.

[92] Nikos G. Tsagarakis et al. "COMpliant huMANoid COMAN: Optimal joint stiffness tuning for modal frequency control". In: *Robotics and Automation (ICRA), IEEE International Conference on*. May 2013, pp. 673–678.

[93] Alireza Ramezani and Jessy W. Grizzle. "ATRIAS 2.0, a new 3D bipedal robotic walker and runner". In: *Proceedings of the 2012 International Conference on Climbing and walking Robots and the Support Technologies for Mobile Machines*. 2012, pp. 467–474.

[94] Nikos G Tsagarakis et al. "WALK-MAN: A high performance humanoid platform for realistic environments". In: *Journal of Field Robotics (JFR)* (2016).

[95] Francesca Negrello et al. "WALK-MAN humanoid lower body design optimization for enhanced physical performance". In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1817–1824.

[96] Marco Hutter et al. "StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion". In: *15th International Conference on Climbing and Walking Robot (CLAWAR)*. 2012.

[97] Marco Hutter et al. "ANYmal-A Highly Mobile and Dynamic Quadrupedal Robot". PhD thesis. 2016.

[98] Daan Hobbelen, Tomas de Boer, and Martijn Wisse. "System overview of bipedal robots Flame and TUlip: Tailor-made for Limit Cycle Walking". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. Sept. 2008, pp. 2486–2491.

[99] Johannes Englsberger et al. "Overview of the torque-controlled humanoid robot TORO". In: *Humanoid Robots (Humanoids), 14th IEEE-RAS International Conference on*. Nov. 2014.

[100] Giorgio Metta et al. "The iCub humanoid robot: an open platform for research in embodied cognition". In: *Proceedings of the 8th workshop on performance metrics for intelligent systems*. ACM. 2008, pp. 50–56.

[101] Claudio Semini et al. "Design of HyQ - a hydraulically and electrically actuated quadruped robot". In: *Journal of Systems and Control Engineering* 225.6 (2011), pp. 831–849.

[102] Claudio Semini et al. "Design of the Hydraulically-Actuated, Torque-Controlled Quadruped Robot HyQ2Max". In: *IEEE/ASME Transactions on Mechatronics* (2016). ISSN: 1083-4435. DOI: 10.1109/TMECH.2016.2616284.

[103] Robert J. Griffin and Alexander Leonessa. "Model predictive control for dynamic footstep adjustment using the divergent component of motion". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 1763–1768.

[104] Robert J Griffin et al. "Walking Stabilization Using Step Timing and Location Adjustment on the Humanoid Robot, Atlas". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. 2017.

[105] Robert J Griffin et al. "Capture Point Trajectories for Reduced Knee Bend using Step Time Optimization". In: *17th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. In Review. 2017.

[106] Robin Deits and Russ Tedrake. "Footstep planning on uneven terrain with mixed-integer convex optimization". In: *14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Nov. 2014, pp. 279–286.

[107] Joel Chestnutt et al. "Planning Biped Navigation Strategies in Complex Environments". In: *3rd IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Oct. 2003.

[108] Armin Hornung et al. "Anytime search-based footstep planning with suboptimality bounds". In: *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Nov. 2012, pp. 674–679.

[109] Andrei Herdt et al. "Online walking motion generation with automatic footstep placement". In: *Advanced Robotics* 24.5-6 (2010), pp. 719–737.

[110] Shuuji Kajita et al. "Biped walking stabilization based on linear inverted pendulum tracking". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2010, pp. 4489–4496.

[111] Tomomichi Sugihara. "Standing stabilizability and stepping maneuver in planar bipedalism based on the best COM-ZMP regulator". In: *IEEE International Conference on Robotics and Automation (ICRA)*. May 2009, pp. 1966–1971.

[112] Camille Brasseur et al. "A robust linear MPC approach to online generation of 3D biped walking motion". In: *15th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Nov. 2015, pp. 595–601.

[113] Georg Weidebach et al. "Walking on Partial Footholds Including Line Contacts with the Humanoid Robot Atlas". In: *16th IEEE-RAS International Conference on Humanoid Robots (Humanoids) (Forthcoming)*. 2016.

[114] Twan Koolen, Michael Posa, and Russ Tedrake. "Balance control using center of mass height variation: limitations impose by unilateral contact". In: *16th IEEE-RAS International Conference on Humanoid Robots (Humanoids) (Forthcoming)*. 2016.

[115] Siyuan Feng et al. "Robust Dynamic Walking Using Online Foot Step Optimization". In: *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*. 2016.

[116] Jonghoon Park and Youngil Youm. "General ZMP Preview Control for Bipedal Walking". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Apr. 2007, pp. 2682–2687.

[117] Johannes Mayr, Hubert Gattringer, and Hartmut Bremer. "Online Walking Gait Generation with Predefined Variable Height of the Center of Mass". In: *Intelligent Robotics and Applications*. Ed. by Sabina Jeschke, Honghai Liu, and Daniel Schilberg. Vol. 7102. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 569–578. ISBN: 978-3-642-25488-8.

[118] Johan Lofberg. *Big-M And Convex Hulls*. 2013. URL: `users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Tutorials.Big-MAndConvexHulls`.

[119] Inc. Gurobi Optimization. *Gurobi optimizer reference manual*. 2016. URL: `http://www.gurobi.com`.

[120] N. Koenig and A. Howard. "Design and use paradigms for Gazebo, an open-source multi-robot simulator". In: *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. Sept. 2004, 2149–2154 vol.3.

[121] Fay B Horak and Lewis M Nashner. "Central programming of postural movements: adaptation to altered support-surface configurations". In: *Journal of neurophysiology* 55.6 (1986), pp. 1369–1381.

[122] Mirjam Pijnappels et al. "Armed against falls: the contribution of arm movements to balance recovery after tripping". In: *Experimental brain research* 201.4 (2010), pp. 689–699.

[123] Brian E Maki and William E McIlroy. "The role of limb movements in maintaining upright stance: the" change-in-support" strategy". In: *Physical therapy* 77.5 (1997), p. 488.

[124] Twan Koolen et al. "Design of a momentum-based control framework and application to the humanoid robot atlas". In: *International Journal of Humanoid Robotics* 13.01 (2016), p. 1650007.

[125] Georg Wiedebach et al. "Walking on Partial Footholds Including Line Contacts with the Humanoid Robot Atlas". In: *16th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2016.

[126] Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. "Three-dimensional bipedal walking control based on divergent component of motion". In: *IEEE Transactions on Robotics* 31.2 (2015), pp. 355–368.

[127] Przemyslaw Kryczka et al. "Online regeneration of bipedal walking gait pattern optimizing footstep placement and timing". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 3352–3357.

[128] Zohaib Aftab, Thomas Robert, and Pierre-Brice Wieber. "Ankle, hip and stepping strategies for humanoid balance recovery with a single Model Predictive Control scheme". In: *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2012, pp. 159–164.

[129] Majid Khadiv et al. "Step Timing Adjustment: A Step toward Generating Robust Gaits". In: *arXiv preprint arXiv:1610.02377*. 2016.

[130] Marko B. Popovic and Hugh Herr. "Ground reference points in legged locomotion: Definitions, biological trajectories and control implications". In: *Int. J. Robot. Res* 24.12 (2005), pp. 1013–1032.

[131] Daniel Mellinger and Vijay Kumar. "Minimum snap trajectory generation and control for quadrotors". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2011, pp. 2520–2525.

[132] David A Winter. *Biomechanics and motor control of human movement*. John Wiley & Sons, 2009.

[133] Kensuke Harada et al. "An analytical method for real-time gait planning for humanoid robots". In: *International Journal of Humanoid Robotics* 3.01 (2006), pp. 1–19.

[134] Zhibin Li et al. "Trajectory generation of straightened knee walking for humanoid robot iCub". In: *11th International Conference on Control, Automation, Robotics and Vision, ICARCV 2010*. 2010, pp. 2355–2360.

[135] K Van Heerden. "Planning COM trajectory with variable height and foot position with reactive stepping for humanoid robots". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. 2015, pp. 6275–6280.

[136] Oscar E Ramos, Nicolas Mansard, and Philippe Soueres. "Whole-body motion integrating the capture point in the operational space inverse dynamics control". In: *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*. IEEE. 2014, pp. 707–712.

[137] Mitsuharu Morisawa et al. "Pattern generation of biped walking constrained on parametric surface". In: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE. 2005, pp. 2405–2410.

[138] David E Orin, Ambarish Goswami, and Sung-Hee Lee. "Centroidal dynamics of a humanoid robot". In: *Autonomous Robots* 35.2-3 (2013), pp. 161–176.

[139] Yangwei You et al. "Straight leg walking strategy for torque-controlled humanoid robots". In: *Robotics and Biomimetics (ROBIO), 2016 IEEE International Conference on*. IEEE. 2016, pp. 2014–2019.

[140] Arthur D Kuo, J Maxwell Donelan, and Andy Ruina. "Energetic consequences of walking like an inverted pendulum: step-to-step transitions". In: *Exercise and sport sciences reviews* 33.2 (2005), pp. 88–97.