Article

The International Journal of Robotics Research
2020, Vol. 39(8) 936–956

DOI: 10.1177/0278364920918014
journals.sagepub.com/home/ijr
SAGE

Dynamic locomotion for passive-ankle biped robots and humanoids using whole-body locomotion control

Donghyun Kim[1], Steven Jens Jorgensen[2], Jaemin Lee[2], Junhyeok Ahn[2], Jianwen Luo[3] and Luis Sentis[2]

Abstract

Whole-body control (WBC) is a generic task-oriented control method for feedback control of loco-manipulation behaviors in humanoid robots. The combination of WBC and model-based walking controllers has been widely utilized in various humanoid robots. However, to date, the WBC method has not been employed for unsupported passive-ankle dynamic locomotion. As such, in this article, we devise a new WBC, dubbed the whole-body locomotion controller (WBLC), that can achieve experimental dynamic walking on unsupported passive-ankle biped robots. A key aspect of WBLC is the relaxation of contact constraints such that the control commands produce reduced jerk when switching foot contacts. To achieve robust dynamic locomotion, we conduct an in-depth analysis of uncertainty for our dynamic walking algorithm called the time-to-velocity-reversal (TVR) planner. The uncertainty study is fundamental as it allows us to improve the control algorithms and mechanical structure of our robot to fulfill the tolerated uncertainty. In addition, we conduct extensive experimentation for: (1) unsupported dynamic balancing (i.e., in-place stepping) with a six-degree-of-freedom biped, Mercury; (2) unsupported directional walking with Mercury; (3) walking over an irregular and slippery terrain with Mercury; and 4) in-place walking with our newly designed ten-DoF viscoelastic liquid-cooled biped, DRACO. Overall, the main contributions of this work are on: (a) achieving various modalities of unsupported dynamic locomotion of passive-ankle bipeds using a WBLC controller and a TVR planner; (b) conducting an uncertainty analysis to improve the mechanical structure and the controllers of Mercury; and (c) devising a whole-body control strategy that reduces movement jerk during walking.

Keywords

Legged robots, dynamic locomotion, humanoid robots

1. Introduction

Passive-ankle walking has some key differences with respect to ankle actuated biped legged locomotion: (1) bipeds with passive ankles have fewer degrees of freedom (DoFs) than ankle actuated legged robots resulting in lower mechanical complexity and lighter lower legs; (2) bipeds with passive ankles have tiny feet, which leads to a small horizontal footprint of the robot. Our article targets passive- and quasi-passive-ankle legged robots in leveraging the above characteristics. In addition, there is a disconnect between dynamic legged locomotion methods (e.g., Hartley et al., 2017; Rezazadeh et al., 2015) and humanoid control methods (e.g., Escande et al., 2014; Koolen et al., 2016; Kuindersma et al., 2015), the latter focusing on coordinating loco-manipulation behaviors. Humanoid robots such as those used during the DARPA robotics challenges (DRCs) have often employed task-oriented inverse kinematics and inverse dynamics methods coupled with control of the robots' horizontal center of mass (CoM) demonstrating versatility for whole-body behaviors (Feng et al., 2015; Johnson et al., 2015; Kohlbrecher et al., 2014; Radford et al., 2015a). However, they have been practically slower and less robust to external disturbances than bipeds employing dynamic locomotion

[1]Massachusetts Institute of Technology, Cambridge, MA, USA
[2]University of Texas at Austin, Austin, TX, USA
[3]Robotics Lab, Stanford University, Palo Alto, CA, USA

Corresponding author:
Luis Sentis, University of Texas at Austin, 2617 Wichita Street, Austin, TX 78712, USA.
Email: lsentis@austin.utexas.edu

methods, which do not rely on horizontal CoM control. This article aims to explore and offer a solution to close the gap between these two lines of controls, i.e., versatile task-oriented controllers and dynamic locomotion controllers.

There is a family of walking control methods (Hubicki et al., 2018; Raibert et al., 1984) that do not rely on explicit control of the horizontal CoM movement enabling passive-ankle walking and also fulfilling many of the benefits listed above. These controllers use foot placements as a control mechanism to stabilize the under-actuated horizontal CoM dynamics. At no point do they attempt to directly control the CoM instantaneous state. Instead, they calculate a control policy in which the foot location is a feedback weighted sum of the sensed CoM state. Our dynamic locomotion control policy falls into this category of controllers albeit using a particular CoM feedback gain matrix based on the concept of time-to-velocity-reversal (TVR) (Kim et al., 2014). Another important dynamic locomotion control strategy relies on the concept of hybrid zero dynamics (HZD) (Westervelt et al., 2007). HZD considers an orbit for dynamic locomotion and a feedback control policy that warranties asymptotic stability to the orbit (Hartley et al., 2017; Hereid et al., 2016).Although these two lines of dynamic walking controls have had an enormous impact in the legged locomotion field, they have not been extended yet to full humanoid systems. In particular, humanoid systems employing task-based whole-body control strategies require closing the gap with the above dynamic locomotion methods. This is precisely the main objective of this article.

The main contribution of this article is to achieve unsupported dynamic walking of passive-ankle and full humanoid robots using the whole-body control method. To do so, we: (1) devise a new task-based whole-body locomotion controller (WBLC) that fulfills maximum tracking errors and significantly reduces contact jerks; (2) conduct an uncertainty analysis to improve the robot mechanics and controls; (3) integrate the whole-body control method with our dynamic locomotion planner into two experimental bipeds robots; and (4) experiment extensively with unsupported dynamic walking such as throwing balls, pushing a biped, or walking in irregular terrains.

One important improvement we have incorporated in our control scheme is to switch from joint torque control to joint position control. This low-level control change is due to the lessons we have learned regarding the overall system performance difference between low-level joint control versus torque control. Namely that joint position control used in this article works better than a joint torque control (Kim et al., 2016). In addition, our decision to use a low-level joint-level control is supported by previous studies that torque control reduces the ability to achieve a high-impedance behavior (Calanca et al., 2016), which is needed for achieving dynamic biped locomotion with passive-ankle bipeds. Indeed, switching to joint position control has been a strong performance improvement to achieve the difficult experimental results.

From the uncertainty analysis of our TVR dynamic locomotion planner, we found that to achieve stable locomotion the robot requires higher position tracking accuracy than initially expected. Our uncertainty analysis concludes that the landing foot positions need to be controlled within a 1 cm error and the CoM state needs to be estimated within a 0.5 cm error. Both the robot's posture control and the swing foot control require high tracking accuracy. For this reason, we remove the torque feedback in the low-level controller and instead impose a feedforward current command to compensate for whole-body inertial, Coriolis, and gravitational effects. However, this is not enough to overcome friction and stiction of the joint drivetrain. To overcome this issue, we introduce a motor position feedback controller (Pratt et al., 2004).

Next, the low-level joint commands are computed by our proposed WBLC. WBLC consists of two sequential blocks: a kinematics-level whole-body control, hereafter referred to as KinWBC, and a dynamics-level whole-body controller (DynWBC). The first block, KinWBC, computes joint position commands as a function of the desired operational task commands using feedback control over the robot's body posture and its foot position.

Given these joint position commands, DynWBC computes feedforward torque commands while incorporating gravity and Coriolis forces, as well as friction cone constraints at the contact points. One key characteristic of DynWBC is the formulation of reduced jerk torque commands to handle sudden contact changes. Indeed, in our formulation, we avoid formulating contacts as hard constraints (Herzog et al., 2016; Saab et al., 2013; Wensing and Orin, 2013) and instead include them as a cost function. We then use the cost weights associated with the contacts to change behavior during contact transitions in a way that it significantly reduces movement jerk. For instance, when we apply heavy cost weights to the contact accelerations, we effectively emulate the effect of contact constraints. During foot detachment, we continuously reduce the contact cost weights. By doing so, we accomplish smooth transitions as the contact conditions change. An approach based on whole-body inverse dynamics has been proposed for smooth task transitions (Salini et al., 2011), but has not been proposed for contact transitions such as ours, neither has it been implemented in experimental platforms.

The above WBLC and joint-level position feedback controller can achieve high-fidelity real-time control of bipeds and humanoid robots. For locomotion control, we employ the TVR planner presented by Kim et al. (2016). We use the TVR planner to update foot landing locations at every step as a function of the CoM state. We do so by planning in the middle of leg swing motions. By continuously updating the foot landing locations, bipeds accomplish dynamic walking that is robust to control errors and to external disturbances. The capability of our walking controller is tested extensively in a passive-ankle biped robot

and in a quasi-passive-ankle lower-body humanoid robot. By relying on foot landing location commands, our control scheme is generic to various types of bipeds and therefore, we can accomplish similar walking capabilities across various robots by simply switching the robot parameters. To demonstrate the generality of our controller, we test not only two experimental bipeds, but also a simulation of other humanoid robots.

Indeed, experimental validation is a main contribution of this article. The passive-ankle biped, Mercury, is used for extensive testing of dynamic balancing, directional walking, and rough terrain walking. We also deploy the same methods to our new biped, DRACO, and accomplish dynamic walking within a few days after the robot had its joint position controllers developed. Such timely deployment showcases the robustness and versatility of the proposed control framework.

This article is organized as follows. Section 3 introduces our robot hardware and its characteristic features. In Section 4, we explain the control framework consisting of the dynamic locomotion planner, WBLC, and the joint-level position controller. Section 5 explains how the measurement noise and landing location error affect the stability of our dynamic locomotion controller, and analyzes the required accuracy for state estimation and swing foot control to asymptotically stabilize bipeds. In Section 6, we address implementation details. Section 7 discusses extensively experimental and simulation results. Finally, Section 8 concludes and summarizes our works.

2. Related work

ATRIAS (Hartley et al., 2017; Rezazadeh et al., 2015) is the closest example to our proposed work for this article. It is one of the first passive-ankle biped robots that is able to dynamically balance and walk unsupported. The key difference is that our framework focuses on methods applicable to whole-body humanoid robot control applied to passive-ankle bipeds. Both our proposed dynamic locomotion planner and WBLC are novel and unique. In particular, our dynamic locomotion planner is based on the concept of TVR and incorporate an uncertainty analysis that drive the mechanical and control design of robots to improve their performance. In addition, our WBLC evolves from a long line of research on task-based whole-body humanoid robot control from our group by incorporating tools to significantly reduce contact-induced movement jerk.

There are several pioneering examples of dynamic biped locomotion: ATLAS (Boston Dynamics, 2018) and ASIMO (Honda, 2011). It is difficult to tell how much these robots rely on ankle actuation and foot support because of the lack of published work. It is also impossible for us to tell what kind of dynamic locomotion planners and whole-body control methods are implemented.

Passive walking robots (Collins et al., 2005; McGeer, 1990) also fall into the dynamic locomotion category. These studies shed light on the important aspects of biped locomotion, but do not provide direct application for feedback control related to our methods. On the other hand, the progress made in actuated planar biped locomotion is impressive. Raibert et al. (1989) and Sreenath et al. (2012) presented biped robots running and their capability to recover from disturbances on irregular terrains. However, there is an obvious gap between supported (or constrained) locomotion and unsupported walking. Raibert et al. (1984) presented unsupported single-leg hopping, which is a remarkable accomplishment. In addition to the strong contribution in dynamic locomotion of that work, the study omitted several important aspects of unsupported biped locomotion such as body posture control, continuous interaction of the stance leg through the ground contact phases, and disturbances from the other limbs' motion, which are a focus of our article.

3. Mercury experimental robot

The methods described in this article have been tested extensively in two biped platforms. Most experiments are performed in our biped robot, Mercury, which we describe here. An additional experiment was performed on a new biped, called DRACO, which is described in Ahn et al. (2019). Mercury has six actuators that control the hip abduction/adduction, flexion/extension, and knee flexion/extension joints. Mercury uses series-elastic actuators (SEAs), which incorporate a spring between the drivetrains and the joint outputs. The springs protect the drivetrains from external impacts and are used for estimating torque outputs at the joints. In addition, Mercury went through significant hardware upgrades from our previous robot, Hume (Kim et al., 2016). In this section, we provide an overview of our system and discuss the upgrade. We also explain similarities and differences with respect to other humanoid robots in terms of mass distribution.

3.1. Robot configuration

Figure 1 shows the sensing system and configuration of Mercury. Mercury's configuration starts from a set of virtual joints fixed to ground, representing the floating base dynamics of the robot. The end joint of the 6-DoF virtual joint is attached to the physical base of Mercury. The orientation of the virtual ball joint is represented by a quaternion and its angular velocity is represented by the space so(3) with respect to the local base frame. The actuated joints start from the right hip abduction/adduction and goes down to the hip flexion/extension, and knee flexion/extension joints. Then, the joint labels continue on to the left leg starting also at the hip joint. Three LED sensors are attached to the front of robot's body frame to estimate the robot's linear velocity and its global position via MoCap. In addition, we also estimate the relative robot position using joint encoder data with respect to the stance foot. This last sensing procedure is partially used to control foot landing locations, and

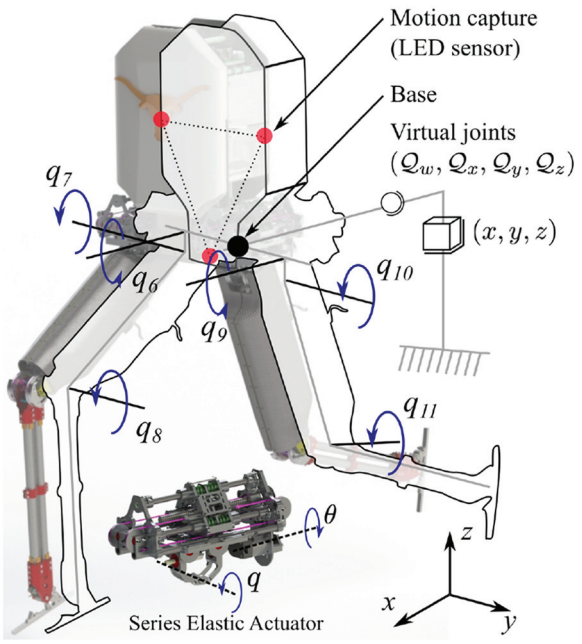


Fig. 1. Configuration of Mercury. To represent the floating base dynamics, we connect virtual joints at the base of Mercury. The virtual joints consist of three prismatic joints and a ball joint that is expressed as a quaternion. Each leg has an actuated abduction/adduction (q_6, q_9), hip flexion/extension (q_7, q_{10}), and knee flexion/extension (q_8, q_{11}) joints. Lastly, three LED sensors are attached on the front of the robot's body to estimate the velocity of its physical base.

therefore, the reference frame changes every time the robot switches contact.

Mercury's SEA actuators were built in 2011 by Meka, each having three encoders to measure joint position, spring deflection, and motor position. An absolute position encoder is used to measure the joint output position, q_j, while a low-noise quadrature encoder measures motor position, θ. Joint position and joint velocity sensing can be done either using the absolute encoder or via applying a transmission ratio transformation on the motor's quadrature encoder data (q_m). In our experiment, we use the absolute encoders to obtain joint positions and motor quadrature encoders to obtain joint velocities. The transmission ratio of all Mercury's joints has a constant value except for the abduction/adduction joints which are non-constant. The constant ratio occurs for transmissions consisting of a pulley mechanism with constant radius. On the other hand, the hip abduction/adduction joints consist of a spring cage directly connected to the joints, which results in a change of the moment arm. To account for this change, we use a look-up table mapping the moment arm length with respect to the joint position.

3.2. Hardware Upgrades

The original biped, Hume, was mostly built in 2011 by Meka as a custom robot for our laboratory. It had several limitations that made dynamic locomotion difficult. It had a low-performance inertial measurement unit (IMU), which made it difficult to control the robot's body orientation. Hume's legs were not strong enough causing buckling of the structure when supporting the robot's body mass. Owing to this structural buckling, the estimated foot positions obtained from the joint encoders was off by 5 cm from their actual positions. We estimated this error by comparing the joint encoder data with the MoCap system data. Hume terminated its legs with cylindrical cups that would make contact with the ground. These cups had a extremely small contact surface with the ground. During walking, Hume suffered from significant vertical rotation, i.e., yaw rotation, owing to the minimal contact of its supporting foot with the ground. All of these problems, i.e., structural buckling, poor IMU sensor, and small contact surfaces, prevented Hume from accomplishing stable walking. Therefore, for the proposed work, we have significantly upgraded the robot in all of these respects and changed its name to Mercury.

To improve on state estimation, we upgraded the original IMU, a Microstrain 3DM-GX3-25-OEM, to a tactical one, a STIM-300 (Figure 2(a)). Both IMUs are microelectromechanical system (MEMS)-based but the bias instability of the tactical IMU is only 0.0087 rad/h. Such low-bias noise allows us to estimate the robot's body orientation by simply integrating over the angular velocity from the initial orientation. Another problem we were facing with our original biped is the aging electronics, originally built by Meka in 2011. For this reason, all control boards (Figure 2(d)) have been replaced with new embedded electronics manufactured by Apptronik. These new control boards are equipped with, a powerful micro-controller, a TI Delfino, that performs complex computations with low latency for signal processing and control. The control boards are installed in a special board case (Figure 2(b)) holding safely all cables connected to the board. This wiring routing and housing detail is important because Mercury hits the ground hard when walking in rough terrains and performs experiments by being hit by people and balls. It secures signal and power cables to enable solid signal communications.

Third, we manufactured carbon shells (Figure 2(c)) to reinforce the thigh linkages. We also redesigned the robot's shank to increase structural stiffness by including two carbon fiber cylinders as supporting linkages. In addition, we designed new passive feet in the form of thin and short prisms that are a few centimeters long. The feet pivot about a pin fulcrum which connects in parallel to a spring between the foot support and the pivoting ankle. A contact switch is located on the front of the foot and engages when the foot makes contact with the ground (see Figure 2(e) for mechanical details). These contact switches are used to terminate swing foot motion controls when the swing foot touches the ground earlier than anticipated. The main purpose of the line feet is to prevent yaw rotations of the entire robot turning around the supporting foot. Previously, our robot had quasi-pointed feet, which caused the robot's

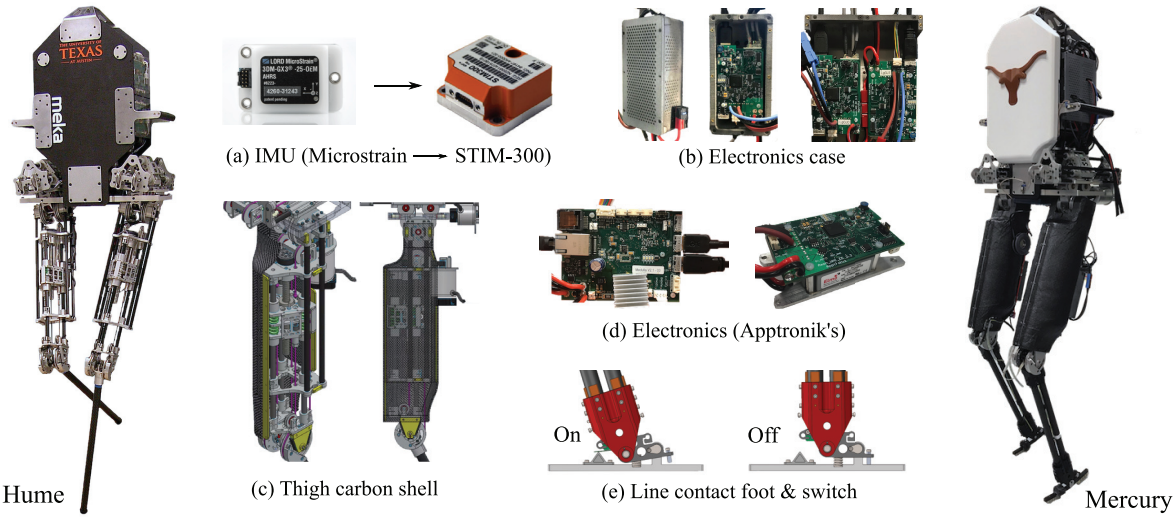


Fig. 2. Hardware upgrades of Mercury. (a) The IMU was upgraded to the Sensonor's STIM-300, which has low angular velocity drift and bias, enabling accurate orientation estimates even with simple forward integration. (b) The on-board electronics have been installed with cases to secure the electric cables in place. Keeping the cables in place significantly reduces losses of connections and cable damage during robot operations. (c) Carbon fiber cases were installed on Mercury's thighs to increase structural stiffness. (d) All of the embedded electronics were replaced with Apptronik's Medulla and Axon boards that come with a variety of low-level controllers for SEAs. (e) Spring-loaded passive ankles with limit switches were also added to limit the uncontrollable yaw body rotation and detect ground contacts.

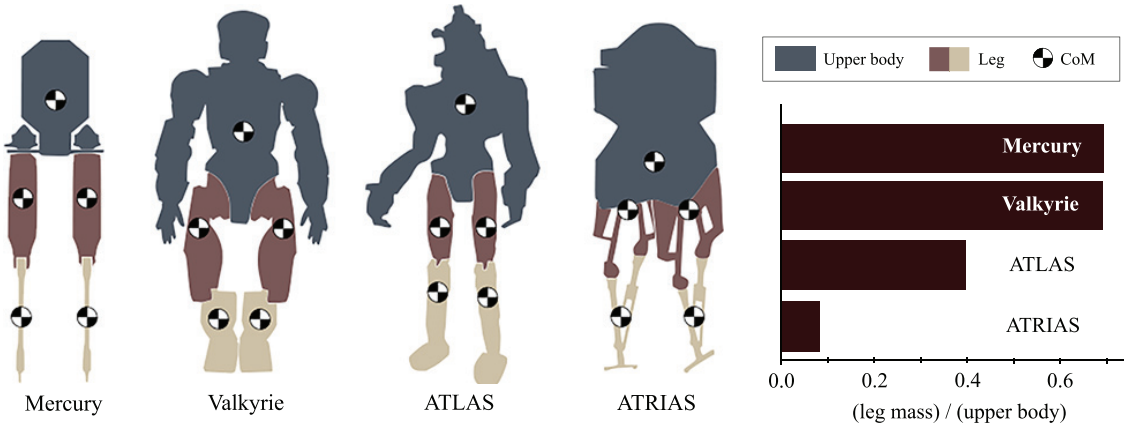


Fig. 3. Mass distribution of various biped robots. Note that the robots shown here are not scaled equally. They are shown to compare the relative location of their CoM. The CoM locations are superimposed on the upper-body and leg links of each robot. The bar graph depicts the ratio of the total leg mass over the upper body mass. Note that ATRIAS has a mass distribution different than typical humanoids by having the torso CoM near the hip joint and a small leg-to-upper-body-mass ratio.

heading to turn due to any vertical moments. The mechanical line contacts provided by the passive feet interact with the ground contacts as a friction moment preventing excessive body rotations.

3.3. Challenges in passive-ankle locomotion

To discuss the locomotion challenges presented by Mercury, it is necessary to discuss the mass distribution of Mercury against other bipeds (Figure 3). The robots' inertia information used for this comparison is taken from open-source robot models found in public repositories.See https://github.com/openhumanoids (Valkyrie), https://github.com/dartsim/ (ATLAS), and https://github.com/siravinash/atrias-matlab (ATRIAS). Mercury was built using linear SEAs to minimize actuator friction. It is known that ball screws have less friction than other gears, such as harmonic drives. The use of linear SEAs led to a leg design based on a serial configuration, which results on a mass distribution with significant distal mass. Thus, Mercury's mass distribution is similar to anthropomorphic humanoids such as Valkyrie (Radford et al., 2015a) or Atlas (Kuindersma

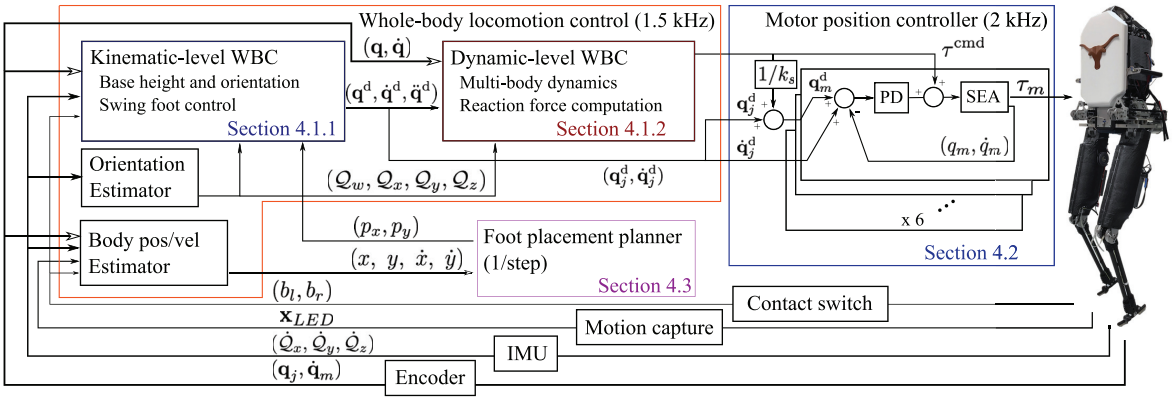


Fig. 4. Diagram of the WBLC control scheme. Our WBLC controller has a cascaded structure with three feedback loops. Here b_l and b_r are left and right foot contact signals, and $\mathbf{x}_{\text{LED}}$ is the position of the sensed MoCap LED markers. (1) An inverse kinematics controller computes joint positions and their first two derivatives based on desired operational space tasks. (2) An inverse dynamics whole-body controller computes contact-consistent torque commands to handle robot dynamics subject to unilateral constraints. (3) Low-level PD controllers on motor positions with feedforward currents from the computed torques are used to achieve the desired joint configurations. A TVR footstep planner plans foot landing locations, one per step at the midpoint of swing leg trajectories. Note that $\mathbf{q}_m$ are joint positions computed from the motor positions via a transmission ratio and $\mathbf{q}_j$ are joint positions measured by absolute joint encoders.

et al., 2015). These robots have (1) a torso CoM located around the center of its body, and (2) a ratio between the total leg mass and the torso mass of about 40%. As such, the effects due to the distal leg mass become non-negligible during swing phases. On the other hand, ATRIAS (Hubicki et al., 2018) has a mass distribution optimized to be a mechanical realization of the inverted pendulum model, which is designed to aid with the implementation of locomotion controllers. Unlike other humanoid robots, ATRIAS's torso CoM location is close to the hip joints and the ratio between the total leg mass to the torso is negligible, which is less than 0.1.

Although Mercury and ATRIAS are similar in their lack of ankle actuation and number of DoFs, the difference in mass distributions creates difficulties on locomotion control. As ATRIAS has its torso CoM close to the hip joint axis, the link inertia reflected to the hip joint is small, which reduces the difficulty of controlling the robot body's orientation. In contrast, the CoM of Mercury and other humanoid robots mentioned above are located well above the hip joint, which creates a larger moment arm and increases the difficulty of body orientation control.

Next, because ATRIAS has negligible leg mass compared with its body, body perturbations caused by the swing leg are also negligible. However, Mercury, having significant leg mass, causes noticeable body perturbations during the swing phase. Thus, it becomes necessary for Mercury to have a whole-body controller that can compensate against Coriolis and gravitational forces introduced by the swing leg to maintain desired body configurations, follow inverted pendulum dynamics, and control the swing foot to desired landing locations. Overall, in addition to Mercury's SEAs and lack of ankle actuation, its mass distribution makes it more difficult to control.

4. Locomotion control architecture

Our proposed control architecture (Figure 4) consists of three components: (1) a WBLC, which coordinates joint commands based on desired operational space goal trajectories, (2) a set of joint-level feedback controllers, which execute the commanded joint trajectories; and (3) a dynamic locomotion planner for passive-ankle bipeds which generates the foot landing locations based on TVR considerations. In this section, we describe the details of these layers as well as their interaction.

4.1. WBLC

Many WBCs include a robot dynamic model to compute joint torque/force commands to achieve desired operational space trajectories. If we had ideal motors with perfect gears, the computed torque commands of a WBC could be sent out as open-loop motor currents. However, excluding some special actuator designs (Wensing et al., 2017), it is non-trivial to achieve the desired torque/force commands using open-loop motor currents because most actuators have high friction and stiction in their drivetrains. One established way to overcome drivetrain friction is to employ torque/force sensor feedback at the joint level. However, negative torque/force feedback control is known to reduce the maximum achievable close-loop stiffness of joint controllers (Calanca et al., 2016). In addition, torque/force feedback controllers used in combination with position control are known to be more sensitive to contact disturbance and time delay. Therefore, we need a solution that addresses all of these limitations.

Another consideration is related to the task space impedance behavior that is needed to achieve dynamic walking.

Our observation is that a high impedance behavior in task space is preferred for dynamic walking because: (1) the foot landing location must be fairly accurate to stabilize the biped; (2) the swing leg must be able to overcome external disturbances; and (3) the robot's body posture needs to suppress oscillations caused by the effect of moving limbs or other disturbances. High stiffness control of robots with sizable mechanical imperfections is the only way to achieve stable passive-ankle biped walking despite making them less compliant with respect to the terrain.

To accomplish high gain position control, we have opted to remove sensor-based torque feedback at the joint level and replace it with motor position feedback control. Our observation is that this change significantly reduces the effect of the imperfect mechanics and achieves higher position control bandwidth than using torque feedback. In addition to the joint position commands, the desired torque commands computed via WBC are incorporated as feedforward motor current commands. Thus, to combine motor position and feedforward motor current commands for dynamic locomotion, we devise a new WBC formulation that we call WBLC.

WBLC is sequentially implemented with two control blocks. The first block is a kinematic-level WBC (KinWBC) that computes joint position, velocity, and acceleration commands. KinWBC does not rely on a dynamical model of the robot, instead it relies only on a kinematics model to coordinate multiple prioritized operation space tasks. The second block, called the dynamic-level WBC (DynWBC), takes the joint commands from KinWBC and computes the desired torque commands that are consistent with the robot dynamics and the changing contact constraints. The output of WBLC is therefore composed of desired joint torque, position, and velocity commands, which are sent out to the joint-level feedback controllers.

4.1.1. KinWBC. We first formulate a kinematic whole-body controller to obtain joint commands given operational space commands. The basic idea is to compute incremental joint positions based on operational space position errors and add them to the current joint positions. This is done using null-space task prioritization as follows:

$$\Delta\mathbf{q}_1 = \boldsymbol{J}_1^{\dagger}(\mathbf{x}_1^{\text{des}} - \mathbf{x}_1) \tag{1}$$

$$\Delta\mathbf{q}_2 = \Delta\mathbf{q}_1 + \boldsymbol{J}_{2|pre}^{\dagger}(\mathbf{x}_2^{\text{des}} - \mathbf{x}_2 - \boldsymbol{J}_2\Delta\mathbf{q}_1) \tag{2}$$

$$\vdots$$

$$\Delta\mathbf{q}_i = \Delta\mathbf{q}_{i-1} + \boldsymbol{J}_{i|pre}^{\dagger}(\mathbf{x}_i^{\text{des}} - \mathbf{x}_i - \boldsymbol{J}_i\Delta\mathbf{q}_{i-1}) \tag{3}$$

where $\boldsymbol{J}_i$, $\mathbf{x}_i^{\text{des}}$, and $\Delta\mathbf{q}_i$ are the ith task Jacobian, a desired position of the ith task, and the change of joint configuration related to the ith task iteration. The $\{\cdot\}^{\dagger}$ denotes an singular value decomposition (SVD)-based pseudo-inverse operator in which small singular values are set to 0. Note that there is no feedback gain terms in this formulation, which can be interpreted as gains being equal to unity. In addition, the prioritized Jacobians take the form:

$$\boldsymbol{J}_{i|pre} = \boldsymbol{J}_i\boldsymbol{N}_{i-1} \tag{4}$$

$$\boldsymbol{N}_{i-1} = \boldsymbol{N}_{1|0}\cdots\boldsymbol{N}_{i-1|i-2} \tag{5}$$

$$\boldsymbol{N}_{i-1|i-2} = \boldsymbol{I} - \boldsymbol{J}_{i-1|pre}^{\dagger}\boldsymbol{J}_{i-1|pre} \tag{6}$$

$$\boldsymbol{N}_0 = \boldsymbol{I} \tag{7}$$

Then, the joint position commands can be found with

$$\mathbf{q}^{\text{d}} = \mathbf{q} + \Delta\mathbf{q} \tag{8}$$

where $\Delta\mathbf{q}$ is joint increment computed in the ith task in (3). In addition, the joint velocity and acceleration for every task iteration can be computed as

$$\dot{\mathbf{q}}_i^{\text{d}} = \dot{\mathbf{q}}_{i-1}^{\text{d}} + \boldsymbol{J}_{i|pre}^{\dagger}\left(\dot{\mathbf{x}}^{\text{des}} - \boldsymbol{J}_i\dot{\mathbf{q}}_{i-1}^{\text{d}}\right) \tag{9}$$

$$\ddot{\mathbf{q}}_i^{\text{d}} = \ddot{\mathbf{q}}_{i-1}^{\text{d}} + \boldsymbol{J}_{i|pre}^{\dagger}\left(\ddot{\mathbf{x}}^{\text{des}} - \dot{\boldsymbol{J}}_i\dot{\mathbf{q}} - \boldsymbol{J}_i\ddot{\mathbf{q}}_{i-1}^{\text{d}}\right) \tag{10}$$

Finally, the joint commands, $\mathbf{q}^{\text{d}}$, $\dot{\mathbf{q}}^{\text{d}}$, and $\ddot{\mathbf{q}}^{\text{d}}$ are sent out to the block, DynWBC. We note that $\mathbf{q}$ is the full configuration of the robot containing both floating base and actuated joints. The task breakdown and their priorities used in our walking experiments are explained in Section 6.2.

4.1.2. DynWBC. Given joint position, velocity, and acceleration commands from the KinWBC, the DynWBC computes torque commands while considering the robot dynamic model and various constraints. The optimization algorithm to compute torque commands in DynWBC is as follows:

$$\min_{\mathbf{F}_r, \ddot{\mathbf{x}}_c, \boldsymbol{\delta}_{\ddot{\mathbf{q}}}} \mathbf{F}_r^{\top}\boldsymbol{W}_r\mathbf{F}_r + \ddot{\mathbf{x}}_c^{\top}\boldsymbol{W}_c\ddot{\mathbf{x}}_c + \boldsymbol{\delta}_{\ddot{\mathbf{q}}}^{\top}\boldsymbol{W}_{\ddot{\mathbf{q}}}\boldsymbol{\delta}_{\ddot{\mathbf{q}}} \tag{11}$$

$$\text{s.t.} \quad \boldsymbol{U}\mathbf{F}_r \geqslant \mathbf{0} \tag{12}$$

$$\boldsymbol{S}\mathbf{F}_r \leqslant \mathbf{F}_{r,z}^{\max} \tag{13}$$

$$\ddot{\mathbf{x}}_c = \boldsymbol{J}_c\ddot{\mathbf{q}} + \dot{\boldsymbol{J}}_c\dot{\mathbf{q}} \tag{14}$$

$$\boldsymbol{A}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g} = \begin{pmatrix} \mathbf{0}_{6\times 1} \\ \boldsymbol{\tau}^{\text{cmd}} \end{pmatrix} + \boldsymbol{J}_c^{\top}\mathbf{F}_r \tag{15}$$

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}^{\text{cmd}} + \boldsymbol{\delta}_{\ddot{\mathbf{q}}} \tag{16}$$

$$\ddot{\mathbf{q}}^{\text{cmd}} = \ddot{\mathbf{q}}^{\text{d}} + k_d(\dot{\mathbf{q}}^{\text{d}} - \dot{\mathbf{q}}) + k_p(\mathbf{q}^{\text{d}} - \mathbf{q}) \tag{17}$$

$$\boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}^{\text{cmd}} \leqslant \boldsymbol{\tau}_{\max} \tag{18}$$

In Equation (12), $\boldsymbol{U}$ computes normal and friction cone forces as described by Bouyarmane et al. (2018), and $\mathbf{F}_r$ represents contact reaction forces. Equation (13) introduces the upper bounds on the normal reaction forces to facilitate smooth contact transitions. As mentioned previously, this upper bound is selected to decrease when the foot contacts detach from the ground and increase again when the foot

makes contact. Equation (14) is for the contact points' acceleration and $\boldsymbol{J}_c$ is the Jacobian coinciding with the contact points.

Equation (15) models the full-body dynamics of the robot including the reaction forces. Here $\boldsymbol{A}$, $\mathbf{b}$, and $\mathbf{g}$ are the generalized inertia, Coriolis, and gravitational forces, respectively. The diagonal terms of the inertia matrix include the rotor inertia of each actuator in addition to the linkage inertia. The rotor inertia is an important inclusion to achieve good performance. Equation (16) shows the relaxation of the joint commands, $\ddot{\mathbf{q}}^{\text{cmd}}$, by the term, $\boldsymbol{\delta}_{\ddot{\mathbf{q}}}$. We include this relaxation because of two reasons. First, the KinWBC specifies virtual joint acceleration that cannot be perfectly attained. Second, the torque limit on the above optimization can prevent achieving the desired joint acceleration. Equation (17) shows how the KinWBC's joint commands are used to find desired acceleration commands. Here, $\mathbf{q}^{\text{d}}$, $\dot{\mathbf{q}}^{\text{d}}$, and $\ddot{\mathbf{q}}^{\text{d}}$ are the computed commands from KinWBC. Equation (18) represents torque limits.

In turn, these computed torque commands are sent out as feedforward motor current commands to the joint-level controllers. One key difference with other QP formulations for whole-body control is that we do not use the null space operators of the contact constraints nor do we use a null velocity or acceleration assumption to describe the surface contacts of the robot with the ground. Instead, contact interactions are addressed with contact acceleration terms in the cost function regulated with weighting matrices that effectively model the changes in the contact state. This new term is particularly important because traditional modeling of contacts as hard constraints causes torque command discontinuities owing to sudden contact switches. As such, we call our formulation reduced "jerk" whole-body control. We note that our formulation is the first attempt that we know of to use WBC for unsupported passive-ankle dynamic locomotion in experimental bipeds. Contact changes in passive-ankle biped locomotion are far more sudden than changes on robots that control the horizontal CoM movement. Our proposed formulation emerges from extensive experimentation and comparison between QP-based WBC formulations using hard contact constraints versus soft constraints as proposed. We report that the above formulation has empirically shown to produce rapidly changing but smooth torque commands than using WBCs with hard constraints.

To achieve smooth contact switching, the contact Jacobian employed above includes both the robot's feet contacts even if one of them is not currently in contact. As a result $\mathbf{x}_c$ in Equations (11) and (14) contains the contact coordinates of both the right and left feet, regardless of the contact phase. As mentioned previously, we never set foot contact accelerations to be zero even if they are in contact. Instead, we penalize foot accelerations in the cost function depending on whether they are in contact or not using the weight $\boldsymbol{W}_c$. When a foot is in contact, we increase the values of $\boldsymbol{W}_c$ for the block corresponding to the contact. Similarly, we reduce the values of the weights when the foot is removed from the contact. At the same time, we increase the weight $\boldsymbol{W}_r$ for the swing foot and reduce the upper bound of the reaction force $\mathbf{F}_{r,z}^{\max}$. In essence, by smoothly changing the upper bounds, $\mathbf{F}_{r,z}^{\max}$, and weights, $\boldsymbol{W}_r$ and $\boldsymbol{W}_c$, we practically achieve jerk-free walking motions. The concrete description for the weights and bounds used in our experiments are explained in Section 6.2.

4.2. Joint-level controller

Each actuated joint has an embedded control board that we use to implement the motor position PD control with feedforward torque inputs:

$$\tau_m = \tau^{\text{cmd}} + k_p\left(q_m^{\text{d}} - q_m\right) + k_d\left(\dot{q}_j^{\text{d}} - \dot{q}_m\right) \tag{19}$$

where τ_m and τ^{cmd} are the desired motor torque and computed torque command, the latter is obtained from Equation (15) in the optimization problem. Thus, τ^{cmd} acts as the feedforward control input. Here $\dot{q}_j^{\text{d}}$ is the desired joint velocity computed from the KinWBC. It is obtained by applying the iterative algorithm in Equation (9). The term q_m^{d} is a desired motor position command and is computed using the following formula,

$$q_m^{\text{d}} = q_j^{\text{d}} + \frac{\tau^{\text{cmd}}}{k_s} \tag{20}$$

where k_s is the spring constant of each SEA joint. Here q_j^d is obtained via the iterative algorithm shown in Equations (1)–(8). We incorporate this spring deflection consideration because the computation of joint positions from motor positions, q_m, considers only the transmission ratio, N, but the spring deflection is ignored in the computation.

4.3. TVR planner

At every step, a TVR planner computes foot placements as a function of the CoM state, i.e., its position and velocity. This is done around the middle of the swing foot motion. Our TVR planner operates with the principle of reversing the CoM velocity every step and it can be shown that the CoM movement is asymptotically stable. The original method was presented in our previous paper (Kim et al., 2016). In this article, we use a simplified version of TVR which considers a constant CoM height. This consideration has been beneficial on various experimental results across multiple biped robotics platforms explored in this article. In Appendix B, we explain the difference of our planner and those proposed by Raibert et al. (1984), Koolen et al. (2012), and Rezazadeh et al. (2015).

5. Uncertainty analysis of the planner

One of the biggest challenges in unsupported passive-ankle dynamic locomotion is to determine what control accuracy is needed to effectively stabilize a biped. Given that a

passive-ankle biped robot cannot use ankle torques to control the robot's CoM movement, foot position accuracy, state estimation, and other related considerations become much more important in achieving the desired dynamic behavior. For instance, the CoM dynamics emerging from passive-ankle behavior evolves exponentially with time, pointing out the need to determine the tolerable foot position and body estimation errors. In this section, we develop the tools to explicitly quantify the required accuracy to achieve asymptotically stable passive-ankle dynamic locomotion.

As mentioned previously, our TVR locomotion planner observes the CoM position and velocity states and computes a foot landing location. For our analysis and experimentation, we enforce a constant CoM height constraint. Our reliance on linear inverted pendulum (LIP) model enables a straightforward uncertainty analysis given noisy CoM state observations and landing location errors under kinematic constraints.

5.1. Formulation of the planner

Our TVR planner relies on the LIP model:

$$\ddot{x} = \frac{g}{h}(x - p) \tag{21}$$

where g is the gravitational acceleration, h is the constant CoM height value, and p is the foot landing location which acts as a stabilizing input for reversing the CoM dynamics at every step. More concretely, the TVR planner aims to reverse the CoM velocity after a set time duration t' by computing a new stance foot location, p. Note that Equation (21) is linear so it has an exact solution for the CoM state, $\mathbf{x}(t)$. Thus, for a given p, the CoM state after a desired swing time T can be described as a discrete system where k corresponds to the kth walking step of the robot:

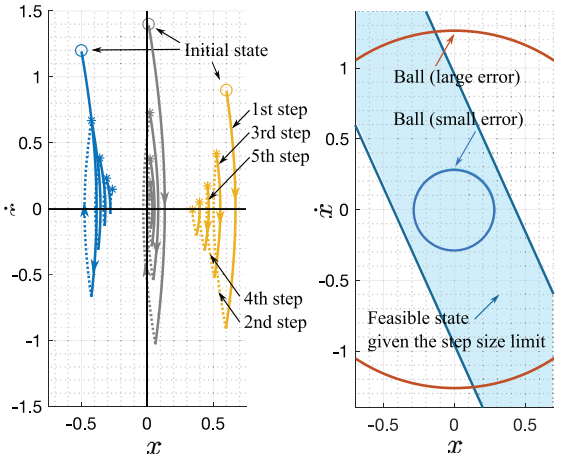
$$\mathbf{x}_{k+1} = \boldsymbol{A}\mathbf{x}_k + \boldsymbol{B}p_k \tag{22}$$

$$\boldsymbol{A} = \begin{bmatrix} \cosh(\omega T) & \omega^{-1}\sinh(\omega T) \\ \omega \sinh(\omega T) & \cosh(\omega T) \end{bmatrix} \tag{23}$$

$$\boldsymbol{B} = \begin{bmatrix} 1 - \cosh(\omega T) \\ -\omega \sinh(\omega T) \end{bmatrix} \tag{24}$$

where $\omega = \sqrt{g/h}$. The system above can be straightforwardly obtained by applying known second-order linear ordinary differential equation (ODE) techniques to Equation (21). Next, let p_k correspond to the foot location of the kth step in a sequence of steps. Our TVR planner is based on the objective of finding a p_k, which reverse the CoM velocity at every step. Letting the velocity component (bottom row) of Equation (22) be zero after the desired reversal time, $t' < T$, results into the quality,

$$0 = [\,\omega \sinh(\omega t') \quad \cosh(\omega t')\,]\mathbf{x}_k - \omega \sinh(\omega t')p_k \tag{25}$$



(a) Phase trajectory (b) Uncertainty analysis

Fig. 5. Phase plot and uncertainty analysis. In (a), the phase trajectories of eight steps from three initial states converging toward the origin are shown. In (b), the region of uncertainty is encircled by balls. States within the blue region and outside of the ball radius are kinematically feasible and asymptotically stable, respectively. The ball radius increases when the system has large errors in state observation and control input.
Note: Colour version of the figure is available online.

Solving for p_k in the above equation will result in the foot landing location policy that reverses CoM velocity after t'. With the CoM velocity being reversed after every step, an additional bias term, κ, is added to steer the robot toward the origin. Further details about κ can be found in Kim et al. (2016). Solving for Equation (25) and including the additional κ term, we obtain

$$p_k = \begin{bmatrix} 1 & \omega^{-1}\coth(\omega t') \end{bmatrix}\mathbf{x}_k + \begin{bmatrix} \kappa & 0 \end{bmatrix}\mathbf{x}_k \tag{26}$$

Incorporating the above feedback policy into Equation (22), we obtain the closed-loop dynamics,

$$\mathbf{x}_{k+1} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})\mathbf{x}_k, \tag{27}$$

$$\boldsymbol{K} = \begin{bmatrix} (1+\kappa) & \omega^{-1}\coth(\omega t') \end{bmatrix} \tag{28}$$

Note that the control policy in Equation (27) has a simple PD control form; therefore, applying standard linear stability methods for PD control, the planner parameters, (κ, t'), can be tuned to achieve magnitudes such that the closed-loop eigenvalues of $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$ are smaller than 1. In our case, we chose eigenvalues with magnitude equal to 0.8. As our desired behavior is to take multiple small steps toward a desired reference position rather than a single big step, the eigenvalue magnitudes are intentionally set to be close to one rather than zero. The resulting motion (simulated numerically) in Figure 5(a), shows the asymptotically converging trajectories in the phase plot.

5.2. Uncertainty analysis

During experimental walking tests, we observed notable body position and landing location errors due to the

deflection of the mechanical linkages of the robot. We note that in our attempt to make Mercury a lightweight robot, we designed body and leg structures made of thin aluminum pieces and carbon fiber structures. In particular, the lower and upper legs of Mercury are constructed using carbon fiber without further rigid support. In addition, the abduction and flexion hip joints contain drivetrains made out of thin aluminum with pin joints that deflect when a contact occurs. Rather than focusing on the effect of these existing mechanical deformations, we decided to focus on the maximum errors that our dynamic locomotion controller can tolerate. After we found the maximum tolerances, we went back to the robot's mechanical design and replaced hip joints and the leg linkages to be significantly more rigid in order to fulfill the maximum tolerances. Therefore, our uncertainty analysis has been fundamental to drive the new mechanical structure on the original biped hardware to achieve the desired performance.

To quantify the acceptable errors for our TVR planner, we perform here an analysis of stability borrowing ideas from robust control (Bahnasawi et al., 1989). We apply some assumptions to simplify our analysis: (1) the robot's step size is limited to 0.5 m based on an approximated leg kinematic limits; and (2) state-dependent errors are ignored.

For our analysis, we model foot landing location errors (presumably resulting from mechanical deflection and limited control bandwidth) with a scalar term, η. On the other hand, we model CoM state estimation errors as a vector of position and velocity errors, $\boldsymbol{\delta}$. Based on these error variables, we extend the dynamics of Equation (22) to be

$$\begin{aligned} \mathbf{x}_{k+1} &= \boldsymbol{A}\mathbf{x}_k + \boldsymbol{B}(p_k + \eta) \\ p_k &= \boldsymbol{K}(\mathbf{x}_k + \boldsymbol{\delta}) \end{aligned} \tag{29}$$

In order to provide design specifications to improve the robot mechanics, controllers, and estimation processes, we choose arbitrary bounds such that

$$\| \boldsymbol{\delta} \| \leqslant \delta_M, \quad \| \eta \| \leqslant \eta_M \tag{30}$$

Once again, we use the proposed uncertainty analysis to determine the maximum tolerance bounds δ_M and η_M, providing design specifications. As the velocity of the state resulting from our TVR planner changes sign after every step, typical convergence analysis regards this effect as an oscillatory behavior despite the fact that the absolute value of the CoM state, $\mathbf{x}$, effectively decreases over time. To remedy this, we perform a convergence analysis after two steps instead of a single step. Therefore, given an initial state, $\mathbf{x}$, after two steps, the new state, $\mathbf{x}''$, is obtained by applying Equation (29) twice,

$$\begin{aligned} \mathbf{x}'' &= \boldsymbol{A}^2\mathbf{x} + \boldsymbol{A}\boldsymbol{B}(p + \eta) + \boldsymbol{B}(p' + \eta') \\ p &= \boldsymbol{K}(\mathbf{x} + \boldsymbol{\delta}) \\ p' &= \boldsymbol{K}(\mathbf{x}' + \boldsymbol{\delta}') \end{aligned} \tag{31}$$

Table 1. Planner parameter. Each parameter has x and y components. We use the same value for both directions.

$[t'_x t'_y]$	$[\kappa_x \kappa_y]$
[0.2, 0.2]	[0.16, 0.16]

where $()$, $()'$, and $()''$ represent the kth, $(k+1)$th, and $(k+2)$th step, respectively. The main idea is to find the region in $\mathbf{x}$ for which a Lyapunov function decreases value after two steps subject to the maximum errors, δ_M and η_M:

$$\Delta V = \mathbf{x}''\mathrm{T}\boldsymbol{P}\mathbf{x}'' - \mathbf{x}^\top \boldsymbol{P}\mathbf{x} \leqslant 0 \tag{32}$$

Substituting Equation (31), arranging the terms, and setting the upper bound ΔV, it can be shown that

$$\begin{aligned} \Delta V &= \mathbf{x}^\top (\boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{A}_{cc} - \boldsymbol{P})\mathbf{x} + 2\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{A}_{cc}\mathbf{x} + \boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta} \\ &\leqslant -a \| \mathbf{x} \|^2 + 2b \| \mathbf{x} \| + c \\ &\leqslant 0 \end{aligned} \tag{33}$$

where

$$\boldsymbol{A}_c = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K} \tag{34}$$

$$\boldsymbol{A}_{cc} = \boldsymbol{A}_c^2 \tag{35}$$

$$\boldsymbol{\zeta} = \boldsymbol{A}_c\boldsymbol{B}\boldsymbol{K}\boldsymbol{\delta} + \boldsymbol{B}\boldsymbol{K}\boldsymbol{\delta}' + \boldsymbol{A}_c\boldsymbol{B}\eta + \boldsymbol{B}\eta' \tag{36}$$

$$a = -\lambda_M\left(\boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{A}_{cc} - \boldsymbol{P}\right) \tag{37}$$

$$\begin{aligned} b = \delta_M &\left(\| \boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B}\boldsymbol{K} \| + \| \boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{B}\boldsymbol{K} \|\right) \\ &+ \eta_M\left(\| \boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B} \| + \| \boldsymbol{A}_{cc}^\top \boldsymbol{P}\boldsymbol{B} \|\right) \end{aligned} \tag{38}$$

$$c = g(\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta}) \tag{39}$$

Note that the upper bounds defined by a, b, and c have a quadratic form that allows us to easily find a solution of the Euclidean norm of the CoM state. Here $\| \cdot \|$ is the l^2-norm, $\lambda_M(\cdot)$ denotes the maximum eigenvalue of its matrix arguments, and $g(\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta})$ is the sum of the l^2-norm of every term in $\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta}$ similar to b. The definition of g is deferred to Appendix C owing to the length of the expression. Note that a is positive if the planner parameters are tuned such that the LIP behavior is stable. Solving for $-a \| \mathbf{x} \|^2 + 2b \| \mathbf{x} \| + c \leqslant 0$, we obtain the uncertainty ball region,

$$B_r = \left\{ \mathbf{x} \;\middle|\; \| \mathbf{x} \| \leqslant \frac{b + \sqrt{b^2 + ac}}{a} \right\} \tag{40}$$

The above ball defines the region of states for which we cannot warranty asymptotic stability. Conversely, the region of states outside of the ball, $\mathbf{x} \notin B_r$, corresponds to asymptotically stable states. Note that a smaller ball means a larger stability region, and if the errors η and δ are zero, the ball would have zero radius and any state would be asymptotically stable. However, because of mechanical deflection,

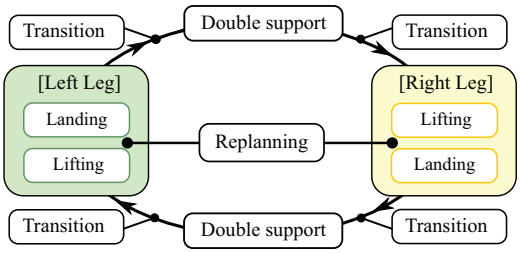


Fig. 6. State machine. Our biped walking motion is achieved via sequential contact phase changes governed by the temporal parameters shown in Table 2. The robot's swing leg phase can be terminated earlier than the predefined swing time if the contact is detected before the end of the swing phase. In the middle of the swing, the next foot placement is computed by the TVR planner.

limited control bandwidth, and estimation errors, b and c are non-zero.

By substituting the planner's parameters from Table 1 into the above equation, we can quantify and analyze the effect of the errors mentioned above. Figure 5(b) shows the CoM phase space plot. Take Equation (26) and write it in the simple form,

$$p = k_p x + k_d \dot{x} \tag{41}$$

As we said, this equation corresponds to the foot landing location control policy to stabilize a biped robot. We also mention that the maximum step size for our robot, Mercury, is $-0.5\ \text{m} < p < 0.5\ \text{m}$. If we apply these kinematic limits to the above foot control policy, we obtain a pair of lines in the phase plane which define the area of feasible CoM states given foot kinematic limits. This area is highlighted in light blue color in our phase plot. To be clear, the light blue colored area defines the state for which the robot can recover within a single walking step without violating kinematic limits.

Next let us consider the uncertainty region defined by Equation (40). Note that the terms b and c depend on the uncertainty errors. For example, if we have a maximum foot landing error of 0.045 m and a maximum state estimation error of 0.03 m, then $\eta_M = 0.045\text{m}$ and $\delta_M = 0.03\ \text{m}$. If we plug these values into Equations (38)–(40), we obtain the orange ball shown in Figure 5(b). The inside of this ball represents states for which we cannot warranty asymptotic stability. The problem is that the orange uncertainty region include states outside of the feasible CoM state, the light blue region. This means that the actual CoM could have a value for which the robot cannot recover because it requires foot steps outside of the robot's kinematic limits. As we mentioned previously, our biped robot, Mercury, underwent significant mechanical, control, and sensing improvements to remedy this problem. The errors represented by the orange ball are close to what we have observed in our walking tests before we upgraded Mercury. After making hardware and control improvement, we reduced the errors to $\eta_M = 0.01\ \text{m}$ and $\delta_M = 0.007\ \text{m}$.

Table 2. Temporal parameter of walking

Double stance	Transition	Swing
0.01 s	0.03 s	0.33 s

In particular, to reduce δ_M, we employed a tactical IMU (STIM-300) and MoCap data from a phase space motion capture system providing a body velocity estimation resolution of 0.005 m/s and a body position accuracy of 0.005 m. The blue ball in Figure 5(b) represents the new uncertainty region given by this significant improvements. We can now see that the blue ball is completely contained within the light blue region. This means that although we do not know where the CoM state is located inside the blue ball, we know that whatever the state is, it is within the feasible CoM state region, and, therefore, the foot control policy will find stabilizing foot locations.

6. Implementation details

6.1. Walking control

For our purposes, a biped's walking control process consists of three phases: swing (or single stance), double stance, and contact transition. In particular, the contact transition ensures smooth transition from single to double contact. Each phase starts and ends following predefined temporal parameters as shown in Table 2. The swing phase can, and it often does, terminate earlier than the specified swing time because the biped might make contact with the ground earlier than planned. We automatically terminate the swing phase upon detecting contact to prevent sudden jerks that can occur when pushing against the ground. The ground contact is detected by the limit switches attached to the spring-loaded passive ankles shown in Figure 2(e). The locomotion phases are illustrated in Figure 6. At the middle of the duration of each swing phase, our TVR planner computes the immediate foot step location to achieve stable locomotion based on the policy given by Equation (26). This decision process works as follows. After breaking contact with the ground, the swing foot first moves to a predefined default location with respect to the stance foot. Then, a new foot landing location is computed using the TVR planner. Based on this computation, the swing trajectory is re-adjusted to move to the computed foot landing location completing the second half of the swing motion until contact occurs.

Owing to the non-negligible body-to-leg-weight ratio, when the swing motion occurs, it disturbs the robot's body. As the inertial coupling between the leg and body has a strong negative effect on the robot's ability to walk and balance, it is important to reduce these types of disturbances. In particular, we mentioned earlier that the robot's swing leg first moves to a default location, and from there it computes a new foot landing location to dynamically balance

and walk. Therefore, we focus on reducing the jerky motion that occurs from re-adjusting the foot trajectory at the middle of the swing motion. In our experiments, we first move to the default swing location using a B-spline, and then compute a minimum jerk trajectory to achieve the final landing location. The inclusion of this minimum-jerk trajectory is important as it significantly reduces the said disturbances between the swinging leg and the robot's body posture.

When the swing motion ends, the state machine switches to the contact transition phase. Here the DynWBC control block shown in Section 4.1.2 plays a key role to smoothly transition the contact from single to double support without introducing additional jerky movement. On the other hand, when a contact occurs, triggering a switch from single to double support, the KinWBC control block can generate a discontinuity of the joint position command. To reduce this additional jerk caused by KinWBC, the joint position command of the swing leg at the end of the swing phase is linearly interpolated with the command from KinWBC for the transition phase. As the contact transition progresses, the ratio between the final joint position command and the transition phase decreases which completes the transition. By doing all of these improvements, we accomplish smooth motions with reduced jerk for effective walking.

6.2. Task and weight setup of WBLC

The WBLC task breakdown and their priorities for each phase are summarized in Table 3. A common task for every control phase is the body posture task which keeps the body's height, roll, and pitch constant. As Mercury has only six actuators, the robot cannot directly control its body yaw rotation and horizontal movement most of the time. Therefore, we only control three components (R_x, R_y, z) of the six-dimensional body motions (R_x, R_y, R_z, x, y, z). Here $R_{\{.\}}$ denotes rotations.

During the swing phase, we control the linear motion of the foot in addition to the robot's body posture. The swing foot task is hierarchically ordered under the body posture task to prevent the swing motion from influencing the body posture control. However, this priority setup is not enough to completely isolate the body posture control from the swing motion control because the null space of the body task does not remove the entire 6-DoF body motion. In our case, the body posture control task only controls three of the six dimensions of body motion, which means that the other three components still reflect on the swing foot task even after the foot task has been projected into the null-space of the body posture task. To further decrease the coupling between the body motion and the swing foot intended motion, we set to zero all of the terms corresponding to the floating base DoFs appearing in the foot task Jacobian. By doing so, the three actuators in the stance leg are dedicated only to body posture control while the other three actuators in the swing leg are dedicated to control the swing foot trajectory.

Table 3. Task breakdown and priorities of KinWBC.

Priority	Double stance	Transition	Swing
1	R_x, R_y, z	R_x, R_y, z	R_x, R_y, z
2	—	—	$\text{Foot}_{x,y,z}$

The values of the weights of the cost function in Equation (11) in DynWBC are specified in Table 4. These values are presented in vector form because all of the cost matrices are diagonal. The matrix $\boldsymbol{W}_{\ddot{\mathbf{q}}}$ is the weight matrix for relaxing desired joint accelerations to adjust for partially feasible acceleration commands. These weights are set to relatively large values to penalize deviations from the commanded joint accelerations. The same values are kept for every phase.

The weights $\boldsymbol{W}_r$ and $\boldsymbol{W}_c$ change as a function of the walking control phase because the reaction forces and feet movements are regulated by those weights. During the double support phase, the weights related to the contact point acceleration, $\boldsymbol{W}_c$, are assigned a large value, 10^3. Penalizing contact accelerations approximates contact conditions without imposing hard constraints. In addition, during double support, the weight matrix regulating reaction forces, $\boldsymbol{W}_r$, is assigned relatively small values to provide sufficiently large forces to support the robot's body. Note that $\boldsymbol{W}_r$ penalizes the tangential direction values more than the normal direction values, which helps to fulfill the friction limits associated with the contact reaction forces.

The weights $\boldsymbol{W}_r$ and $\boldsymbol{W}_c$ change value during the contact transition phase. The right arrows in Table 4 indicate that the weights transition smoothly from the left to the right values. For instance, $1 \rightarrow 5$ means that the value applied to the weight is set to 1 at the beginning of the

Table 4. Weight Setup. The values of the weight matrices are described in vector form because we consider only diagonal weight matrices. The components associated with reaction and contact weights are six dimensional, starting from the right foot's x, y, and z directions and then considering the left foot's Cartesian components; therefore, $\boldsymbol{W}_r$ and $\boldsymbol{W}_c$ have six components.

	Double support	Transition (right)	Swing (right)
$\boldsymbol{W}_{\ddot{\mathbf{q}}}$	$10^2 \times \mathbf{1}_{12\times 1}$	$10^2 \times \mathbf{1}_{12\times 1}$	$10^2 \times \mathbf{1}_{12\times 1}$
$\boldsymbol{W}_r$	$[1, 1, 0.01, 1, 1, 0.01]^\top$	$[1 \rightarrow 5, 1 \rightarrow 5, 0.01 \rightarrow 0.5, 1, 1, 0.01]^\top$	$[5, 5, 0.5, 1, 1, 0.01]^\top$
$\boldsymbol{W}_c$	$10^3 \times \mathbf{1}_{6\times 1}$	$[(10^3 \rightarrow 10^{-3}) \times \mathbf{1}_{1\times 3}, 10^3 \times \mathbf{1}_{1\times 3}]^\top$	$[10^{-3} \times \mathbf{1}_{1\times 3}, 10^3 \times \mathbf{1}_{1\times 3}]^\top$

transition phase, and we linearly increase it to 5 by the end of the phase. Let us take the example with the right foot during the transition phase. At the beginning of the transition phase the weight values coincide with the values in the previous phase, i.e., double support. At the end of the transition phase, when the right leg is about to leave the ground and start the swing phase, the first three terms of $\boldsymbol{W}_r$, coinciding with the Cartesian components of the right foot reaction force, are set to large values to penalize reaction forces. At the same time, the three first terms of $\boldsymbol{W}_c$ are set to tiny values to boost swing accelerations on the right foot.

During this transition we perform an additional step. For the constraint defined in Equation (13) of DynWBC, i.e., $\boldsymbol{S}\mathbf{F}_r \leqslant \mathbf{F}_{r,z}^{\max}$, we linearly decrease the value of the upper bound $\mathbf{F}_{r,z}^{\max}$ to drive the right foot normal force to zero before the swing motion initiates. This linear decrease starts with the value set during double support and ends with a value equal to zero.

6.3. Base state estimation

As the true CoM state is subject to errors from the model and disturbances from the swing leg motion, our current implementation instead uses the robot's base state, and assumes that the CoM of the robot is approximately at this location. The robot's base is a concrete point on the torso indicated by a black dot in Figure 1. The base point was chosen by empirically comparing the CoM position and base position to find the lowest discrepancies. Figure 7 shows velocity estimation values. As we can observe, the difference between the CoM velocity (black dotted line) and base velocity (blue solid line) is imperceivable. This enables us to (1) decouple the computation of the CoM state from the swing leg motion, and (2) perform a straightforward sensor-fusion process with a Kalman filter by combining the sensed body positions from joint-encoders and the overhead MoCap system.

As stated previously, Figure 7 compares the base velocity data obtained from different sensors. In Section 3, we stated that there are two ways to measure joint data: one is using the absolute encoders directly attached to the robot joints, and another is using the quadrature encoders attached to the back of the electric motors by multiplying their value with the actuator's transmission ratio. The green lines and the blue lines on the above figures correspond to the base velocities computed from data measured by absolute encoders and quadrature encoders. The blue lines are less noisy, but both green and blue data are not proper for our walking planner because the velocity profile shows a significant fluctuation, which makes the prediction of the state challenging. However, the velocity data obtained from the MoCap system, i.e., the red lines, shows a consistent trend with the walking phases such that we decided to rely on it. To deal with MoCap marker occlusions, we perform sensor fusion between the MoCap and encoder data via Kalman filtering and average filtering techniques. This data

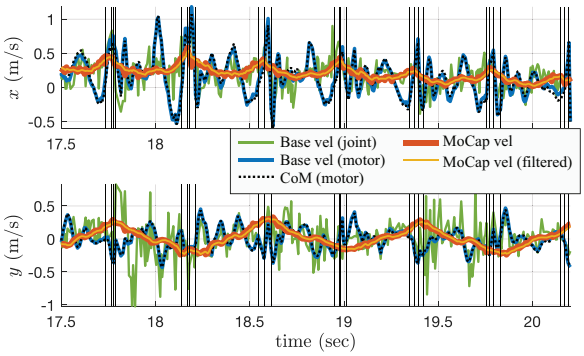


Fig. 7. Base and CoM velocity. The base and CoM velocity from different measurements are plotted. The base velocities are computed with joint velocity data measured by absolute encoders or quadrature encoders. The base velocity estimated by absolute encoders are too noisy and significantly fluctuates during the swing phases. Even with quadrature encoders, the fluctuation remains although the noisy level is lower than those estimated by using absolute encoders. During experiments, we use the results indicated by the yellow line, which corresponds to the filtered velocity data obtained from the MoCap system.
Note: Colour version of the figure is available online.

is shown as a yellow line on the previous figure showing that it is fairly similar to the red line.

For the estimation of the base positions in global frame we use the MoCap system. As for estimating base positions with respect to the stance foot we rely only on the robot's IMU and joint encoder data without using the MoCap system. This last process is more robust than attaching LED sensors to the feet because they incur frequent occlusions and break often due to the repetitive impacts.

6.4. Kinematic model verification with MoCap data

As we mentioned in the previous section, an accurate kinematic model is very important to compute stabilizing foot landing locations via the TVR planner. Moreover, for real-time WBLC, the model's accuracy significantly influences the landing location accuracy. To perfect our kinematic model which was initially built using the parameters obtained from CAD design, we utilize the MoCap system. By manually comparing MoCap and kinematic data, we tune effectively the model parameters such as orientation and position of the joint axes and linkages to reduce uncertainty. These mechanical structures can become slightly tilted and displaced after long experimental sessions. The parameters are adjusted by hand until the two sets of data are sufficiently close.

For this calibration process, we first fix Mercury's torso on top of a table as shown in Figure 8(a). For this fixed posture, we let the robot swing one of its legs and simultaneously gather MoCap and kinematic data. The positions of the LED sensors attached to the leg are post-processed to

be described in the robot's local frame, which is defined by three LED sensors attached to the robot's body (see Figure 1). The two different sets of data, one obtained from the MoCap system and the other one obtained from the current robot kinematic model are used to further tune the kinematic parameters. Figure 8(b) shows both the LED position data measured by the MoCap system and the same position data measured by the joint encoders using the tuned kinematic model. The result shows that the error of our final kinematic model has less than a 5 mm error.

7. Results

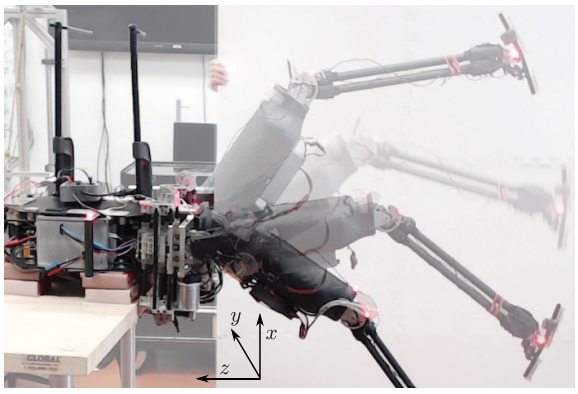
We conducted extensive walking and stepping experiments of various kinds using our passive-ankle biped robot, Mercury. For all of these experiments, Mercury was unsupported, that is, without overhead support. The experiments show stable behavior during directional walking, push recovery, and mildly irregular terrain walking. We also deploy the same control and dynamic walking schemes to our new lower-body humanoid robot, DRACO, and rapidly accomplished dynamic balancing. Finally, we conducted simulations using other humanoid robots to show the versatility of our whole-body controller and walking algorithm. A video showing experimental and simulation results is provided as Extension 1.

7.1. Directional walking

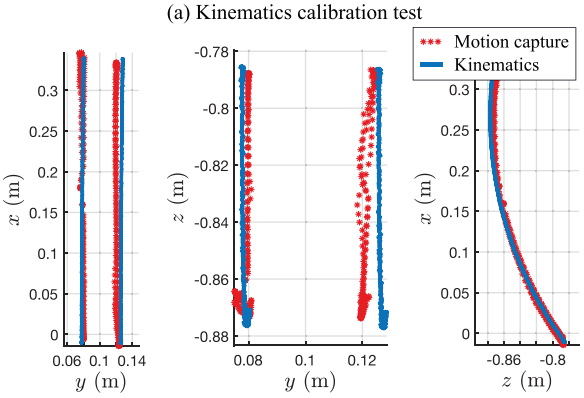
Directional walking means achieving dynamic walking toward a particular direction. To achieve this, we manipulate the origin of Mercury's reference frame. In turn, our TVR planner controls Mercury's foot stepping to converge to the reference frame, which for this test is a moving target. In other words, we steer the robot in the four cardinal directions in this manner, see Figure 9(a). Figure 9(b) shows the time trajectory of the desired robot's path and the actual robot's location. The actual location is obtained using the MoCap system based on the LED attached to the robot's base. These results show that Mercury follows the commanded path relatively well albeit slow convergence rates in the lateral direction possibly due to the limited hip's abduction/adduction range.

Figure 9(c) shows commanded and sensed joint torque data. The vertical black lines indicate the walking control phases. As we can see, the torque commands smoothly transition despite contact changes. The knee torque commands change between 0 and 40 Nm depending on the control phase of the leg, but there is no discontinuity causing jerky behavior of the desired torque commands despite the short (0.06 s) transition periods.

The right and left knee joint position data is shown in Figure 9(d). As mentioned in Section 4.2, the desired motor position commands are adjusted to account for spring deflections. The data shows that joint positions sensed with the absolute encoders are close to the position commands while the motor position data is off by the amount



(a) Kinematics calibration test



(b) Position of LED sensors attached at the left foot

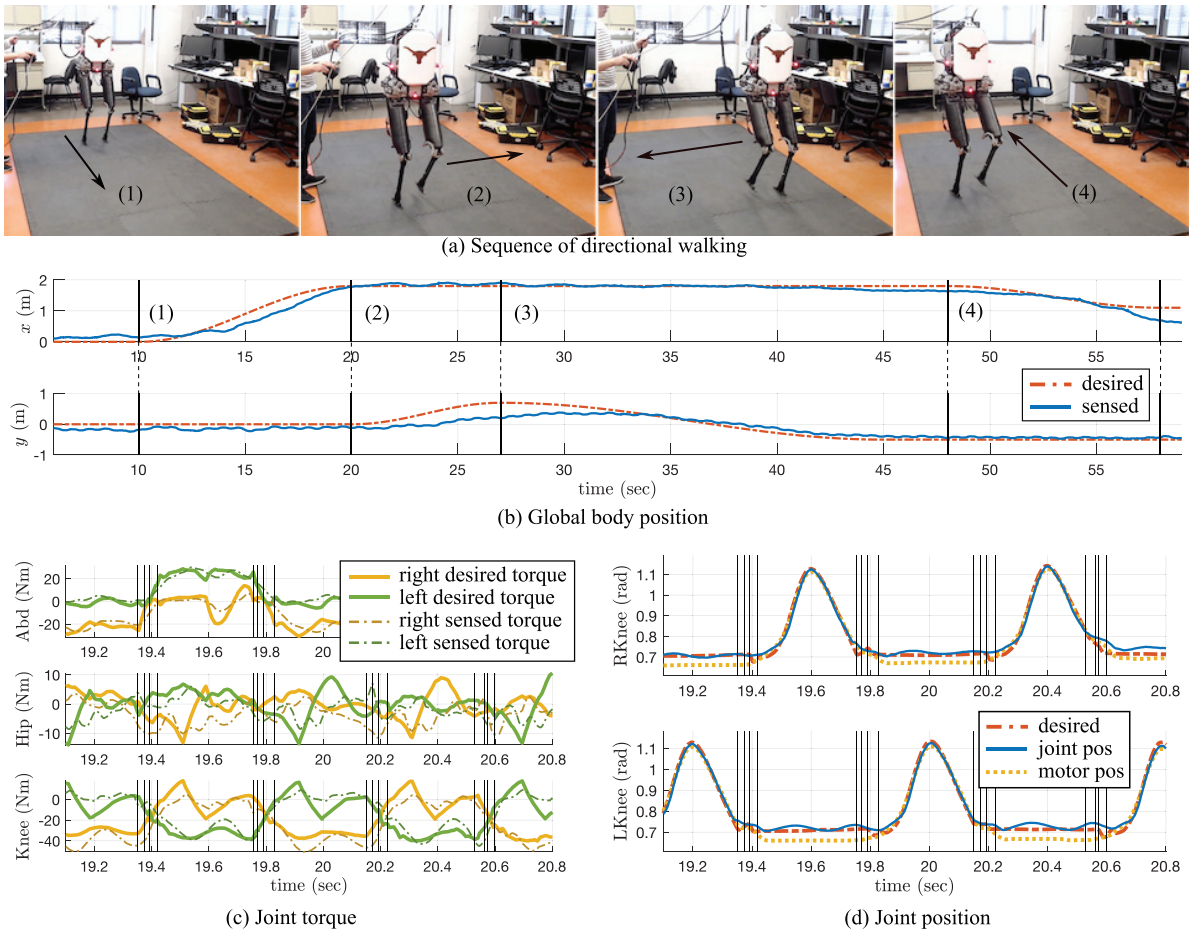
Fig. 8. Kinematic model calibration. (a) Mercury swings its leg while its torso is fixed on a table. (b) To tune the kinematic model parameters, we compare the LED position data obtained from the MoCap system and the position data computed by the kinematic model.

corresponding to spring deflections. The spring deflection compensation is notable when the knee joint supports the body weight, i.e., the periods between 19.8–20.2 s for the right knee and 19.4–19.8 s for the left knee.

7.2. Robustness of balance controller

To demonstrate the robustness of the proposed walking control scheme, we conducted multiple instances of an experiment involving external disturbances. The first test, shown in Figure 10, analyzes Mercury recovering its balance after a junior football ball of weight 0.32 kg and horizontal speed of about 9 m/s impacts its body. A second test, shown in Figure 11, shows a person continuously pushing Mercury's body with gentle forces to see how the robot reacts. Finally, the last experiment, shown in Figure 12, shows Mercury walking in a mildly irregular terrain without knowledge or sensing of the terrains topology. In the three experiments, Mercury successfully recovers from the disturbances.

For the ball impact experiment shown in Figure 10, we show the phase plots of the lateral CoM direction. As the ball hits the robot laterally, the analysis is done on the y direction. Lateral impact recovery is difficult because the



(a) Sequence of directional walking

(b) Global body position

(c) Joint torque

(d) Joint position

Fig. 9. Directional walking of Mercury. (a) Mercury starts walking from the back and goes forward, left, right, and back again. (b) The robot's location from the base LED sensor is compared with the commanded walking trajectory. Mercury follows fairly well the desired trajectory but shows limited convergence rate with respect to moving towards the lateral direction. (c) Joint torques change smoothly despite quick contact transitions thanks to our WBLC method. (d) Knee joint position data shows that spring deflection models are effective for reducing joint position tracking error.

hip abduction/adduction joints have a very limited range of motion, $\pm 15^{\circ}$. Owing to the very small width of the feet, the landing location has to be very accurate as discussed previously. Each phase plot in this figure, shows two sequential steps, depicted in blue and red color lines. For instance, for the 28th step, we differentiate the solid blue line, which represents the sensed base trajectory for the actual 28th step, from the solid red line, which represents the trajectory for the next step, the 29th. Dotted blue and red lines represent the predicted trajectory given the TVR control policy and pendulum dynamics hypothesis. The particular operating details of the TVR controller during this impact experiment are described in the caption of Figure 10. In essence, the ball hits the robot at the 28th step, and at the 30th step, Mercury fully recovers its balance, going back to the normal regime at the 31st step.

Also from Figure 10, we analyze the foot landing accuracy. In the phase plots, the red star, the red circle, and the blue cross represent the stance foot, commanded foot landing location, and actual foot landing location, respectively. Except during the recovery steps, 29th and 30th steps, the foot landing location errors are less than 0.5 cm, as seen in the 28th and 31st steps. This is significantly less error than the maximum tolerable one as shown in the uncertainty analysis of Figure 5. In analyzing extended experimental data, the foot landing error is consistently less than 0.5 cm.

Our control and walking methods are robust to mild terrain variations as shown in Figure 12. For this particular experiment, we set κ_x, shown in Table 1, to a value of 0.25 to enable the robot to keep moving forward despite the terrain variations. In addition, the robot's feet sometimes get stuck on the edge of the mats, which adds difficulty to the locomotion process. However, the robot successfully traverses the terrain.

7.3. Experimental evaluation on new biped robot DRACO

DRACO is our newest humanoid lower body, having 10 viscoelastic liquid-cooled actuators (Kim et al., 2018) on

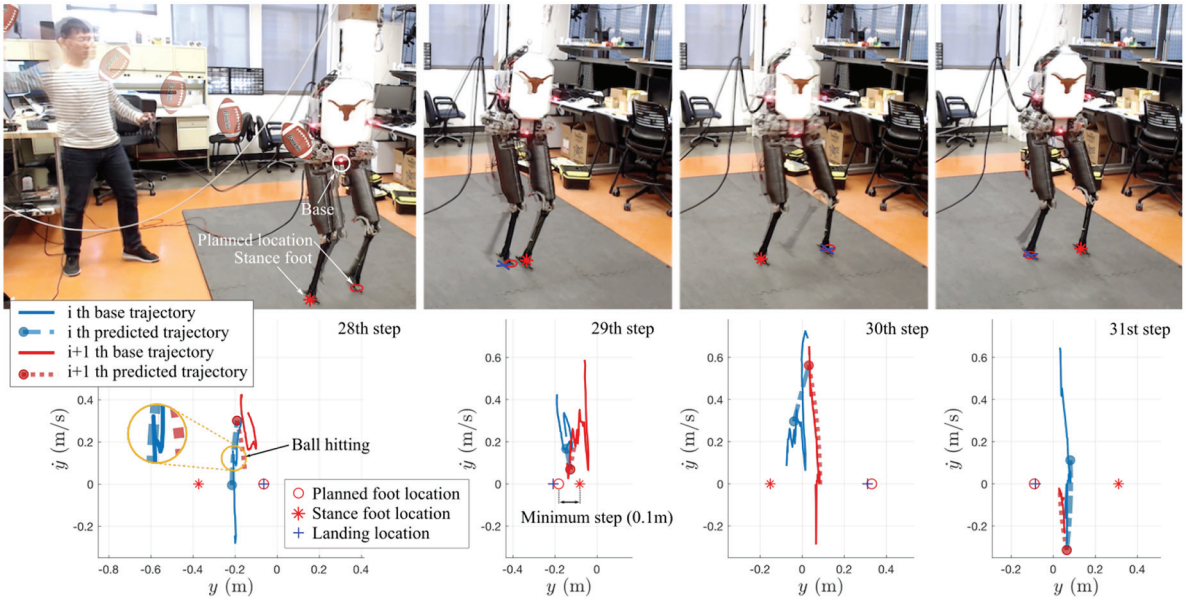


Fig. 10. Mercury recovers its balance after being disturbed by a lateral impact applied by throwing a junior American rubber football ball, weighting 0.32 kg. In the 28th step, the ball hits Mercury on its side as depicted in the lateral change of the CoM state, i.e., y direction. For this instance, when the lateral impact happens, the next foot landing location, in our case the left leg, has already been planned and there is nothing else that can be done. Thus, Mercury finishes the lateral step without responding to the disturbance. For the following step, step 29th, the CoM velocity is positive in the y direction due to the lateral disturbance. This value on the CoM state causes our TVR walking planner to trigger a recovery step using the right foot which is commanded to move inward towards the stance foot. However, the amount it has to move would cause a collision with the stance leg, therefore our planner chooses to land the right foot at the minimum lateral range of 10 cm from the stance leg. This choice, causes the robot to only partially recover from the disturbance but failing to reverse velocity. As a result, for the next step, step 30th, Mercury's TVR walking controller decides to take a large step, 48 cm from the stance leg, which enables it to reverse velocity in the direction opposite to the impact. Finally, Mercury goes back to its nominal balancing motion in step 31.

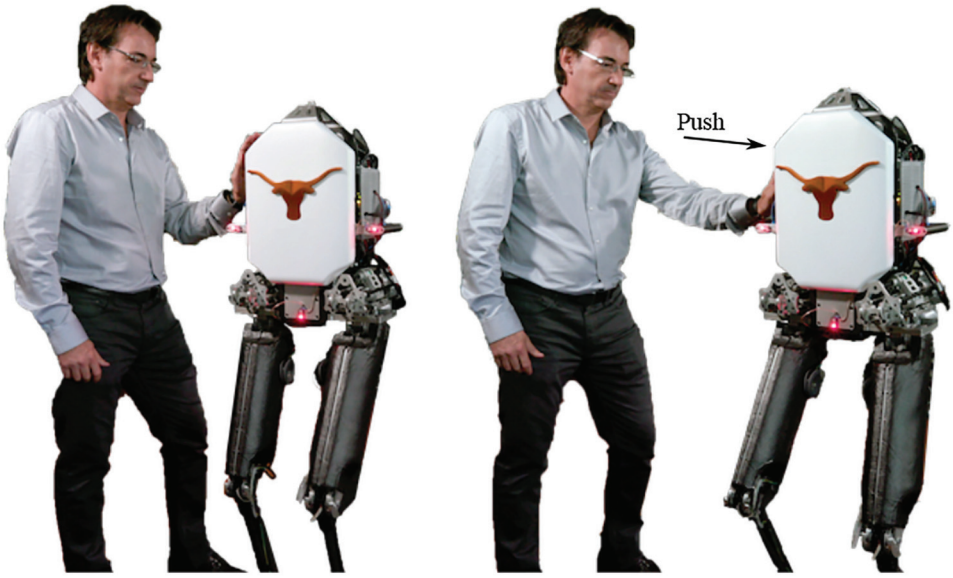


Fig. 11. Interaction with a human subject. Mercury maintains its balance despite the continuous pushing forces.

its hips and legs. Each limb has five actuators: hip yaw, roll, pitch, knee pitch, and ankle pitch. The IMU is the same as in Mercury, a STIM 300, and the MoCap LED sensor system is configured similarly to Mercury. The robot has many interesting features such as liquid cooling, tiny feet, quasi-passive ankles, and elastomers in the actuator drivetrains. We do not describe the hardware details of DRACO as they are being prepared for submission for an upcoming paper.

To equate DRACO to Mercury in some respects, we apply a soft joint stiffness policy to the ankle pitch

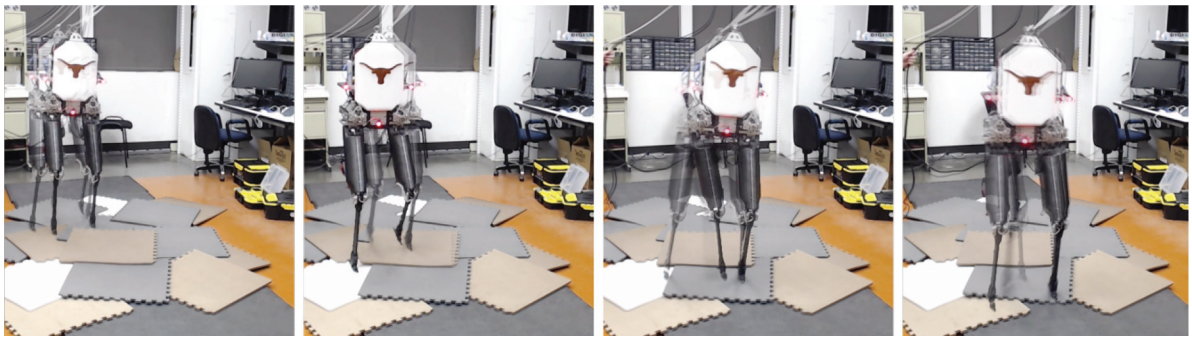
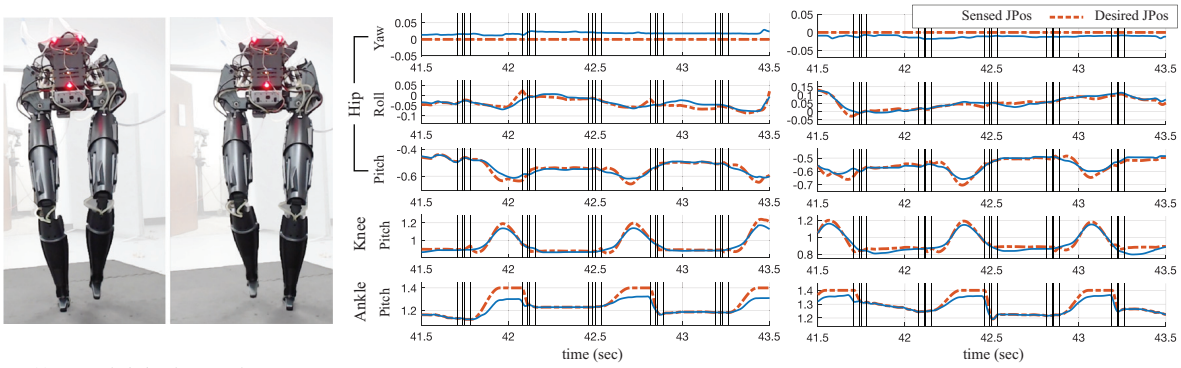


Fig. 12. Forward walking over an irregular terrain. Mercury walks forward over an irregular terrain constructed with foam mats arranged on top of each other. The robot's feet sometimes slip over the mat segments because the latter do not stick tightly to each other. Therefore, there are multiple disturbances. Our control and walking algorithms accomplish the necessary robustness to traverse these type of terrain including height variations of 2.5 cm), foot slippage, and foot trippings.



(a) DracoBip balancing experiment (b) Left leg joint position (c) Right leg joint position

Fig. 13. Balancing experiment of DRACO. Using the same WBLC and TVR algorithms from Mercury, DRACO was able to balance unsupported within a few days.

emulating a passive joint. For this first experiment, we set the hip yaw joint to a fix position with a joint control task implemented in WBLC. From a controller's standpoint, Mercury and DRACO are very similar for this experiment. DRACO is forced to perform dynamic locomotion without controlling ankles in similar ways than Mercury. For now, we check for foot contacts on DRACO based on ankle joint velocity measurements. As shown in Figure 13, DRACO balances successfully unsupported, just like Mercury did.

For WBLC on DRACO, we generated the robot's model using the CAD files and slightly adjusted mass values from gravity compensation tests. Except for feedback gains of the joint position controllers, we use similar planner parameters to Mercury. Here t' is set to $[0.21, 0.2]$ and κ is set to $[0.08, 0.13]$ for the experiment. Testing on DRACO was successfully accomplished, thus demonstrating that our WBLC-TVR framework is easily transferable to multiple robots, showing the generality of our methods.

7.4. Simulation results in assorted platforms

To show further applicability of the proposed control methods, we implement and test our WBLC and TVR algorithms on assorted robotic platforms such as Mercury, DRACO, Atlas, and Valkyrie. We implemented two types of simulation scenarios: dynamic walking and locomanipulation. Mercury, DRACO, and Atlas are utilized to implement dynamic walking motions. As mentioned in Section 6.2, for locomotion we define a foot task and a body posture task, $\boldsymbol{X}_{Mercury} = \{\ddot{\mathbf{x}}_{\text{foot}}, \ddot{\mathbf{x}}_{\text{body}}\}$, where $\ddot{\mathbf{x}}_{\text{foot}}$ and $\ddot{\mathbf{x}}_{\text{body}}$ are specifications for the foot and body tasks. The height, roll, and pitch of the body are controlled as constant values, respectively. As DRACO includes hip joints on both left and right side legs, we additionally formulate a hip configuration task for both hip joints of DRACO in addition to Mercury's tasks, $\boldsymbol{X}_{DRACO} = \boldsymbol{X}_{Mercury} \cup \{\ddot{\mathbf{x}}_{hip}\}$. The body task of DRACO controls its body height, roll, pitch, and yaw orientation. As shown in Figure 14(a) and (b), the simulation results of Mercury and DRACO demonstrate that both robot simulations are able to perform dynamic walking without much algorithmic modifications. The parameters of the planner are set to $t' = [0.2, 0.2]$ and $\kappa = [0.16, 0.16]$

Unlike the above two robots, Atlas and Valkyrie are full-body humanoid robots and their ankle joints are actuated so that we modify the task sets and constraints to test

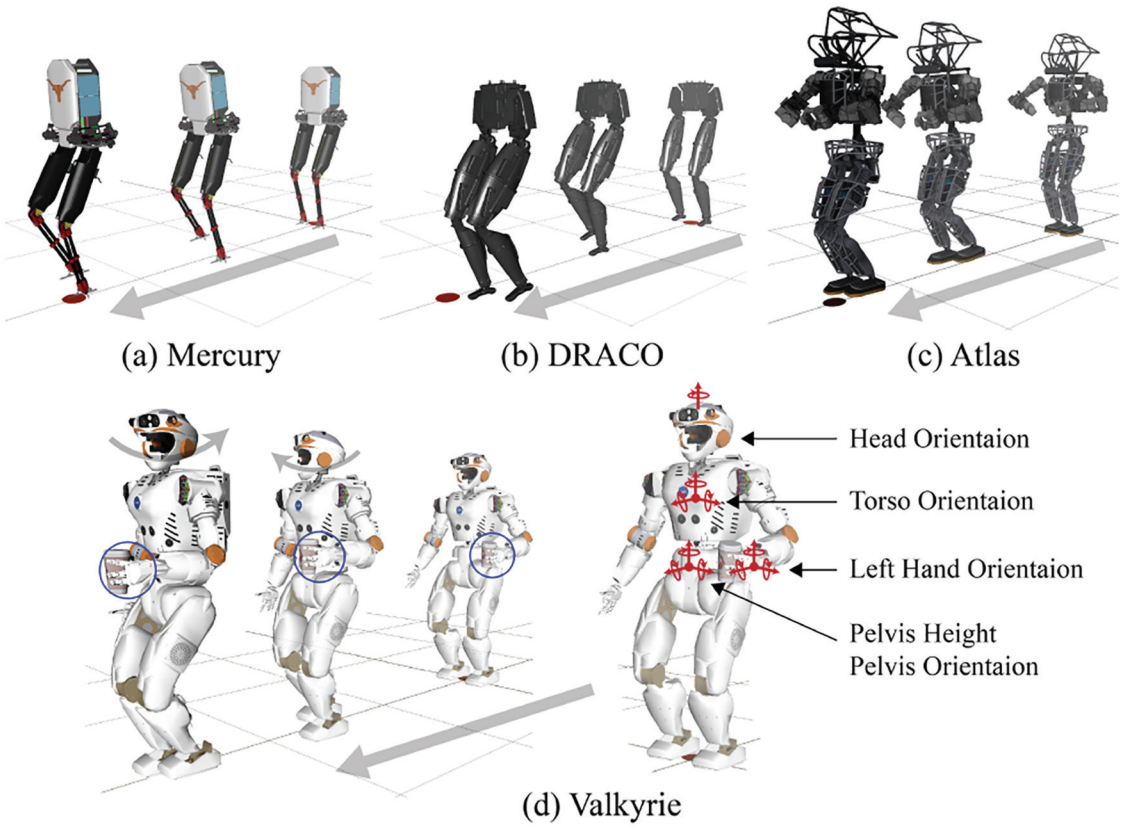


Fig. 14. Simulations using four robotic platforms: Mercury, DRACO, Atlas, and Valkyrie. The foot task is present in all of the simulations. DRACO contains an additional hip position task. Atlas and Valkyrie have additional tasks such as pelvis and torso orientation, foot orientation, and arm joint configuration. Lastly, Valkyrie has an additional head orientation and left hand orientation tasks. (a), (b), and (c) Simulation results for dynamic walking using Mercury, DRACO, and Atlas, respectively. (d) Locomanipulation capabilities of our WBLC performed by Valkyrie.

our algorithm using simulations. We modify the inequality constraint for the contact wrench cone to surface contacts in (12). For these full-body humanoid robots, the height of the pelvis, which corresponds to the floating base, is constantly controlled in the same way as Mercury and DRACO. We define the orientation tasks for the pelvis and torso. In addition, a task for controlling foot orientation is introduced for stable contact on the feet. Based on the defined tasks, the task set of Atlas is designed to be $\boldsymbol{X}_{\text{Atlas}} = \{\ddot{\mathbf{x}}_{\text{foot}}, \ddot{\mathbf{x}}_{\text{pelvis}}, \ddot{\mathbf{x}}_{\text{torso}}, \ddot{\mathbf{x}}_{\text{jpos}}\}$ where $\ddot{\mathbf{x}}_{\text{jpos}}$ represents a task for controlling the entire joint positions of the robot. As shown in the simulation, Atlas is able to perform dynamic walking similarly to Mercury and DRACO without modifying our algorithms as shown in Figure 14(c).

We define additional tasks for controlling the left hand and the head orientations to demonstrate locomanipuation capabilities on Valkyrie, $\boldsymbol{X}_{\text{Valkyrie}} = \boldsymbol{X}_{\text{Atlas}} \cup \{\ddot{\mathbf{x}}_{\text{hand}}, \ddot{\mathbf{x}}_{\text{head}}\}$. The simulation result shows that our algorithm can accomplish the desired locomanipulation behavior as shown in Figure 14(d). These four simulations show that our algorithm is applicable to various biped humanoids.

8. Conclusions

We have demonstrated robust dynamic walking of various biped robots, including one with no ankle actuation, using a novel locomotion-control scheme consisting of two components dubbed WBLC controller and TVR planner. The algorithmic generality has been verified on hardware with the bipeds Mercury and DRACO and in simulation with other humanoids such as Valkryie and Atlas. We have performed an uncertainty analysis of the TVR planner and found maximum allowable errors for our state estimator and controllers, which enabled us to significantly redesign and rebuild the Mercury robot and tune the controllers and estimators. By integrating a high-performance whole-body feedback controller, WBLC, a robust locomotion planner, TVR, and a reliable state estimator, our passive-ankle biped robot and lower-body humanoid robot accomplish unsupported dynamic locomotion robust to impact disturbances and rough terrains.

In devising our control scheme, we have experimented with a variety of whole-body control formulations and feedback controllers. We compared different WBC operational task specifications such as foot position versus leg joint

position control, base versus CoM position control, having versus not having task priorities, etc. In the low-level controller we also experimented with torque feedback with disturbance observers, joint versus motor position feedback, and joint position control with and without feedforward torques. The methodology presented here is our best performing controller after system-level integration and exhaustive testing.

With our new biped, DRACO, we have explored initial dynamic locomotion. In the future, we will explore more versatile locomotion behaviors such as turning and walking in a cluttered environment. In the case of Mercury, we could not change the robot's heading because of the lack of yaw directional actuation. With simple additions to the current TVR planner, we will be able to test turning of DRACO because the robot has hip yaw actuation. In addition, we will conduct robustness tests in a more complex way by exploring a cluttered environment involving contacts with many objects including human crowds.

Acknowledgements

The authors would like to thank the members of the Human Centered Robotics Laboratory at The University of Texas at Austin and the members of Apptronik for their support.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the Office of Naval Research (ONR grant number N000141512507), NSF (grant number 1724360), and a NASA Space Technology Research Fellowship (NSTRF) (grant number NNX15AQ42H).

ORCID iDs

Donghyun Kim https://orcid.org/0000-0001-9534-5383
Steven Jens Jorgensen https://orcid.org/0000-0003-1305-8714
jianwen Luo https://orcid.org/0000-0002-1594-3867
Luis Sentis https://orcid.org/0000-0003-2856-4863

Note

1. See https://github.com/openhumanoids (Valkyrie), https://github.com/dartsim/ (ATLAS), and https://github.com/sir-avinash/atrias-matlab (ATRIAS).

Supplemental material

Supplemental material for this article is available online.

References

Ahn J, Kim D, Bang S, Paine N and Sentis L (2019) Control of a high performance bipedal robot using viscoelastic liquid cooled actuators. *arXiv preprint arXiv:1906.03811*

Bahnasawi AA, Al-Fuhaid AS and Mahmoud MS (1989) Linear feedback approach to the stabilisation of uncertain discrete systems. *IEE Proceedings D Control Theory and Applications* 136(1): 47.

Boston Dynamics (2018) Getting some air, Atlas? Available at: https://youtu.be/vjSohj-Iclc (accessed: 25 December 2018).

Bouyarmane K, Caron S, Escande A and Kheddar A (2018) *Multi-contact Motion Planning and Control*. Dordrecht: Springer Netherlands, pp. 1763–1804.

Calanca A, Muradore R and Fiorini P (2016) A review of algorithms for compliant control of stiff and fixed-compliance robots. *IEEE/ASME Transactions on Mechatronics* 21(2): 613–624.

Collins SH, Ruina A, Tedrake R and Wisse M (2005) Efficient bipedal robots based on passive-dynamic walkers. *Science* 307(5712): 1082–1085.

Escande A, Mansard N and Wieber PB (2014) Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research* 33(7): 1006–1028.

Feng S, Whitman E, Xinjilefu X and Atkeson CG (2015) Optimization-based full body control for the DARPA Robotics Challenge. *Journal of Field Robotics* 32(2): 293–312.

Hartley R, Da X and Grizzle JW (2017) Stabilization of 3D underactuated biped robots: Using posture adjustment and gait libraries to reject velocity disturbances. In: *2017 IEEE Conference on Control Technology and Applications (CCTA*. IEEE, pp. 1262–1269.

Hereid A, Cousineau EA, Hubicki CM and Ames AD (2016) 3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1447–1454.

Herzog A, Rotella N, Mason S, Grimminger F, Schaal S and Righetti L (2016) Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid. *Autonomous Robots* 40(3): 473–491.

Honda (2011) Honda's all-new Asimo running, jumping. Available at: https://youtu.be/Bmglbk_Op64 (accessed: 25 December 2018).

Hubicki C, Abate A, Clary P, et al. (2018) Walking and running with passive compliance: Lessons from engineering a live demonstration of the ATRIAS biped. *IEEE Robotics Automation Magazine* 25(3): 23–39.

Johnson M, Shrewsbury B, Bertrand S, et al. (2015) Team IHMC's lessons learned from the DARPA Robotics Challenge Trials. *Journal of Field Robotics* 32(2): 192–208.

Kim D, Ahn J, Campbell O, Paine N and Sentis L (2018) Investigations of a robotic testbed with viscoelastic liquid cooled actuators. *IEEE/ASME Transactions on Mechatronics* 23(6): 2704–2714.

Kim D, Thomas G and Sentis L (2014) Continuous cyclic stepping on 3D point-foot biped robots via constant time to velocity reversal. In: *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*. IEEE, pp. 1637–1643.

Kim D, Zhao Y, Thomas G, Fernandez BR and Sentis L (2016) Stabilizing series-elastic point-foot bipeds using whole-body operational space control. *IEEE Transactions on Robotics* 32(6): 1362–1379.

Kohlbrecher S, Romay A, Stumpf A, et al. (2014) Human–robot teaming for rescue missions: Team ViGIR's approach to the 2013 DARPA Robotics Challenge Trials. *Journal of Field Robotics* 32(3): 352–377.

Koolen T, Bertrand S, Thomas G, et al. (2016) Design of a momentum-based control framework and application to the humanoid robot Atlas. *International Journal of Humanoid Robotics* 13(01): 1650007–35.

Koolen T, De Boer T, Rebula JR, Goswami A and Pratt JE (2012) Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models. *The International Journal of Robotics Research* 31(9): 1094–1113.

Kuindersma S, Deits R, Fallon M and Valenzuela A (2015) Optimization-based locomotion planning, estimation, and control design for the Atlas humanoid robot. *Autonomous Robots* 40(3): 429–455.

McGeer T (1990) Passive dynamic walking. *The International Journal of Robotics Research* 9(2): 62–82.

Pratt GA, Willisson P, Bolton C and Hofman A (2004) Late motor processing in low-impedance robots: Impedance control of series-elastic actuators. In: *Proceedings of the 2004 American Control Conference*, Vol. 4. IEEE, pp. 3245–3251.

Radford NA, Strawser P, Hambuchen K, et al. (2015 a) Valkyrie: NASA's first bipedal humanoid robot. *Journal of Field Robotics* 32(3): 397–419.

Raibert MH, Brown HB and Chepponis M (1984) Experiments in balance with a 3D one-legged hopping machine. *The International Journal of Robotics Research* 3(2): 75–92.

Raibert MH, Brown HB Jr, Chepponis M, et al. (1989) Dynamically stable legged locomotion. Technical Report AITR-1179, MIT, Cambridge, MA. Available at: https://dspace.mit.edu/handle/1721.1/6820 (accessed: 28 March 2020).

Rezazadeh S, Hubicki C, Jones M, et al. (2015) Spring-mass walking with ATRIAS in 3D: Robust gait control spanning zero to 4.3 kph on a heavily underactuated bipedal robot. In: *ASME 2015 Dynamic Systems and Control Conference*. Columbus, OH: ASME, V001T04A003.

Saab L, Ramos OE, Keith F, Mansard N, Soueres P and Fourquet JY (2013) Dynamic whole-body motion generation under rigid contacts and other unilateral constraints. *IEEE Transactions on Robotics* 29(2): 346–362.

Salini J, Padois V and Bidaud P (2011) Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions. In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1283–1290.

Sreenath K, Park HW and Grizzle JW (2012) Design and experimental implementation of a compliant hybrid zero dynamics controller with active force control for running on MABEL. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 51–56.

Wensing PM and Orin D (2013) Generation of dynamic humanoid behaviors through task-space control with conic optimization. In: *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 3103–3109.

Wensing PM, Wang A, Seok S, Otten D, Lang J and Kim S (2017) Proprioceptive actuator design in the MIT Cheetah: Impact mitigation and high-bandwidth physical interaction for dynamic legged robots. *IEEE Transactions on Robotics* 33(3): 509–522.

Westervelt ER, Grizzle JW, Chevallereau C, Choi JH and Morris B (2007) *Feedback Control of Dynamic Bipedal Robot Locomotion*. Boca Raton, FL: CRC Press.

Appendix A. Index to multimedia extensions

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Table of Multimedia Extensions

Extension	Media type	Description
1	Video	Experimental and simulation results from Section 7

Appendix B. Difference among footstep planners

The footstep planners proposed by Raibert et al. (1984), Koolen et al. (2012), Rezazadeh et al. (2015), and ours share the idea that footsteps are chosen based on a weighted sum of the CoM state, which is formulated by the general equation,

$$p = k_p x + k_d \dot{x} \tag{42}$$

where p, x, $\dot{x}$, k_p, and k_d are foot landing positions, sensed CoM position and velocity, and weights (or gains) for position and velocity feedback, respectively. This equation can be slightly varied by including desired position or velocity terms, but the basic idea does not change because of these additions.

Raibert et al. (1984) introduced long ago an unsupported hopping robot and its control method. For hopping control, the foot placement is decided by the equation,

$$p = \frac{T_{\text{st}}}{2}\dot{x} + K(\dot{x} - \dot{x}_d) \tag{43}$$

where T_{st} is the duration of the stance phase. Since the desired velocity, $\dot{x}_d$, is defined by the equation, $\dot{x}_d = -K_p x - K_v \dot{x}$ in the paper, the resulting equation for foot placement becomes

$$p = KK_p x + \left(K(K_v + 1) + \frac{T_{\text{st}}}{2}\right)\dot{x} \tag{44}$$

which has a similar form to Equation (42) with $k_p = KK_p$ and $k_v = K(K_v + 1) + T_{\text{st}}/2$.

Koolen et al. (2012) proposed a method called capture point (CP), which computes the foot's center of pressure (CoP) to drive the CoM velocity to zero at the CoP location given the current CoM state. Using the LIP model, CP is defined by the equation,

$$cp = x + \sqrt{\frac{h}{g}}\dot{x} \tag{45}$$

where g and h are the gravitational acceleration and the CoM height, respectively. This can be expressed using the form of Equation (42) by plugging 1 into k_p and $\sqrt{h/g}$ into k_d. As we mentioned previously, given CoM state, a robot will stop and stay on top of the capture point if the robot maintains its CoP on the CP. Differently, our TVR planner finds a foot placement location such that it reverses the CoM velocity before the CoM reaches that location.

In Rezazadeh et al. (2015), the step location for in-place walking is provided by the equation,

$$p = K_P\dot{x} + K_D(\dot{x} - \dot{x}_{n-1}) + K_I x \tag{46}$$

which has a slightly different form than the previous controllers because of the term $\dot{x}_{n-1}$ representing the CoM velocity at the previous step. However, for in-place walking, the effect of the velocity error between the current and previous step is not very significant. Therefore, we can regard the previous equation as one variation of Equation (42). In the same vein, our TVR planner, as presented in Equation (26), is also a variation of Equation (42), with weights, $k_p = (1 + \kappa)$ and $k_d = w^{-1}\coth(\omega t')$.

In conclusion, various locomotion planners can be cast using variations of Equation (42). The resulting behaviors vary depending on the chosen weights, e.g., CP makes a robot stop while ours makes the robot reverse its direction of motion. The benefit of our TVR planner over the others is that we provide an intuitive method and analysis to help with feedback gain selection. Our planner parameter t' must be close to half of the designated swing time and additional tuning for asymptotic stability is possible by checking the eigenvalues of the matrix, $\boldsymbol{A} + \boldsymbol{BK}$ in Equation (27).

Appendix C. Definition of $g(\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta})$

We have

$$g(\boldsymbol{\zeta}^\top \boldsymbol{P}\boldsymbol{\zeta}) = \delta_M^2 D + 2\delta_M \eta_M E + \eta_M^2 F \tag{47}$$

where

$$D = \| (\boldsymbol{A}_c\boldsymbol{BK})^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{BK} \| + 2 \| (\boldsymbol{A}_c\boldsymbol{BK})^\top \boldsymbol{PBK} \| + \| (\boldsymbol{BK})^\top \boldsymbol{PBK} \| \tag{48}$$

$$E = \| (\boldsymbol{A}_c\boldsymbol{BK})^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B} \| + \| (\boldsymbol{A}_c\boldsymbol{BK})^\top \boldsymbol{PB} \| + \| (\boldsymbol{A}_c\boldsymbol{B})^\top \boldsymbol{PBK} \| + \| \boldsymbol{B}^\top \boldsymbol{PBK} \| \tag{49}$$

$$F = \| (\boldsymbol{A}_c\boldsymbol{B})^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B} \| + 2 \| \boldsymbol{B}^\top \boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B} \| + \| \boldsymbol{B}^\top \boldsymbol{PB} \| \tag{50}$$