2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)
Beijing, China, November 6-9, 2018

Control Scheme and Uncertainty Considerations for Dynamic Balancing of Passive-Ankled Bipeds and Full Humanoids

D. Kim[1], S.J. Jorgensen[2], H. Hwang[3], and L. Sentis[4]

Abstract— **We propose a methodology for dynamically balancing passive-ankled bipeds and full humanoids. As dynamic locomotion without ankle-actuation is more difficult than with actuated feet, our control scheme adopts an efficient whole-body controller that combines inverse kinematics, contact-consistent feed-forward torques, and low-level motor position controllers. To understand real-world sensing and controller requirements, we perform an uncertainty analysis on the linear-inverted-pendulum (LIP)-based footstep planner. This enables us to identify necessary hardware and control refinements to demonstrate that our controller can achieve long-term unsupported dynamic balancing on our series-elastic biped, Mercury. Through simulations, we also demonstrate that our control scheme for dynamic balancing with passive-ankles is applicable to full humanoid robots.**

I. Introduction

Dynamic biped locomotion without ankle-actuation is difficult as feedback controllers must be stable under impact due to sudden contacts. Furthermore, accurate control of dynamic locomotion necessitates high-bandwidth control and state estimators. Therefore, existing dynamic locomotion techniques have been locomotion-specific formulations. For instance, [1] presented a method that relies and exploits the robot's natural dynamics, which was mechanically designed for biped locomotion with lightweight legs and a hip located center-of-mass. However, this procedure would not be trivial to apply to a different robot morphology, such as a full humanoid. As another approach, Hybrid zero dynamics [2] has stability properties for periodic motions. But, this approach is limited since additional tasks such as manipulation would be difficult to implement.

On a related note, the generality of whole-body control (WBC) approaches and their ability to specify multiple operational space tasks have made them a popular framework for controlling humanoid robots [3]–[7] with promising disaster response-related applications [8]–[11]. However, while there exist WBC-based dynamic locomotion techniques for bipeds with actuated ankles, such as the use of capture-point in various humanoids [4], [12], [13], there is no established WBC-based dynamic locomotion technique for passive-ankled bipeds due to additional practical implementation challenges. For instance, typical utilization of WBC requires a low-level torque feedback controller. However, having torque feedback poses some problems: (a) there exists stability and bandwidth trade-offs for cascaded WBC structures with high-level position control and low-level torque control [14], (b) torque feedback control performance is susceptible to time delays [15], and (c) passivity such as damping is reduced which decreases stability as torque control fundamentally removes Coloumb friction.

To address the desire of dynamic locomotion of passive-ankled bipeds with a generic controller, this paper presents a WBC-based control methodology utilizing inverse kinematics to resolve desired configurations from operational space tasks, a WBC-based quadratic program (QP) to extract contact-consistent torques as feed-forward, and a low-level joint position controller based on motor-position feedback signals with feed-forward currents for configuration control. Unlike typical implementations of WBC, this formulation achieves high-bandwidth position control and bypasses the problems incurred with a torque feedback loop by instead feed-forwarding currents from torques computed by our QP to the low-level joint-position controller. For a hardware demonstration of the algorithm, we identified additional sensing and control requirements by analyzing the uncertainty of our LIP-based velocity-reversal planner [14]. This determines the required accuracy of the center-of-mass (CoM) state estimation and the foot landing location subject to the robot's kinematic limits. Based on this analysis, we implement necessary sensing and control changes. Algorithmically, we verify that our approach is applicable to other humanoid robots in simulation where we have also imposed that the humanoids' ankles only have passive damping.

With our WBC-based methodology and uncertainty analysis, we achieve long-term unsupported dynamic balancing of our series-elastic and passive-ankled biped robot, Mercury. We emphasize that existing WBC-based biped dynamic locomotion rely on actuated feet and are not as dynamically challenging as ours. The contributions of our paper are as follows: (1) we devised and implemented a control scheme for dynamic balancing of passive-ankled bipeds, (2) we present an uncertainty analysis on the planner to identify sensing and control requirements, and (3) we experimentally validate our methods with our passive-ankled biped robot.

II. Robot System Description

A. Mercury Hardware and Usage Details

Mercury is the new name of our previous robot, Hume [14], with significant hardware upgrades (see Fig. 1). Joint

[1]Research scientist, Aerospace Engineering, University of Texas at Austin dk6587@utexas.edu
[2] NASA Space Tech. Research Fellow (NSTRF) and Mechanical Engineering PhD Candidate at the University of Texas at Austin, TX, USA. stevenjj@austin.utexas.edu
[3] Research Assistant, University of Texas at Austin and Robotics Engineering Bachelor, Hanyang University utmango15@utexas.edu
[4] Faculty of Aerospace Engineering, University of Texas at Austin, TX, USA lsentis@austin.utexas.edu

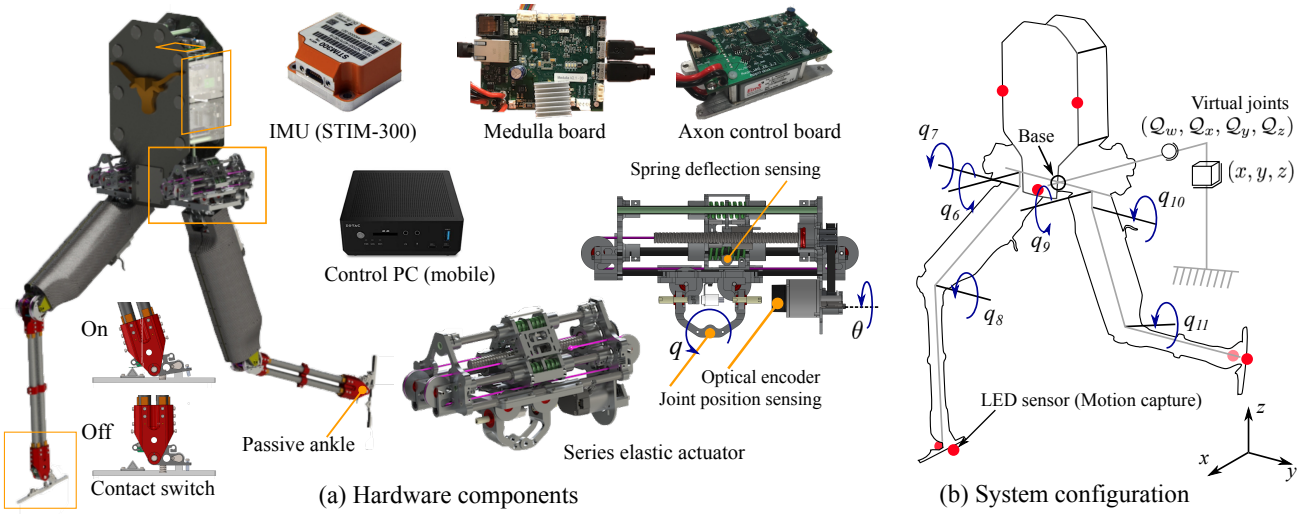


Fig. 1. **System description of Mercury.** Carbon fiber cases were installed on Mercury's thighs to increase structural stiffness. Spring-loaded passive-ankles with limit switches were also added to limit the uncontrollable yaw rotation and detect ground contacts. The IMU was upgraded to Sensonor's STIM-300, which has less angular velocity drift and bias, enabling accurate orientation estimates even with simple forward integration. Mercury's actuators are Meka SEAs with each having three sensors to measure joint position, spring deflection, and motor position. An absolute position encoder is used to measure the joint position, q, while a low-noise quadrature encoder measures motor position, θ. All of the embedded electronics were replaced with Apptronik's Medulla and Axon board system that comes with a variety of low-level controllers for SEAs. The robot's control PC is an off-the-shelf mobile PC (ZBOX-MI549) with RT preempt on Ubuntu 16.04 LTS to enable real-time control.

position sensing can be done either with the absolute encoder or through a transmission ratio with the motor's low-noise quadrature encoder. Our implementation uses the latter. When ground contact is unexpectedly detected by the limit switches on the spring-loaded passive ankles, Mercury's swing phase is terminated earlier than the specified swing duration to remove the jerk that would have been apparent from continuing to follow the designated swing leg trajectory.

B. Challenges in the Locomotion Control of Mercury

To highlight the locomotion challenge presented by Mercury, it is necessary to discuss the mass distribution of Mercury against other bipeds (Fig. 2)[1]. Mercury's mass distribution is similar to typical humanoid robots such as Valkyrie [16] or Atlas [7]. These robots have (1) a torso CoM location above the hip joint (around the length of half the body), and (2) the ratio between the total leg mass to the torso mass is not negligible (eg: greater than 0.4). On the other hand, ATRIAS [1] has a mass distribution optimized to be a mechanical realization of the inverted pendulum model, which aids with the implementation of locomotion controllers. Unlike typical humanoids, ATRIAS's torso CoM location is close to the hip joint and the ratio between the total leg mass to the torso is negligible (less than 0.1) .

While Mercury and ATRIAS are similar in their lack of ankle actuation and number of degrees-of-freedom, the difference in mass distributions have prominent consequences on the locomotion control difficulty. Since 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 body's orientation. In contrast, the CoM of Mercury and the other humanoids are located at half-a-body length above the hip joint, which creates a larger moment arm and increases the difficulty of body orientation control.

Next, since ATRIAS has negligible leg mass compared to its body, the body perturbations due to 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 which can compensate against Coriolis and gravitational forces introduced by the swing leg to maintain a desired body configuration, follow inverted pendulum dynamics, and control the swing foot to a desired landing location. Overall, in addition to Mercury's SEAs and lack of ankle actuation, its mass distribution make it a more difficult robot to control.

III. Control Scheme

The controller is a cascaded structure with three feedback loops: a kinematics controller, our whole-body controller, and a low-level joint controller with current feed-forward. For the following controller discussion, let $\mathbf{q}, \dot{\mathbf{q}} \in \mathbb{R}^n$ be the robot's current generalized position and velocity respectively with n degrees of freedom which includes the m actuated joints ($\mathbf{q}_{\text{act}}$), and noting that the first six elements of $\mathbf{q}$ are the floating base configuration.

A. Kinematics Controller

The kinematics control block computes the desired configuration, velocity, and acceleration terms, which are inputs to the whole body dynamic controller and low-level controllers (Fig. 3). A high level planner provides a desired Cartesian

[1]The robots' inertia information is available as open source. Valkyrie: `https://github.com/openhumanoids`, ATLAS: `https://github.com/dartsim/`, ATRIAS: `https://github.com/sir-avinash/atrias-matlab`

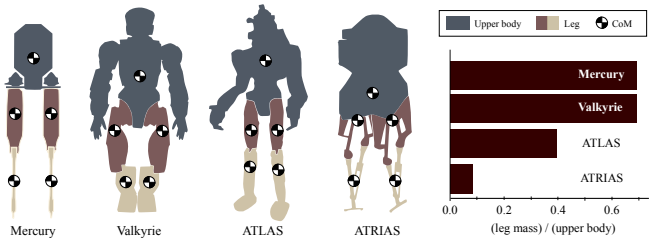


Fig. 2. **Mass distribution properties of some biped robots.** The center-of-mass locations are superimposed on the upper-body and leg links of each robot. The bar graph depicts the ratio of the total leg mass over the upper body mass. Notice that ATRIAS has a mass distribution property unlike typical humanoids by having a torso mass near the hip joint and a small leg-to-upper-body mass ratio.

posture, $\mathbf{x}^{\mathrm{d}} = \begin{bmatrix}\mathbf{x}_b^{\mathrm{d}}, & \mathbf{x}_f^{\mathrm{d}}\end{bmatrix}^\top$, where $\mathbf{x}_b^{\mathrm{d}} = \begin{bmatrix}h^{\mathrm{d}}, & \mathcal{Q}_b^{\mathrm{d}}\end{bmatrix}^\top \in \mathbb{R}^5$ is the desired base height, h^{d}, and orientation, $\mathcal{Q}_b^{\mathrm{d}}$ (denoted as a unit quaternion), while $\mathbf{x}_f^{\mathrm{d}} \in \mathbb{R}^3$ is the desired cartesian space foot position. To satisfy the desired Cartesian posture, the stance and swing leg configurations, $\mathbf{q}_{st}$ and $\mathbf{q}_{sw}$ respectively, are computed separately.

For controlling the base height and orientation, the Jacobian of the robot's base, $\boldsymbol{J}_b \in \mathbb{R}^{3\times n}$ is constructed with rows corresponding to the base height, roll, and pitch dimensions. For Mercury, the yaw orientation is assumed to be uncontrollable but fixed due to its passive foot. Thus, the base Jacobian is constructed without the yaw dimension. The configuration change needed to track the desired base configuration is defined as a single-step inverse kinematics (IK) with

$$\Delta\mathbf{q}_{st} = (\boldsymbol{J}_b\boldsymbol{\mathcal{N}}_c)^\dagger \Delta\mathbf{x}_{st}, \tag{1}$$

where $\Delta\mathbf{x}_{st} \in \mathbb{R}^3$ is the operational space error for the base height and orientation, $\boldsymbol{\mathcal{N}}_c = (\boldsymbol{I} - \boldsymbol{J}_c^\dagger \boldsymbol{J}_c)$ is the null space of the contact Jacobian, $\boldsymbol{J}_c$, and $\dagger$ is the pseudo-inverse operator. During the single support phase, the contact Jacobian is $\boldsymbol{J}_c = \boldsymbol{J}_i \in \mathbb{R}^{3\times n}$, with $i \in \{l, r\}$ indicating either the left or right foot, while in double support it is $\boldsymbol{J}_c = \begin{bmatrix}\boldsymbol{J}_l^\top, & \boldsymbol{J}_r^\top\end{bmatrix}^\top \in \mathbb{R}^{6\times n}$.

Let $h \in \mathbb{R}$ be the current base height, and $\boldsymbol{S}_{\mathrm{ori}} \in \mathbb{B}^{2\times 3}$ be a binary matrix which selects only the roll and pitch components of the axis-angle representation of the orientation error, $\mathbf{w}_e$. Then, $\Delta\mathbf{x}_{st}$ is defined as

$$\Delta\mathbf{x}_{st} = \begin{bmatrix}(h^{\mathrm{d}} - h), & \boldsymbol{S}_{\mathrm{ori}}\mathbf{w}_e\end{bmatrix}^\top \in \mathbb{R}^3. \tag{2}$$

Thus, the desired base configuration can be tracked by setting the desired configurations to be

$$\mathbf{q}_{st} = \mathbf{q} + \begin{bmatrix}\mathbf{0}_{1\times 6}, & (\boldsymbol{S}_{\mathrm{act}}\Delta\mathbf{q}_{st})^\top\end{bmatrix}^\top, \tag{3}$$

where $\boldsymbol{S}_{\mathrm{act}} \in \mathbb{B}^{m\times n}$ is a binary matrix extracting only the components of $\Delta\mathbf{q}_{st}$ related to actuated joints.

During single support, one of the legs will be in the swing phase. There are many ways to specify the joint trajectory of the swing foot, but what was used for the experiment is described as follows. In the beginning of the swing phase, the high-level planner provides a desired foot retracting location. The $x-y$ components of the retracting location are set to be the midpoint between the current swing foot location and a default landing location, which is a fixed offset from the stance foot frame. The z-component is set to a default swing height. Then, at the mid-point of the swing trajectory, the planner provides a desired landing location. For either case, an IK is performed on the swing foot while assuming that the robot's base orientation is aligned vertically. This is effectively a simple fixed-manipulator IK with the assumption that the support leg is tracking the desired base configuration. The final swing leg joint position can be obtained, and a sinusoidal spline trajectory is constructed using initial, mid, and final joint positions to define boundary conditions. When the high-level planner provides a new desired foot landing location, a new joint trajectory is reconstructed with corresponding boundary conditions for smoothness.

Finally, the computed desired configuration and its two time-derivatives depend on the contact configuration of the robot. Namely,

$$\mathbf{q}^{\mathrm{d}} = \begin{cases}\mathbf{q}_{st} & \text{if double support} \\ \begin{bmatrix}\mathbf{0}_{1\times 6}, & (\boldsymbol{S}_{st}\mathbf{q}_{st} + \boldsymbol{S}_{sw}\mathbf{q}_{sw})^\top\end{bmatrix}^\top & \text{else}\end{cases} \tag{4}$$

where $\mathbf{q}_{sw}$ is the swing leg trajectory, $\boldsymbol{S}_{st} \in \mathbb{B}^{m\times n}$ and $\boldsymbol{S}_{sw} \in \mathbb{B}^{m\times n}$ are binary matrices that only select the stance and swing joint components respectively with the property that an element extracted by $\boldsymbol{S}_{st}$ is set to zero by $\boldsymbol{S}_{sw}$ and vice-versa. Then, the desired configuration, velocity, and acceleration are inputs to the whole-body dynamic controller (WBDC) which compute torque command.

B. Whole-Body Dynamic Controller (WBDC)

A simple version of the whole-body dynamic controller (WBDC) proposed in [5] is used here in which the operational space tasks are simply single joint position control tasks. A necessary condition of WBDC is that the contact constraints and the first task must span the floating base dynamics to make the reaction forces computationally (Eq. (5)) feasible. Thus, the joint position control task must include the floating base configurations in the task definition. The purpose of including the floating base is simply to solve the optimization problem rather than actually controlling the base. Therefore, we use small weights in the cost matrix of the QP's objective function for the floating base configurations.

Given a desired configuration, velocity, acceleration, WBDC finds the instantaneous feed forward joint torque values needed to cancel the robot dynamics subject to unilateral contact constraints. Given the current state, WBDC solves the following QP:

$$\min_{\mathbf{F}_r, \boldsymbol{\rho}} \quad \mathbf{F}_r^\top \boldsymbol{W}\mathbf{F}_r + \boldsymbol{\rho}^\top \boldsymbol{R}\boldsymbol{\rho} \tag{5}$$

$$\text{s.t.} \quad \boldsymbol{S}_f(\boldsymbol{A}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g} - \boldsymbol{J}_c^\top \mathbf{F}_r) = \mathbf{0}_{6\times 1} \tag{6}$$

$$\boldsymbol{U}\mathbf{F}_r \geq \mathbf{0} \tag{7}$$

$$\ddot{\mathbf{q}}^{\mathrm{cmd}} = \ddot{\mathbf{q}}^{\mathrm{d}} + \boldsymbol{K}_D(\dot{\mathbf{q}}^{\mathrm{d}} - \dot{\mathbf{q}}) + \boldsymbol{K}_P(\mathbf{q}^{\mathrm{d}} - \mathbf{q}), \tag{8}$$

$$\ddot{\mathbf{q}} = -\overline{\boldsymbol{J}}_c\dot{\boldsymbol{J}}_c\dot{\mathbf{q}} + \overline{\boldsymbol{N}}_c\left((\ddot{\mathbf{q}}^{\mathrm{cmd}} + \boldsymbol{\rho}) + \overline{\boldsymbol{J}}_c\dot{\boldsymbol{J}}_c\dot{\mathbf{q}}\right), \tag{9}$$

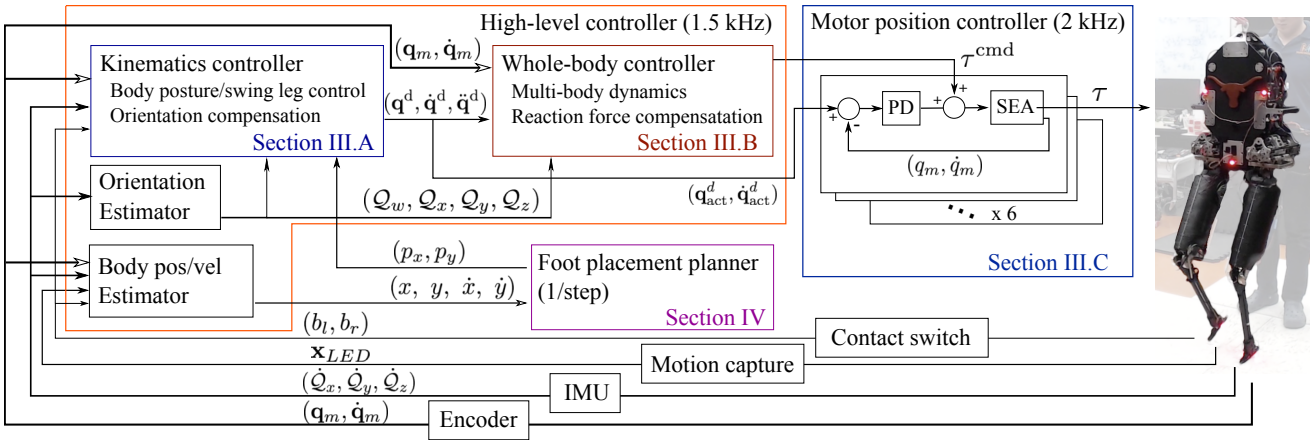


Fig. 3. **Diagram of the Control Methodology.** Our controller has a cascaded structure with three feedback loops. b_l and b_r are left and right foot contact signals. $\mathbf{x}_{\text{LED}}$ is the position of MoCap LED sensors. (1) The kinematics controller computes joint trajectories based on desired operational space tasks. (2) The whole-body controller computes contact-consistent torque commands to cancel robot dynamics subject to unilateral constraints. (3) Low-level PD controllers on the motor positions with feed-forward currents from the computed torques are used to achieve the desired joint configurations. The footstep planner plans a single desired foot landing location per step at the midpoint of the swing leg trajectory. Note that $\mathbf{q}_m$ are joint positions computed from the motor positions via a transmission ratio.

where $\mathbf{F}_r$ is the contact reaction forces for a corresponding contact Jacobian, $\boldsymbol{J}_c$, $\boldsymbol{W}$ and $\boldsymbol{R}$ are cost matrices, $\boldsymbol{\rho}$ is a task command relaxation term, $\boldsymbol{A}$ is the inertia matrix with additional motor rotor inertia terms on the diagonal corresponding to the actuated joints, $\mathbf{b}$ and $\mathbf{g}$ are the Coriolis and gravitational forces, $\boldsymbol{U}$ describes the unilateral constraints with a polyhedral approximation of the friction cone, $\overline{(\cdot)}$ is the dynamically-consistent pseudo inverse of $(\cdot)$, defined as $\overline{\boldsymbol{X}} = \boldsymbol{A}^{-1}\boldsymbol{X}^\top(\boldsymbol{X}\boldsymbol{A}^{-1}\boldsymbol{X}^\top)^{-1}$. $\boldsymbol{N}_c$ $(= \boldsymbol{I} - \overline{\boldsymbol{J}}_c\boldsymbol{J}_c)$ is a dynamically consistent null-space projection. $\boldsymbol{S}_f$ is a binary matrix extracting only the floating base dynamics. Eq.(9) enforces consistency with the contact constraints. Then the torque command can be extracted via

$$\begin{bmatrix}\mathbf{0}_{6\times 1}\\ \boldsymbol{\tau}^{\text{cmd}}\end{bmatrix} = \boldsymbol{A}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g} - \boldsymbol{J}_c^\top \mathbf{F}_r, \tag{10}$$

where $\boldsymbol{\tau}^{\text{cmd}}$ is the joint torque vector used as feed-forward input to the low-level controller by converting it to currents using motor-torque constants. During contact transitions between single and double support phases, the bounds of the reaction forces are smoothly changed to avoid QP solution discontinuities. Refer to our previous work [5] for details about smooth contact transitions and cost weights.

C. Low-Level Controller: PD with Current Feed-Forward

Each actuated joint has a low-level controller which takes as input the previous feed-forward torque, $\boldsymbol{\tau}^{\text{cmd}}$, and a desired joint position and velocity $(\mathbf{q}_{\text{act}}, \dot{\mathbf{q}}_{\text{act}})$. The input torque is converted as a feed-forward current to the motor. A PD controller is used to track the desired joint position and velocity using the motor position and velocity data as feedback signals. We compute the joint positions from motor positions using the transmission ratio, N. Currently, the spring deflection is ignored when calculating joint positions from motor positions. While it is also possible to use the joint's absolute position encoders as the feedback signals, our empirical tests show that we obtain better joint velocity tracking when using motor position and velocity as feedback signals. The feed-forward currents also increase the joint position and velocity tracking performance.

D. Application of the Generic Algorithm on Multiple Robots

The proposed methodology is generic and applicable to many types of biped robots. To demonstrate, we verify walking with Mercury, Valkyrie, and Atlas in a physics-based simulation (Fig. 4). For Valkyrie and Atlas, we also include the pelvis and torso orientations as part of $\boldsymbol{J}_b$. The proposed algorithm and simulation results are published as open-source[2].

IV. Uncertainty Analysis Of The Planner

Fundamentally, our locomotion planner observes the CoM position and velocity states, $\mathbf{x} = [x, \dot{x}]^\top \in \mathbb{R}^2$ and computes a foot landing location $p \in \mathbb{R}$. The planner implemented in Mercury is the same one as the one presented in [14]. Here, we enforce a linear height surface constraint to simplify the implementation and analysis, but the formulation can address any type of smooth surfaces. The use of the linear inverted pendulum (LIP) model simplifies the planner formulation and enables the uncertainty analysis of noisy CoM state observations and landing location errors under kinematic constraints.

A. Formulation of Planner

The planner is formulated based on the LIP model:

$$\ddot{x} = \frac{g}{h}(x - p), \tag{11}$$

where g and h are the gravity and constant CoM height, respectively. Regarding p as the input for the LIP dynamics, the robot is stabilized by reversing the CoM direction after every step. Concretely, the planner aims to reverse the CoM

[2] https://github.com/dhkim0821/Humanoid_2018

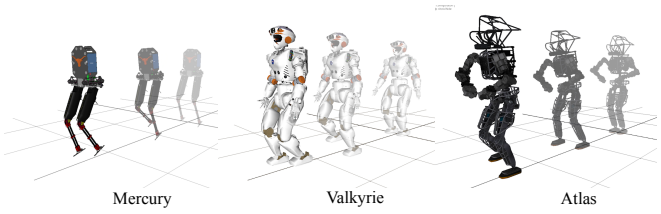


Fig. 4. **Walking simulation of three robots.** Our control methodology enables locomotion of three different biped humanoids, Mercury, Valkyrie, and Atlas. For Valkyrie and Atlas, underactuation of the ankles are simulated by adding only damping to the joint.

velocity after a pre-defined duration t' has elapsed by finding a new stance foot location p. Notice that Eq. (11) is linear so it has an exact solution for the CoM state, $\mathbf{x}(t)$. Thus, for a given p, the CoM state after time T can be described as a discrete system with

$$\mathbf{x}_{k+1} = \boldsymbol{A}\mathbf{x}_k + \boldsymbol{B}p_k, \tag{12}$$

$$\boldsymbol{A} = \begin{bmatrix} \cosh(\omega T) & \omega^{-1}\sinh(\omega T) \\ \omega \sinh(\omega T) & \cosh(\omega T) \end{bmatrix}, \tag{13}$$

$$\boldsymbol{B} = \begin{bmatrix} 1 - \cosh(\omega T) \\ -\omega \sinh(\omega T) \end{bmatrix}, \tag{14}$$

where k indicates the k-th locomotion step and $\omega = \sqrt{g/h}$. Since the goal is to select a p_k which reverses the CoM velocity, let the velocity component (bottom row) of Eq. (12) be zero after time, t', resulting into

$$0 = \begin{bmatrix} \omega \sinh(\omega t') & \cosh(\omega t') \end{bmatrix} \mathbf{x}_k - \omega \sinh(\omega t') p_k. \tag{15}$$

Observe that solving for p_k in Eq. (15) will give a stance foot location that will cause the CoM velocity to be zero after a duration of t' and negative immediately after. With the CoM velocity being reversed after every step, an additional bias term, κ, is added to move the robot toward the origin. Further details about κ are presented in [14]. As a result, the landing location, p_k is defined by

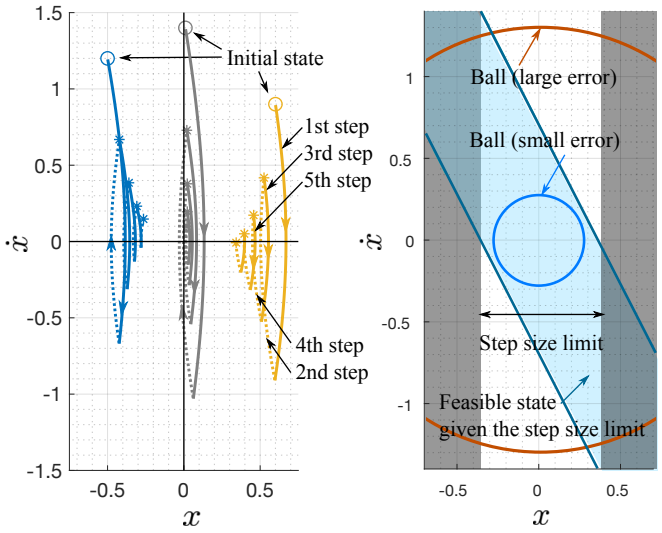
$$p_k = \begin{bmatrix} 1 & \omega^{-1}\coth(\omega t') \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} \kappa & 0 \end{bmatrix} \mathbf{x}_k. \tag{16}$$

This results into equivalent discrete step dynamics,

$$\mathbf{x}_{k+1} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})\mathbf{x}_k, \tag{17}$$

$$\boldsymbol{K} = \begin{bmatrix} (1+\kappa) & \omega^{-1}\coth(\omega t'), \end{bmatrix} \tag{18}$$

Notice that the planner ends up with a PD control form; therefore, by applying the stability criteria of PD controllers, the planner parameters, (κ, t'), can be tuned by setting magnitudes of eigenvalues of $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$ smaller than 1. In our case, we use an eigenvalues with magnitude 0.8. Since the desired behavior is to take multiple small steps toward the origin rather than a single big step, the eigenvalue magnitudes are intentionally set to be close to one rather than zero. The desired motion is visualized as a converging oscillatory trajectory in a phase plot, Fig. 5(a).



(a) Phase trajectory (b) Uncertainty analysis

Fig. 5. **Phase plot and uncertainty analysis of the LIP-based planner.** In (a), the phase trajectories of 8 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.

B. *Uncertainty Analysis*

During walking tests, we observed notable body position and landing location errors due to the deflection of the mechanical structures. To quantify the acceptable error for the planner, we analyze the stable region of states given error magnitudes by borrowing ideas from robust control [17]. We limit the scope of analysis on the LIP dynamics and the planner with the following assumptions:

1) The step size limit is set to 0.4 m by approximating the robot's kinematic limits.
2) State-dependent errors are ignored.

Since large errors in our tests comes from the undetected mechanical deflections, there is no direct mapping between these errors and the state. However, we are able to reason about the errors due to the state observation and control inputs. Based on this assumption, the discrete step dynamics are rewritten with terms representing the uncertainty,

$$\begin{aligned} \mathbf{x}_{k+1} &= \boldsymbol{A}\mathbf{x}_k + \boldsymbol{B}(p_k + \eta), \\ p_k &= \boldsymbol{K}(\mathbf{x}_k + \boldsymbol{\delta}), \end{aligned} \tag{19}$$

where η and $\boldsymbol{\delta}$ represent the landing location error and the observation error respectively. Furthermore, the errors are assumed to be bounded by their maximum values:

$$||\boldsymbol{\delta}|| \leq \delta_M, \quad ||\eta|| \leq \eta_M. \tag{20}$$

Since the velocity of the state changes sign after every step, typical convergence analysis regards this as an unstable oscillatory behavior despite the fact that the absolute value of $\mathbf{x}$ decreases as we require. To enable convergence analysis, we analyze the dynamics after two steps instead of a single

step (Eq. (19)). Therefore, from an initial state, $\mathbf{x}$, after two steps, the new state, $\mathbf{x}''$, is defined as

$$\begin{aligned} \mathbf{x}'' &= \boldsymbol{A}^2\mathbf{x} + \boldsymbol{A}\boldsymbol{B}(p+\eta) + \boldsymbol{B}(p'+\eta'), \\ p &= \boldsymbol{K}\left(\mathbf{x}+\boldsymbol{\delta}\right), \\ p' &= \boldsymbol{K}\left(\mathbf{x}'+\boldsymbol{\delta}'\right), \end{aligned} \tag{21}$$

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

$$\Delta V = \mathbf{x}''^{\top}\boldsymbol{P}\mathbf{x}'' - \mathbf{x}^{\top}\boldsymbol{P}\mathbf{x} \leq 0. \tag{22}$$

Substituting Eq. (21) and arranging the terms, and observing that ΔV has an upper bound,

$$\begin{aligned} \Delta V &= \mathbf{x}^{\top}(\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{A}_{cc} - \boldsymbol{P})\mathbf{x} + 2\boldsymbol{\zeta}^{\top}\boldsymbol{P}\boldsymbol{A}_{cc}\mathbf{x} + \boldsymbol{\zeta}^{\top}\boldsymbol{P}\boldsymbol{\zeta} \\ &\leq -a||\mathbf{x}||^2 + 2b||\mathbf{x}|| + c \\ &\leq 0, \end{aligned} \tag{23}$$

where,

$$\boldsymbol{A}_c = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K} \tag{24}$$

$$\boldsymbol{A}_{cc} = \boldsymbol{A}_c^2 \tag{25}$$

$$\boldsymbol{\zeta} = \boldsymbol{A}_c\boldsymbol{B}\boldsymbol{K}\boldsymbol{\delta} + \boldsymbol{B}\boldsymbol{K}\boldsymbol{\delta}' + \boldsymbol{A}_c\boldsymbol{B}\eta + \boldsymbol{B}\eta' \tag{26}$$

$$a = -\lambda_M\left(\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{A}_{cc} - \boldsymbol{P}\right), \tag{27}$$

$$\begin{aligned} b = \delta_M\left(||\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B}\boldsymbol{K}|| + ||\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{B}\boldsymbol{K}||\right) + \\ \eta_M\left(||\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{A}_c\boldsymbol{B}|| + ||\boldsymbol{A}_{cc}^{\top}\boldsymbol{P}\boldsymbol{B}||\right), \end{aligned} \tag{28}$$

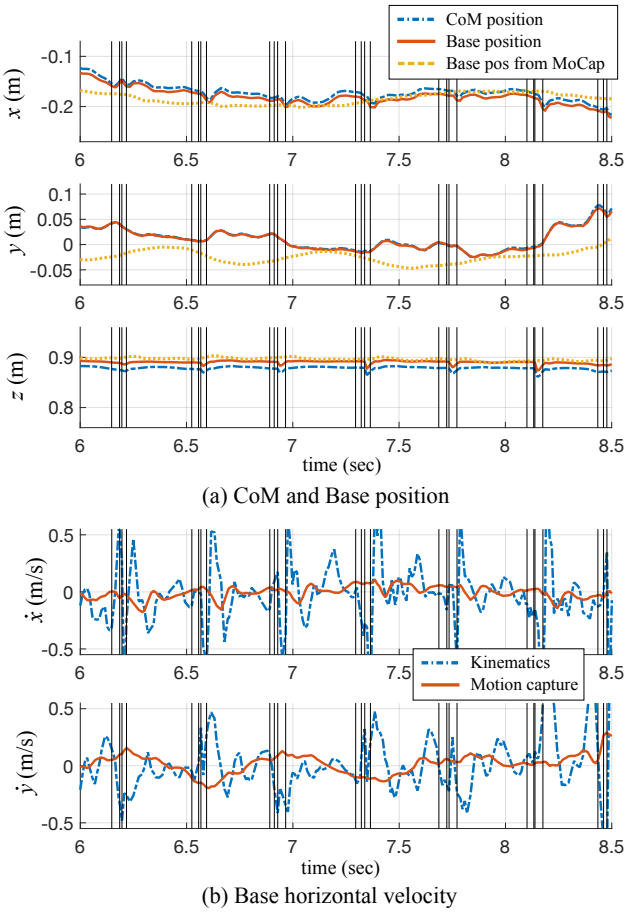
$$c = g(\boldsymbol{\zeta}^{\top}\boldsymbol{P}\boldsymbol{\zeta}) \tag{29}$$

$||.||$ is the l^2-norm, $\lambda_M(.)$ denotes the maximum eigenvalue of the matrix, 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. Note that a is positive if the planner parameters are tuned so that the LIP is stable. Now, we can define a ball region based on Eq. (23):

$$B_r = \left\{ \mathbf{x} \;\middle|\; ||\mathbf{x}|| \leq \frac{b + \sqrt{b^2 + ac}}{a} \right\}. \tag{30}$$

Then, $\mathbf{x} \notin B_r$ are asymptotically stable states as ΔV will be negative. Note that a smaller ball means a bigger stability region, and if the errors are zero, the ball will have a 0 radius and any state will be asymptotically stable.

By substituting the planner's parameters from Table I into Eq. (30), we obtain information about the magnitude of allowable errors. Fig. 5(b) shows the feasible state region given the step size limit, and the uncertainty regions with two different error setups. The orange circle indicates the uncertain region of the state when the observation error is $0.03\mathrm{m}$, and the foot landing location error is $0.045\mathrm{m}$. These errors are close to what we have observed in our walking tests before we included the motion capture (MoCap) data in our state-estimator and the body orientation based swing leg trajectory adjustment (see Section V-C). After making these additional implementation improvements, the blue circle indicates a $2\mathrm{mm}$ state observation error, which coincides with the accuracy of the MoCap system, and a reduced foot landing location error of $0.015\mathrm{m}$. As shown in Fig. 5(b), this uncertain region with higher accuracy (blue circle) is embedded within the feasible state set (blue area), implying that the planner can dynamically stabilize the robot from many starting states.



(a) CoM and Base position

(b) Base horizontal velocity

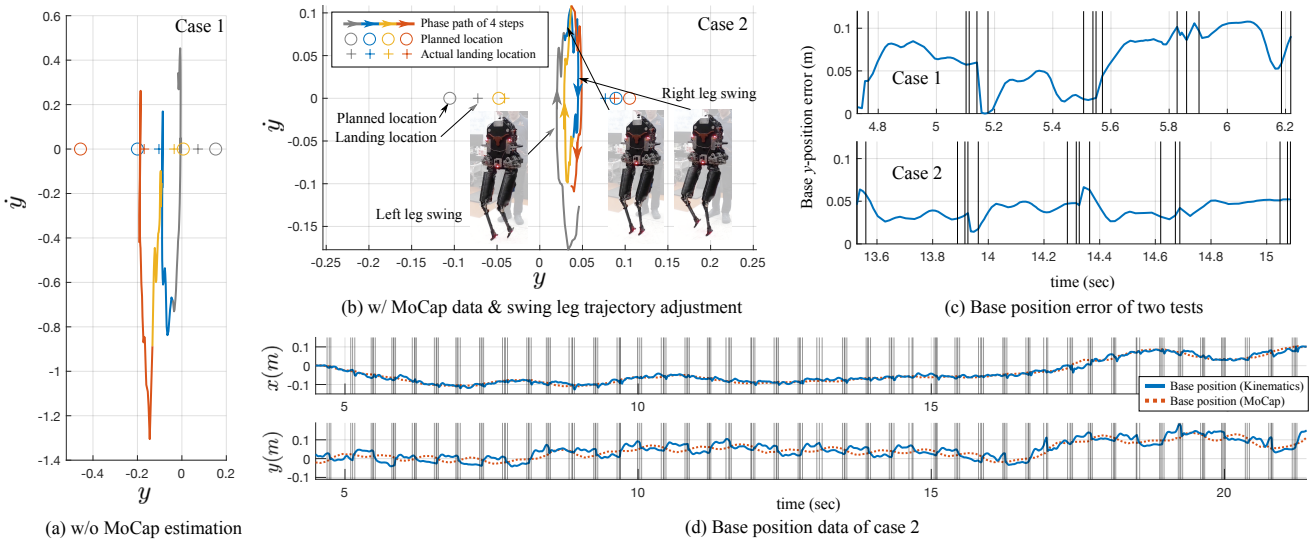
Fig. 6. **Comparison between the computed Base and CoM state and the MoCap data from a sample walking test.** x, y, z correspond to forward, lateral and vertical directions. The black vertical lines indicate phase changes during walking, ie: double to single support. In (a), the computed CoM and base positions from kinematics and the sensed position from the MoCap data are plotted. The difference between the computed CoM and base positions is negligible. However, the difference between the computed and actual positions differ especially in the lateral direction, which can have errors of over $5\mathrm{cm}$. (b) The base velocity computed from kinematics shows serious fluctuation compared to its true velocity. Note that computations from kinematics utilize on-board sensing data from the IMU and encoders.

V. Implementation Details

The additional implementation details are aimed at lowering the body's state estimation and the foot's landing location errors to satisfy sensing and control requirements from Section IV-B.

A. Using the Base state instead of the CoM state

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 base position (Fig. 1(b)), and assumes that the CoM of the robot is always at this location. As seen in Fig. 6(a), the difference between the CoM and



(a) w/o MoCap estimation

(b) w/ MoCap data & swing leg trajectory adjustment

(c) Base position error of two tests

(d) Base position data of case 2

Fig. 7. **Experimental results from two dynamic balancing test cases.** Using the same controller and planner parameters, we test two scenarios: case 1 in which the robot does not use the MoCap and the additional swing leg adjustment described in Section V, and case 2 in which they are used. (a) The lateral-direction phase path of four steps from case 1 shows that state velocities go over $0.5\mathrm{m/s}$, which is five times larger than case 2, and foot landing locations typically greater than $0.045\mathrm{m}$. (b) An example phase path of four steps from a 50 step run using MoCap-based estimation and swing leg trajectory adjustments described in Section V-C. The velocity state is bounded within $0.1\mathrm{m/s}$ with foot landing locations typically less than $0.015\mathrm{m}$ (c) The lateral direction error between the estimated kinematics body position and the MoCap data are presented for both cases. Notice that the observation errors from kinematic computations go over $0.05\mathrm{m}$, which is unacceptable according to our uncertainty analysis (Section IV-B). Thus, we need to utilize the MoCap data to demonstrate the controller with our hardware. (d) A time-series plot of the base position, which demonstrates that our control methodology achieves prolonged unsupported dynamic balancing.

base positions are small. This enables us to (1) decouple the computation of the CoM state from the swing leg motion, and (2) perform a straight-forward sensor-fusion process with a Kalman filter by combining the sensed body positions from joint-encoders and the overhead MoCap system.

B. Base State Estimation With Motion Capture System

Fig. 6(a) compares the base positions between MoCap measurements and kinematics-based computations. The estimated positions are notably different in the lateral directional which can have errors over $0.05\mathrm{m}$. From the analysis presented in Section. IV-B, this error is not acceptable. Additionally, the base velocity estimation from kinematics fluctuates in all phases of walking when compared to the MoCap velocity estimation (Fig. 6(b)). To reduce base state estimation errors, the MoCap data is included in the state estimator. Since the LEDs of the MoCap is susceptible to self-occlusion, we also perform sensor-fusion between the computed base states from kinematics and the MoCap data with traditional Kalman-Filtering techniques.

C. Swing Leg Trajectory Adjustment

Since the swing leg uses a joint position trajectory rather than a cartesian space trajectory, the foot landing location will depend on the torso orientation. To compensate, the desired joint position of the abduction and hip joints are adjusted based on the torso orientation. Thus, after a desired landing location is determined by the planner at half swing-time, the new swing leg joint position command is set to

$$\begin{aligned} q^{\mathrm{d}}_{\mathrm{abd}} &= s_{\mathrm{abd}}(t) + K_\phi \phi, \\ q^{\mathrm{d}}_{\mathrm{hip}} &= s_{\mathrm{hip}}(t) + K_\psi \psi, \end{aligned} \tag{31}$$

where $s_{(.)}$ is the planned joint trajectory, and ϕ and ψ are the error roll and pitch angles of the body, respectively. This simple adjustment reduces our nominal landing location error from $0.045\mathrm{m}$ to $0.015\mathrm{m}$.

VI. Experimental Results

We present experimental results to demonstrate our control algorithm on the Mercury biped with passive-ankles. We also compare two trials to verify the accuracy requirements of our uncertainty analysis. To begin the experiment, Mercury is briefly supported by the experimenter while it pushes itself to a desired height. Once it reaches its starting height, the experimenter lets go as it takes its first step. The walking behavior is a time-scripted state machine, ie: double support phase $\rightarrow$ contact transition phase $\rightarrow$ single support and leg swing phase $\rightarrow$ double support phase. While state transitions occur after a predefined time duration, the swing phase is terminated earlier if contact occurs before the swing time is over. Temporal and planner parameters are listed in Table. I, which correspond to a walking rate of 2.7 steps per second.

Fig. 7 shows a summary of the experimental results. The plots only focus on the lateral directional movement as stabilizing this direction is more difficult for Mercury due to a small feasible step size and larger observed errors between the kinematic estimate and the MoCap data as seen in Fig. 6(a). The step size is limited by the abduction joints' small range of motion and the kinematic constraint preventing Mercury from crossing its legs.

In test case 1, the MoCap data and the swing leg adjustment as described in Section V are not used. The resulting walking behavior has high lateral velocities, landing location

Double stance	Transition	Swing	t'	κ
0.01 sec	0.03 sec	0.33 sec	[0.18, 0.18]	[0.12, 0.09]

TABLE I

PLANNER PARAMETERS

errors anywhere from $0.01\mathrm{m}$ to $0.05\mathrm{m}$ (Fig 7 (a)), and state estimation errors of up to $0.1\mathrm{m}$ (top plot of Fig 7 (c)). As predicted by our uncertainty analysis, the state estimation and foot landing location errors are too large, and Mercury fails to dynamically balance after only six steps.

In contrast, test case 2 performs sensor fusion with the MoCap data and adjusts the swing leg to address the sensing and control requirements of our uncertainty analysis (Section. IV-B). Fig. 7(b) shows that lateral velocities are bounded within $0.1\mathrm{m/s}$, and foot landing location errors that are nominally less than $0.015\mathrm{m}$. While the state estimation errors between kinematic computations and the MoCap data are still high (bottom plot of Fig 7 (c)), test case 2 uses the MoCap data for state estimation which brings the state estimation error down to the MoCap's sensing error of 2mm. With a state observation error of 2mm and nominal foot landing locations of $0.015\mathrm{m}$, our uncertainty analysis predicts a stable walking behavior. As a result, we achieve the challenging unsupported extended dynamic balancing behavior (Fig. 7(d)) that our methodology had targeted. [3]

VII. CONCLUSIONS AND DISCUSSIONS

We demonstrate extended dynamic balancing of a biped humanoid with no-ankle actuation using a novel locomotion-control scheme. The algorithmic generality has also been verified with walking simulations of three biped humanoids. We performed an uncertainty analysis of our footstep planner and found maximum allowable errors for our state estimator and controllers, which enabled us to address additional sensing and control requirements. By integrating a high-performance whole-body feedback controller, a robust locomotion planner, and a reliable state estimator, our passive-ankled biped robot accomplishes extended unsupported dynamic balancing.

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 vs leg joint position control, base vs CoM position control, having vs not having task priorities, etc. In the low-level controller we also experimented with torque feedback with disturbance observers, joint vs motor position feedback, and joint position control with and without feed-forward torques. The methodology presented here is our best performing controller after a system-level integration. Our next paper will perform additional analysis and experiments at length to characterize more properties of the controller.

ACKNOWLEDGMENTS

The authors would like to thank the members of the Human Centered Robotics Laboratory at The University of Texas at Austin for their support. This work was supported by the Office of Naval Research, ONR grant #N000141512507, NASA Johnson Space Center, NSF/NASA NRI grant #NNX12AM03G, and a NASA Space Technology Research Fellowship (NSTRF) grant #NNX15AQ42H.

[3] https://youtu.be/C5MzNbl-CkI

REFERENCES

[1] C. Hubicki, A. Abate, P. Clary, S. Rezazadeh, M. Jones, A. Peekema, J. V. Why, R. Domres, A. Wu, W. Martin, H. Geyer, and J. Hurst, "Walking and running with passive compliance: Lessons from engineering a live demonstration of the atrias biped," *IEEE Robotics Automation Magazine*, pp. 1–1, 2018.

[2] R. Hartley, X. Da, and J. W. Grizzle, "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, 2017, pp. 1262–1269.

[3] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based full body control for the darpa robotics challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[4] T. Koolen, S. Bertrand, G. Thomas, T. De Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt, "Design of a Momentum-Based Control Framework and Application to the Humanoid Robot Atlas," *International Journal of Humanoid Robotics*, vol. 13, no. 01, pp. 1 650 007–35, Mar. 2016.

[5] D. Kim, J. Lee, O. Campbell, H. Hwang, and L. Sentis, "Computationally-Robust and Efficient Prioritized Whole-Body Controller with Contact Constraints," *arXiv.org*, Jul. 2018.

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

[7] S. Kuindersma, R. Deits, M. Fallon, and A. Valenzuela, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous Robots*, vol. 40, no. 3, pp. 429–455, 2015.

[8] S. Kohlbrecher, A. Romay, A. Stumpf, A. Gupta, O. von Stryk, F. Bacim, D. A. Bowman, A. Goins, R. Balasubramanian, and D. C. Conner, "Human-robot Teaming for Rescue Missions: Team ViGIR's Approach to the 2013 DARPA Robotics Challenge Trials," *Journal of Field Robotics*, vol. 32, no. 3, pp. 352–377, Dec. 2014.

[9] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based Full Body Control for the DARPA Robotics Challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, Jan. 2015.

[10] M. Johnson, B. Shrewsbury, S. Bertrand, T. Wu, D. Duran, M. Floyd, P. Abeles, D. Stephen, N. Mertins, A. Lesman, J. Carff, W. Rifenburgh, P. Kaveti, W. Straatman, J. Smith, M. Griffioen, B. Layton, T. De Boer, T. Koolen, P. Neuhaus, and J. Pratt, "Team IHMC's Lessons Learned from the DARPA Robotics Challenge Trials," *Journal of Field Robotics*, vol. 32, no. 2, pp. 192–208, Feb. 2015.

[11] N. A. Radford, P. Strawser, K. Hambuchen, J. S. Mehling, W. K. Verdeyen, A. S. Donnan, J. Holley, J. Sanchez, and et al., "Valkyrie: NASA's First Bipedal Humanoid Robot," *Journal of Field Robotics*, vol. 32, no. 3, pp. 397–419, May 2015.

[12] M. A. Hopkins, A. Leonessa, B. Y. Lattimer, and D. W. Hong, "Optimization-Based Whole-Body Control of a Series Elastic Humanoid Robot," *International Journal of Humanoid Robotics*, pp. 1 550 034–34, Jul. 2015.

[13] J. Englsberger, C. Ott, and A. Albu-Schaffer, "Three-Dimensional Bipedal Walking Control Based on Divergent Component of Motion." *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[14] D. Kim, Y. Zhao, G. Thomas, B. R. Fernandez, and L. Sentis, "Stabilizing Series-Elastic Point-Foot Bipeds Using Whole-Body Operational Space Control," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1362–1379, 2016.

[15] Y. Zhao, N. Paine, and L. Sentis, "Feedback parameter selection for impedance control of series elastic actuators," *IEEE-RAS International Conference on Humanoid Robots*, pp. 999–1006, 2014.

[16] N. A. Radford and et al., "Valkyrie: NASA's First Bipedal Humanoid Robot," *Journal of Field Robotics*, vol. 32, no. 3, pp. 397–419.

[17] A. A. Bahnasawi, A. S. Al-Fuhaid, and M. S. Mahmoud, "Linear feedback approach to the stabilisation of uncertain discrete systems," *IEE Proceedings D Control Theory and Applications*, vol. 136, no. 1, p. 47, 1989.