

Bipedal Hopping: Reduced-order Model Embedding via Optimization-based Control

Xiaobin Xiong and Aaron D. Ames

Abstract—This paper presents the design and validation of controlling hopping on the 3D bipedal robot Cassie. A spring-mass model is identified from the kinematics and compliance of the robot. The spring stiffness and damping are encapsulated by the leg length, thus actuating the leg length can create and control hopping behaviors. Trajectory optimization via direct collocation is performed on the spring-mass model to plan jumping and landing motions. The leg length trajectories are utilized as desired outputs to synthesize a control Lyapunov function based quadratic program (CLF-QP). Centroidal angular momentum, taking as an addition output in the CLF-QP, is also stabilized in the jumping phase to prevent whole body rotation in the underactuated flight phase. The solution to the CLF-QP is a nonlinear feedback control law that achieves dynamic jumping behaviors on bipedal robots with compliance. The framework presented in this paper is verified experimentally on the bipedal robot Cassie.

I. INTRODUCTION

Reduced-order models such as the canonical Spring Loaded Inverted Pendulum (SLIP) have been widely applied for controlling walking [2] [3] [4], running [5] and hopping [6] of legged robots. One important benefit of using low order dynamical systems for control is that it renders the gait and motion generation problems for legged robots computationally tractable. However, reduced-order models are often directly implemented on the full-order model of the robot, e.g., through inverse kinematics [7] or inverse dynamics [8], without a faithful connection to the structure and morphology of the robot.

In this paper, we present an approach to identifying the spring-mass model for bipedal robots with mechanical compliance, and synthesizing nonlinear controllers by embedding the spring-mass model into the full-order dynamics. Specifically, the spring in the spring-mass model comes from viewing each leg as a deformable prismatic spring as motivated by the mechanical design of robots with compliance [9]. We borrow the idea of end-effector stiffness from manipulation community [10], and formally derive the stiffness/damping of the leg spring from the compliant components in the leg as functions of robot configurations. This facilitates the spring-mass model being virtually actuated by changing robot configurations, i.e., by changing the leg length on the spring-mass model. Trajectory optimization can thus be utilized to create hopping behaviors on the spring-mass model.

The planned leg length trajectory from the spring-mass model encodes the underactuation of the leg compliance of the robot, and can therefore be used to synthesize controllers

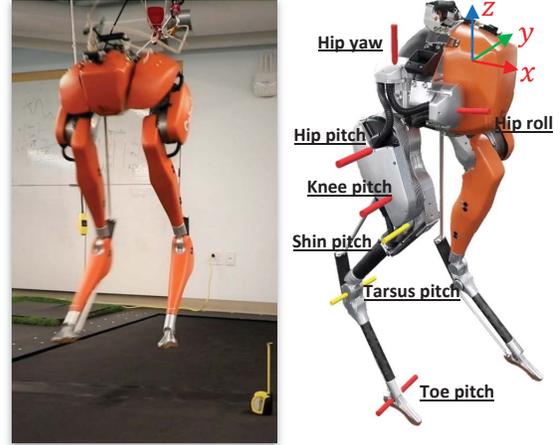


Fig. 1. Hopping on Cassie [1] (left) and its coordinate system (right).

that achieve this behavior on the full-order model. Roughly speaking, the springs on the physical robot are expected to behave similarly to the spring in the spring-mass model when the robot tracks the leg length trajectory. This motivates defining the leg length trajectory as a desired output on each leg and thus formulate a control Lyapunov function based quadratic program (CLF-QP) [11] [12] for output stabilization. The end result is a nonlinear optimization-based controller that represents the reduced-order dynamics in the full-order model of the robot.

The QP formulation for hopping is inspired by the approach for walking in [4], wherein the SLIP dynamics is embedded onto the center of mass (COM) dynamics of the full robot via an equality constraint in the QP. The difference of our approach is that the hopping dynamics is embedded by taking the leg length trajectory as a desired output, which becomes an inequality CLF constraint in the QP, rendering a more feasible QP formulation. Additionally, the hopping motion naturally requires a consideration on momentum regulation due to the conservation law on centroidal angular momentum [13] in flight phase. This is done by including the angular momentum as an output to stabilize in the CLF-QP.

The proposed approach is successfully implemented on the 3D underactuated bipedal robot Cassie (see Fig. 1) in both simulation and experiment [1], achieving the hopping on Cassie with ground clearance of ~ 7 inches and air-time of $\sim 0.423s$. The ground reaction force and toe-off timing of hopping motions on the robot match closely with these of the spring-mass model. This further indicates a faithful construction and embedding of the reduced-order model onto the full order model of the robot.

*This work is supported by NSF grant NRI-1526519.

The authors are with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91125 xxiong@caltech.edu, ames@caltech.edu

II. ROBOT MODEL

The Cassie-series robot from Agility Robotics [14] is a full 3D bipedal robot that is designed to be agile and robust. Like its predecessor ATRIAS [9], Cassie is designed with concentrated mass at its pelvis and lightweight legs with leaf springs and closed kinematic chains. The mechanical design, thus, embodies the SLIP model [2]. From the perspective of model-based control, the compliant closed chain on each leg can, however, create additional complexities. Therefore, rigid model [4] or overly simplified model [9] is oftentimes applied. Here, we present the full body dynamics model with justifiable simplifications.

As shown in Fig. 1, Cassie has five motor joints (with the axis of rotation shown in red) on each leg, three of which locate at the hip and the other two are the knee and toe pitch. Fig. 2(a) and (b) provide a close look at the leg kinematics and the abstract model, respectively. We model the shin and heel springs as torsion springs at the corresponding deflection axes. Therefore the spring torques are:

$$\tau_{\text{shin/heel}} = k_{\text{shin/heel}}q_{\text{shin/heel}} + d_{\text{shin/heel}}\dot{q}_{\text{shin/heel}}, \quad (1)$$

where $k_{\text{shin/heel}}$, $d_{\text{shin/heel}}$ are the stiffness and damping, provided by the manufacturer [14]. Since the achilles rod is very lightweight, we ignore the achilles rod and replace it by setting a holonomic constraint h_{rod} on the distance between the connectors (one locates on the inner side of hip joint, the other locates at the end of the heel spring). The plantar rod is also removed and the actuation is applied to the toe pitch directly thanks to the parallel linkage design. These two simplifications removed unnecessary passive joints and associated configuration variables. As a consequence, the configuration of the leg can be described only by five motor joints, two spring joints and a passive tarsus joint. The total number of degrees of freedom of the floating base model is then $n = 8 \times 2 + 6 = 22$. The dynamics can be derived from the Euler-Lagrange equation with holonomic constraints as:

$$M(q)\ddot{q} + H(q, \dot{q}) = Bu + J_s^T \tau_s + J_{h,v}^T F_{h,v}, \quad (2)$$

$$J_{h,v}(q)\ddot{q} + \dot{J}_{h,v}(q)\dot{q} = 0, \quad (3)$$

where $q \in \mathbb{R}^n$, $M(q)$ is the mass matrix, $H(q, \dot{q})$ is the Coriolis, centrifugal and gravitational term, B and $u \in \mathbb{R}^{10}$ are the actuation matrix and the motor torque vector, τ_s and J_s are the spring joint torque vector and the corresponding Jacobian, and $F_{h,v} \in \mathbb{R}^{n_{h,v}}$ and $J_{h,v}$ are the holonomic force vector and the corresponding Jacobian respectively. The subscript v is used to indicate different domains which have different numbers of holonomic constraints. For instance, when the robot has no contact with the ground, $n_{h,v} = 2$ as there are two holonomic constraints on h_{rod} . In case when the feet contact the ground, five additional holonomic constraints are introduced on each foot, hence $n_{h,v} = 12$.

III. SPRING-MASS MODEL

In this section, we derive our spring-mass model from the kinematics of the robot. The compliant components on the leg are characterized as a prismatic spring on the leg in the

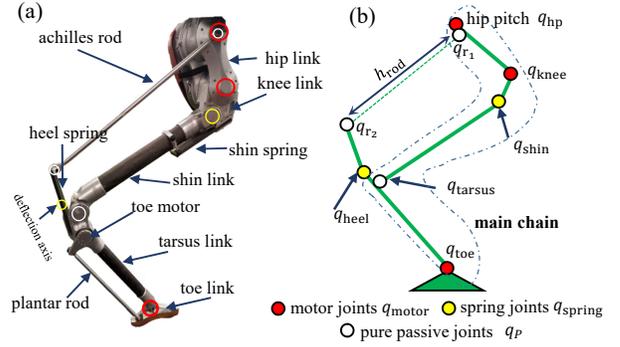


Fig. 2. (a,b) Cassie's leg and its model.

spring-mass model. It is expected that the stiffness of the leg spring changes with different robot configurations, thus we explicitly derive the leg stiffness K_{leg} as a function of joint angles, the analogy of which is the end-effector stiffness for robotic manipulators [10]. With an eye towards the motion planning for the spring-mass model, K_{leg} is approximated by a polynomial function of leg length L . Lastly, we present the trajectory optimization via direct collocation for the spring-mass model.

A. Leg Stiffness and Leg Length

The leg stiffness K_{leg} is the resistance of the leg to external forces. The complementary concept is called leg compliance $C_{\text{leg}} = K_{\text{leg}}^{-1}$. When the leg is under external load at the foot, the leg deforms due to compliant elements in the leg. Assuming that we only consider the transitional deformations, the external force can be calculated by,

$$F_{\text{ext}} = K_{\text{leg}}\delta, \quad (4)$$

where $F_{\text{ext}} \in \mathbb{R}^3$, $K_{\text{leg}} \in \mathbb{R}^{3 \times 3}$ and $\delta \in \mathbb{R}^3$. Under the assumption that the deformation is small and only happens at the joints, the leg deformation δ can be mapped from joint deformations Δq by the foot Jacobian as,

$$\delta = J\Delta q, \quad (5)$$

where $J \in \mathbb{R}^{3 \times n}$ and $\Delta q \in \mathbb{R}^n$. Let τ denotes the moments at the joints caused by the external load, thus $\tau = J^T F_{\text{ext}}$. If the stiffness at each joint is k_i with $i = 1, \dots, n$, then the joint stiffness matrix is defined as $K_J = \text{diag}(k_1, \dots, k_n)$, and $\tau = K_J \Delta q$. The joint stiffness matrix K_J and leg stiffness K_{leg} are hence related by the joint moments,

$$K_J \Delta q = \tau = J^T F_{\text{ext}} = J^T K_{\text{leg}} \delta = J^T K_{\text{leg}} J \Delta q. \quad (6)$$

Then the leg stiffness can be calculated from the joint stiffness matrix by,

$$K_{\text{leg}}(q) = (J(q)K_J^{-1}J(q)^T)^{-1}. \quad (7)$$

This indicates the leg stiffness is a function of the Jacobian and thus a function of the configuration q . The leg damping $D_{\text{leg}}(q)$ is derived in the same way by dropping the assumption on small deformation at joints.

Now we apply the calculation of leg stiffness and damping on Cassie. Note that the main difference is the closed

kinematic chain inside the leg with pure passive joints and compliant joints. The pure passive joints have no contribution towards leg stiffness, so we need to derive the forward kinematics from active joints, i.e. the spring joints and motor joints. As there are two chains towards the toe, the velocity of the toe relative to the hip can be calculated as,

$$v_{\text{Toe} \leftarrow \text{Hip}} = J_1(q_1)\dot{q}_1 = J_2(q_2)\dot{q}_2, \quad (8)$$

where $q_1 = [q_{\text{hip}}; q_{\text{knee}}; q_{\text{shin}}; q_{\text{tarsus}}; q_{\text{toe}}]$ and $q_2 = [q_{r_2}; q_{r_1}; q_{\text{heel}}; q_{\text{toe}}]$ (see Fig. 2). Eq. (8) can be rewritten as,

$$\underbrace{\begin{bmatrix} J_1(q_1) & -J_2(q_2) \end{bmatrix}}_E \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} = 0. \quad (9)$$

Then we can rearrange the matrix E and group q_1, q_2 into active joints $q_A = \{q_{\text{spring}}, q_{\text{motor}}\}$ and pure passive joints q_P . Eq. (8) becomes,

$$\begin{bmatrix} E_A(q) & E_P(q) \end{bmatrix} \begin{bmatrix} \dot{q}_A \\ \dot{q}_P \end{bmatrix} = 0. \quad (10)$$

Then the passive joint velocity $\dot{q}_P = -E_P^{-1}E_A\dot{q}_A$. As there is only one passive joint, the tarsus, on the main kinematic chain, we can find $\dot{q}_{\text{tarsus}} = -E_P^{-1}E_A\dot{q}_A$, where $q_A = \{q_{\text{knee}}, q_{\text{shin}}, q_{\text{heel}}, q_{\text{toe}}\}$. Then the forward kinematics and the leg stiffness can be expressed in terms of q_A ,

$$v_{\text{Toe} \leftarrow \text{Hip}} = J_A(q_A)\dot{q}_A, \quad (11)$$

$$K_{\text{leg}}(q_A) = (J_A(q_A)K_A^{-1}J_A(q_A)^T)^{-1}, \quad (12)$$

where $K_A = \text{diag}(\infty, k_{\text{shin}}, k_{\text{heel}}, \infty)$ as we assume the motor joints being rigidly controlled to fixed positions. Assuming the spring joints have small deflections under normal load, $K_{\text{leg}}(q_A)$ can be approximated as $K_{\text{leg}}(q_{\text{knee}}, q_{\text{shin}} = 0, q_{\text{heel}} = 0, q_{\text{toe}})$. q_{toe} has trivial contribution in terms of J_A and K_{leg} . Thus $K_{\text{leg}}(q_A) \approx K_{\text{leg}}(q_{\text{knee}})$.

This naturally inspires a definition of **virtual leg length** $L(q_{\text{knee}})$ to approximate $K_{\text{leg}}(q_{\text{knee}})$ by $K_{\text{leg}}(L)$. The real leg length $L_r(q_{\text{knee}}, q_{\text{shin}}, q_{\text{heel}})$ is defined as the distance between the hip pitch joint and the toe joint, whereas the virtual leg length is the real leg length with zero spring deflections,

$$L(q_{\text{knee}}) = L_r(q_{\text{knee}}, 0, 0), \quad (13)$$

as illustrated in Fig. 3(a). Due to Cassie's specific leg design, the compliance mainly appears in the direction of the leg. As we are interested in vertical hopping, the last element in K_{leg} , denoted by K_{leg}^z , is taken as the stiffness of the leg. Fig. 3(b) shows how K_{leg}^z changes with L at different static stance configurations. We apply a polynomial regression¹ to approximate the function $K_{\text{leg}}^z(L)$. The leg damping is approximated in the same way.

B. The Actuated Spring-Mass Model

As the total mass of Cassie is concentrated above its hip, we model the pelvis as the point mass and its compliance as the spring attached beneath the mass. As the leg length can change with different kinematic configurations, we model the

¹in the form of $K_{\text{leg}}^z(L) = \beta_0 + \beta_1 L + \beta_2 L^2 + \beta_4 L^4$

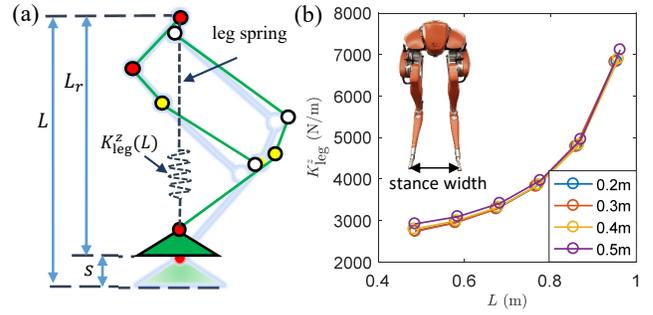


Fig. 3. (a) Illustration of the leg spring from the compliant joints. (b) The vertical leg stiffness v.s virtual leg length for different stance width.

leg as a massless rod between the point mass and the spring. The spring stiffness and damping are obtained from the abovementioned polynomial regressions. As the leg length can change under motor joint actuation, we define the second order derivative of the leg length \ddot{L} as the virtual actuation. Fig. 4(a) illustrates the actuated spring-mass model on the ground, where $x = L_r$ is the vertical position of the mass. The dynamics are,

$$\begin{aligned} \ddot{x} &= \frac{F^s}{m} - g, \\ \ddot{s} &= \ddot{L} - \ddot{x}, \end{aligned}$$

where s is the spring deformation, and $F^s = K(L)s + D(L)\dot{s}$ is the spring force. $K(L)$ and $D(L)$ are the stiffness and damping of the spring, respectively. The kinematic constraints are $s + x = L$ and $\dot{s} + \dot{x} = \dot{L}$.

C. Jumping and Landing Optimization

This actuated spring-mass model naturally enables the traditional trajectory optimization framework to function for planning dynamic tasks such as jumping to a desired height and landing to a static configuration. One can parameterize a time-based actuation profile for \ddot{L} and then find the optimal trajectories for the task while minimizing a cost such as the consumed energy. Here, we apply direct collocation methods [15] to formulate the trajectory optimization problems; the approach is similar to [16]. An even nodal spacing is used for discretizing the trajectory in time. The defect constraint is introduced algebraically by an implicit trapezoidal integration scheme. To enable hopping, we optimize two tasks on the spring-mass model as follows.

Jumping. First, we optimize the spring-mass system to jump to a desired height x_{des} from standing at rest on the ground. When the spring-mass is off the ground, it is in ballistic phase. Therefore, we only discretize the trajectories to the end of ground contact. The task of jumping to the desired height is defined as the equality constraint,

$$x_f + \frac{\dot{x}_f^2}{2g} = x_{\text{des}}, \quad (14)$$

where x_f is the last state of x at standing. Initial states also need to satisfy the initial condition of the system, and the

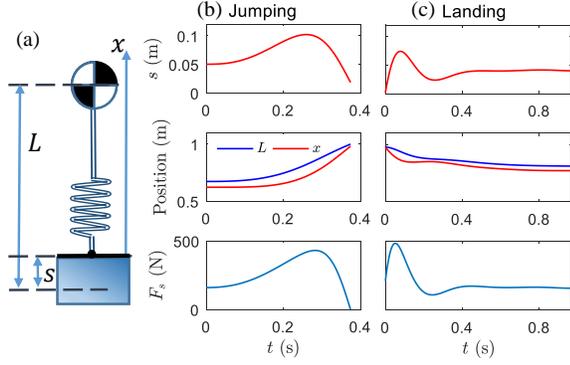


Fig. 4. (a) The spring-mass model. (b,c) Results of the jumping and landing trajectory optimization.

leg length L is constrained to be in the range of kinematic capability of Cassie. Additionally, the ground reaction force, which equals the spring force, must be nonnegative during standing and reaches to zero at the end of ground contact. It is desirable to minimize the virtually consumed energy for this task by defining the cost²,

$$J_{\text{Jumping}} = \int_0^T \ddot{L}(t)^2 dt, \quad (15)$$

where T is the duration of the ground contact phase.

Landing. After the ballistic phase, the spring-mass system will land on the ground. One could optimize the trajectory to enable continuous jumping. In our case, we want the system to come to a desired resting configuration, which is enforced by equality constraints on the final states:

$$\begin{aligned} x_f &= x_{\text{des}}, \\ \dot{L}_f &= \dot{s}_f = 0, \quad \ddot{L}_f = \ddot{s}_f = 0, \\ F_f^s - mg &= 0, \quad \ddot{L}_f = 0. \end{aligned}$$

Again, the spring force has to be nonnegative. More importantly, we enforce that the spring force must be larger than a desired constant value to increase the feasibility of the trajectory on the full dynamics of the robot, i.e., the robot does not leave off the ground again. Also, the spring deflection s has to be smaller than a certain value so that the spring deflections on the robot are within the ranges of the hardware limits. The cost function can be the same as that in jumping to minimize the consumed energy. We add an extra term to minimize the spring oscillation simultaneously, yielding:

$$J_{\text{Landing}} = \int_0^T \ddot{L}(t)^2 + \alpha \dot{s}(t)^2 dt, \quad (16)$$

where α is a weighting coefficient.

As a direct result of the low dimensionality and mild nonlinearity of the system, the optimization is solved in a fast and reliable fashion, typically within 2 seconds. Examples of the optimization results are shown in Fig. 4 (b) and (c).

²The cost is calculated via trapezoidal integration over time discretization. To simplify the notation, we use \int instead of \sum to avoid introducing additional variables for each discretization.

IV. CONTROLLER SYNTHESIS FOR HOPPING

The hopping of the spring-mass system offers important insights into the intrinsic hopping dynamics of the full-order dynamics from which it was derived. In this section, we explain how the trajectory of leg length can be encoded via a control Lyapunov Function based quadratic program (CLF-QP). This yields a nonlinear controller that achieves hopping on the biped Cassie—thus a *priori* nonlinear optimization on the full robot [15] is not required.

A. The Multi-domain Hybrid System of Hopping

The hopping motion inherently is a hybrid dynamical phenomenon with a ground and flight phase. The behavior will, therefore, be described by a hybrid control system of the form:

$$\mathcal{H}\mathcal{C} = (\Gamma, \mathcal{D}, \mathcal{U}, \mathcal{S}, \Delta, FG). \quad (17)$$

Detailed definitions can be found in [17]. We assume that the robot only hops in the sagittal plane and its left and right toe always have the same contact mode. It is also desirable to assume that the front and back part of the toe have the same contact modes for simplification purposes. As illustrated in Fig. 5, three domains are defined for hopping, i.e. $\mathcal{D} = \{\mathcal{D}_J, \mathcal{D}_F, \mathcal{D}_L\}$, where $\{J, F, L\}$ represent *Jumping*, *Flight* and *Landing* respectively. Consequently the directed graph $\Gamma = (V, E)$ is defined by the vertices $V = \{J, F, L\}$ and the edges $E = \{J \rightarrow F, F \rightarrow L\}$.

In *Jumping*, the feet are in contact with the ground. The number of holonomic constraints is $n_{h,J} = 12$. The transition from *Jumping* to *Flight* happens when the ground reaction normal forces cross zero. Thus the domain and associated guard can be defined by,

$$\mathcal{D}_J := \{(q, \dot{q}, u) : h_J(q) = 0, F_z^{\text{Foot}}(q, \dot{q}, u) > 0\}, \quad (18)$$

$$\mathcal{S}_{J \rightarrow F} := \{(q, \dot{q}, u) : h_J(q) = 0, F_z^{\text{Foot}}(q, \dot{q}, u) = 0\}. \quad (19)$$

As there is no impact at the transition from *Jumping* to *Flight*, the reset map is an identity map.

In *Flight*, the feet are off the ground, $n_{h,F} = 2$, and the transition from *Flight* to *Landing* happens when the feet strike the ground. Therefore, we define the domain and corresponding guard by,

$$\mathcal{D}_F := \{(q, \dot{q}, u) : P_z^{\text{Foot}}(q) > 0, F_z^{\text{Foot}}(q, \dot{q}, u) = 0\}, \quad (20)$$

$$\mathcal{S}_{F \rightarrow L} := \{(q, \dot{q}, u) : P_z^{\text{Foot}}(q) = 0, v_z^{\text{Foot}}(q, \dot{q}) < 0\}. \quad (21)$$

We model the impact between the feet and the ground as plastic impact, the reset map of which is detailed in [17].

In *Landing*, the feet are in contact with the ground again, $n_{h,L} = 12$, and the domain \mathcal{D}_L can be defined as the same as \mathcal{D}_J . As we only focus on a single hopping behavior, the system stays in *Landing* after it is reached. There is no need to define its guard.

The continuous dynamics of the system for each domain can be obtained from (2) and (3), wherein the exact forms are specified from the corresponding holonomic constraints. Let $h_{\text{rod}}, h_{\text{Foot}}$ denote the holonomic constraints on the closed kinematic chains and the foot contacts, respectively. Then $h_J = h_L = \{h_{\text{rod}}, h_{\text{Foot}}\}$ and $h_F = \{h_{\text{rod}}\}$.

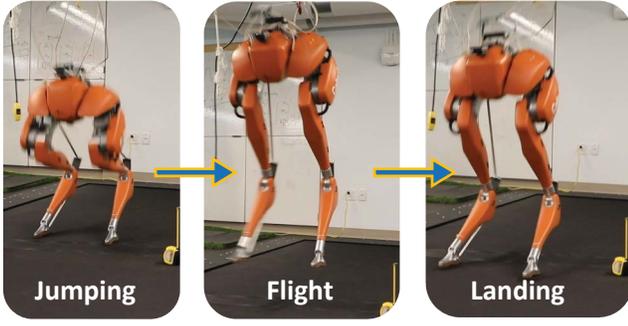


Fig. 5. Discrete domains associated with hopping.

B. CLF-QP

To control the motion of hopping, we suppose that the desired motion is defined by some desired outputs y^d for each domain $v \in V$. For robotic systems, the desired outputs can be outputs with relative degree 1 (\mathcal{RD}_1) and degree 2 (\mathcal{RD}_2) [18]. Let \mathcal{RD}_1 outputs be represented by $y_1 \in \mathbb{R}^{o_1}$ and \mathcal{RD}_2 outputs represented by $y_2 \in \mathbb{R}^{o_2}$. We assume that the desired motion is of time-based trajectories, thus the outputs can be defined as follows [17] [18]:

$$y_1(q, \dot{q}, t) = \dot{y}_1^a(q, \dot{q}) - y_1^d(t), \quad (22)$$

$$y_2(q, t) = y_2^a(q) - y_2^d(t), \quad (23)$$

where the superscript a denotes the actual and d denotes the desired. The objective of the control is to drive $y_1 \rightarrow 0$ and $y_2 \rightarrow 0$. Differentiating y_1 once and y_2 twice yields the affine control system on the output dynamics:

$$\begin{bmatrix} \dot{y}_1 \\ \ddot{y}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathcal{L}_f y_1(q, \dot{q}) - \dot{y}_1^d \\ \mathcal{L}_f^2 y_2(q, \dot{q}) - \ddot{y}_2^d \end{bmatrix}}_{\mathcal{L}_f} + \underbrace{\begin{bmatrix} \mathcal{L}_g y_1(q, \dot{q}) \\ \mathcal{L}_g \mathcal{L}_f y_2(q, \dot{q}) \end{bmatrix}}_{\mathcal{A}} u, \quad (24)$$

where \mathcal{A} is the decoupling matrix, \mathcal{L} denotes the Lie derivative and $u \in \mathbb{R}^m$ is the control input. The dependency on t is dropped from here to simplify the notation. In case when u can be found to satisfy the following equality:

$$\mathcal{A}u = -\mathcal{L}_f + \mu, \quad (25)$$

the output dynamics becomes this linear control system:

$$\dot{\eta} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & I_2 \\ 0 & 0 & 0 \end{bmatrix}}_F \eta + \underbrace{\begin{bmatrix} I_1 & 0 \\ 0 & 0 \\ 0 & I_2 \end{bmatrix}}_G \mu, \quad (26)$$

where $\eta = [y_1, y_2, \dot{y}_2]^T$, I_1 and I_2 are identity matrices with dimension o_1 and o_2 respectively, and μ is the auxiliary control input [18]. One can choose μ to exponentially stabilize the linear system. For example, choosing [11]

$$\mu = \begin{bmatrix} -\epsilon y_1 \\ -2\epsilon \dot{y}_2 - \epsilon^2 y_2 \end{bmatrix} \quad (27)$$

results the linear output dynamics:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \ddot{y}_2 \end{bmatrix} = \begin{bmatrix} -\epsilon I_1 & 0 & 0 \\ 0 & 0 & I_2 \\ 0 & -\epsilon^2 I_2 & -2\epsilon I_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \dot{y}_2 \end{bmatrix}, \quad (28)$$

which is exponentially stable when $\epsilon > 0$. However, such μ does not utilize the natural dynamics of the system and oftentimes may not be realizable on the robotic system if there are stringent physical constraints (e.g. torque bounds) that must be enforced.

The above construction motivates constructing rapidly exponentially stabilizing control Lyapunov functions (RES-CLF) [12] from continuous time algebraic Riccati equations (CARE)³ or continuous time Lyapunov equations (CTLE)⁴ to stabilize the output dynamics exponentially at a chosen rate ϵ . Given a solution $P = P^T > 0$ to CTLE or CARE with $Q = Q^T > 0$, the Lyapunov function is constructed as,

$$V_\epsilon(\eta) = \eta^T \underbrace{I_\epsilon P I_\epsilon}_{P_\epsilon} \eta, \quad (29)$$

where $I_\epsilon = \text{diag}(I_1, \frac{1}{\epsilon} I_2, I_2)$. The goal of exponential stabilizing $\eta \rightarrow 0$ is encoded by the condition:

$$\dot{V}_\epsilon(\eta) \leq -\frac{\gamma}{\epsilon} V_\epsilon(\eta), \quad (30)$$

with some $\gamma > 0$, where,

$$\dot{V}_\epsilon(\eta) = \mathcal{L}_F V_\epsilon(\eta) + \mathcal{L}_G V_\epsilon(\eta) \mu, \quad (31)$$

$$\mathcal{L}_F V_\epsilon(\eta) = \eta^T (F^T P_\epsilon + P_\epsilon F) \eta, \quad (32)$$

$$\mathcal{L}_G V_\epsilon(\eta) = 2\eta^T P_\epsilon G. \quad (33)$$

Eq. (30) and (31) indicate an inequality constraint on μ to achieve exponential stability. This naturally leads to the formulation of quadratic program (QP) to find μ to minimize the quadratic cost $\mu^T \mu$. With (25), the cost and constraint of the QP can be transformed back onto the original control input u by noting that:

$$\mu^T \mu = u^T \mathcal{A}^T \mathcal{A} u + 2\mathcal{L}_f^T \mathcal{A} u + \mathcal{L}_f^T \mathcal{L}_f, \quad (34)$$

and the inequality from (30) and (31) becomes:

$$\mathcal{L}_F V_\epsilon(\eta) + \mathcal{L}_G V_\epsilon(\eta) \mathcal{L}_f + \mathcal{L}_G V_\epsilon(\eta) \mathcal{A} u \leq -\frac{\gamma}{\epsilon} V_\epsilon(\eta). \quad (35)$$

Now the QP can be formulated in terms of solving for u at a current state (q, \dot{q}) as follows,

$$\begin{aligned} u^* &= \underset{u \in \mathbb{R}^m}{\text{argmin}} u^T \mathcal{A}^T \mathcal{A} u + 2\mathcal{L}_f^T \mathcal{A} u \\ \text{s.t.} & \quad \mathcal{A}^{\text{CLF}}(q, \dot{q}) u \leq b^{\text{CLF}}(q, \dot{q}), \end{aligned} \quad (\text{CLF})$$

where,

$$\mathcal{A}^{\text{CLF}}(q, \dot{q}) := \mathcal{L}_G V_\epsilon(q, \dot{q}) \mathcal{A}(q, \dot{q}), \quad (36)$$

$$\begin{aligned} b^{\text{CLF}}(q, \dot{q}) &:= -\frac{\gamma}{\epsilon} V_\epsilon(q, \dot{q}) - \mathcal{L}_F V_\epsilon(q, \dot{q}) \\ &\quad - \mathcal{L}_G V_\epsilon(q, \dot{q}) \mathcal{L}_f(q, \dot{q}). \end{aligned} \quad (37)$$

The result of solving the CLF-QP is a feedback optimal control law to drive the outputs $[y_1^a(q, \dot{q}); y_2^a(q)]$ to follow the desired time based trajectories $[y_1^d(t); y_2^d(t)]$ with exponentially convergence. This formulation also applies when there are only relative degree 2 outputs to be tracked [4]. For applications of using CLF-QP on robotic systems, torque bounds and additional nontrivial constraints can be included in the QP [11].

³ $F^T P + P F - P G G^T P + Q = 0$.

⁴ $F^T P + P F + Q = 0$.

C. Output Definition for Hopping

To apply the CLF-QP formulation on the multi-domain hybrid control system associated with hopping, we define the outputs with reference output trajectories for each domain separately so that the hopping behavior can be enabled. It is important to define two outputs: leg length and centroidal momentum.

Leg Length. The virtual leg length trajectories $L(t)$ from the jumping and landing of the spring-mass model (see Section III) are mainly used as the desired virtual leg length $L^{\text{des}}(t)$ on the full robot. The springs on the full robot are expected to behave similarly to the spring on the spring-mass model when the virtual leg length of the full robot follows $L^{\text{des}}(t)$. In other words, the underactuation of the springs is expected to behave accordingly so that the robot can jump off the ground and land on the ground.

Centroidal Momentum. In *Flight*, the only external force is the gravitational force, thus the robot obeys the conservation of angular momentum about its COM, i.e., the centroidal angular momentum [13]. To keep the control in *Flight* simple, it is desirable to have small centroidal angular momentum when the robot jumps off the ground. The centroidal momentum of a multi-link robotic system can be expressed as [13]:

$$H_G(q, \dot{q}) = A_G(q)\dot{q}, \quad (38)$$

where $H_G \in \mathbb{R}^6$ is the centroidal momentum vector and $A_G(q) \in \mathbb{R}^{6 \times n}$ is the centroidal momentum matrix. As we are mainly concerned with vertical jumping, only the pitch angular momentum is selected:

$$H_{\text{Pitch}}(q, \dot{q}) = A_{\text{Pitch}}(q)\dot{q}. \quad (39)$$

Note that the pitch centroidal momentum is a relative degree 1 output. Differentiating it once yields,

$$\dot{H}_{\text{Pitch}} = \dot{A}_{\text{Pitch}}(q, \dot{q})\dot{q} + A_{\text{Pitch}}(q)\ddot{q}. \quad (40)$$

Now we can specify the outputs for each domain.

1) *Jumping*: When two feet are on the ground, 10 additional holonomic constraints are added to the system. As the robot has 16 degrees of freedom (DoF) except for the tarsus joints and spring joints, we need to apply 6 outputs to guide the motion. As noted above, we select the centroidal pitch momentum as the \mathcal{RD}_1 output. Left and right leg length are used as two \mathcal{RD}_2 outputs to embed the spring-mass jumping dynamics onto the full robot. It is desired to keep the COM position projected onto the center of the support polygon and to avoid yaw motion of the pelvis, which requires $x_{\text{com}} \rightarrow 0$, $y_{\text{com}} \rightarrow 0$ and $\phi_{\text{yaw}} \rightarrow 0$. Therefore, we define the outputs for *Jumping* as,

$$y_1^J(q, \dot{q}) = H_{\text{Pitch}}(q, \dot{q}) - 0, \quad (41)$$

$$y_2^J(q, t) = \begin{bmatrix} L_L(q) \\ L_R(q) \\ x_{\text{com}}(q) \\ y_{\text{com}}(q) \\ \phi_{\text{yaw}}(q) \end{bmatrix} - \begin{bmatrix} L^{\text{des}}(t) \\ L^{\text{des}}(t) \\ 0 \\ 0 \\ 0 \end{bmatrix}. \quad (42)$$

One can interpret the outputs as these on the 6 Dof of the pelvis (the mass in the spring-mass model). The specific leg length on left and right leg constraints the height and the roll of the pelvis. The specific horizontal COM positions constraint the forward and lateral positions of the pelvis. The pitch momentum output constraints the pitch of the pelvis.

2) *Flight*: The robot is off the ground and the 6 Dof of the floating base is in underactuation. Ten outputs are needed and thus we specify all the motor positions as the outputs:

$$y_2^F(q) = q_m - q_m^{\text{des}}. \quad (43)$$

As the centroidal momentum is controlled to be 0 or small in *Jumping*, the robot is not expected to have large whole-body rotation in *Flight*. We simply let the motor positions at the end of *Jumping* be the desired motor positions for *Flight*. The desired toe motor positions are adjusted to keep the toes parallel to the ground.

3) *Landing*: The outputs are defined similarly to the outputs in *Jumping*. The left and right leg length are selected as two outputs to embed the spring-mass landing dynamics onto the robot. It is not necessary to keep the pitch momentum zero for all time during landing, so we remove the \mathcal{RD}_1 output H_{Pitch} . Instead, the pelvis pitch is selected. Thus the outputs are defined as:

$$y_2^L(q, t) = \begin{bmatrix} L_L(q) \\ L_R(q) \\ x_{\text{com}}(q) \\ y_{\text{com}}(q) \\ \phi_{\text{pitch}}(q) \\ \phi_{\text{yaw}}(q) \end{bmatrix} - \begin{bmatrix} L^{\text{des}}(t) \\ L^{\text{des}}(t) \\ x_{\text{com}}^{\text{des}}(t) \\ 0 \\ \phi_{\text{pitch}}^{\text{des}}(t) \\ 0 \end{bmatrix}. \quad (44)$$

The desired pelvis pitch trajectory $\phi_{\text{pitch}}^{\text{des}}(t)$ and desired COM forward trajectory $x_{\text{com}}^{\text{des}}(t)$ are designed smoothly from the post-impact positions to 0.

D. Main Control Law

We apply the CLF-QP based feedback control to the hybrid control system associated with hopping. With an eye towards the constrained optimization formulation [11], we rewrite (2) viewing the torque and force as inputs:

$$M(q)\ddot{q} + \underbrace{H(q, \dot{q}) - J_s^T(q)\tau_s(q)}_{Y(q, \dot{q})} = \underbrace{[B \quad J_{h,v}^T(q)]}_{\bar{B}_v(q)} \underbrace{\begin{bmatrix} u \\ F_{h,v} \end{bmatrix}}_{\bar{u}_v}, \quad (45)$$

where $\bar{u}_v \in \mathbb{R}^{m+n_{h,v}}$ is the augmented control input. The affine control system can thus be defined as:

$$\dot{x} = f(x) + g_v(x)\bar{u}_v, \quad (46)$$

where,

$$f(x) = \begin{bmatrix} \dot{q} \\ -M^{-1}(q)Y(q, \dot{q}) \end{bmatrix}, g_v(x) = \begin{bmatrix} 0 \\ M^{-1}(q)\bar{B}_v(q) \end{bmatrix}. \quad (47)$$

The reason of including $F_{h,v}$ as the control input is to easily incorporate the holonomic constraints as equality constraints and ground reaction force constraints as inequality constraints in the quadratic program.

Holonomic Constraint. Eq. (3) can be rewritten as a function of \bar{u}_v ,

$$A_{h,v}(q, \dot{q})\bar{u}_v = b_{h,v}(q, \dot{q}), \quad (48)$$

$$A_{h,v}(q, \dot{q}) = J_{h,v}(q)M^{-1}(q)\bar{B}_v(q), \quad (49)$$

$$b_{h,v}(q, \dot{q}) = J_{h,v}(q)M^{-1}(q)Y(q, \dot{q}) - \dot{J}_{h,v}(q, \dot{q})\dot{q}. \quad (50)$$

Ground Reaction Force. In *Jumping* and *Landing*, the feet contact the ground. The ground reaction forces have to satisfy the physics constraints such as nonnegative normal forces and non-slipping, formulated by $RF^{\text{Foot}} \leq 0$, where R is a constant matrix. The constraint on \bar{u}_v can be written as

$$A_v^{\text{GRF}}\bar{u}_v \leq b_v^{\text{GRF}}, \quad (51)$$

where $A_{J/L}^{\text{GRF}}\bar{u}_{J/L} = RF^{\text{Foot}}$, $A_F^{\text{GRF}} = 0$, $b_v^{\text{GRF}} = 0$.

Torque Constraints. The motor torque must be within the feasible limits of the robot hardware: $u_{lb} \leq u \leq u_{ub}$.

Optimization-based controller: The resulting QP controller for each domain $v \in V$ is given as follows:

$$\begin{aligned} \bar{u}_v^* = \operatorname{argmin}_{\bar{u}_v \in \mathbb{R}^{m+n_{h,v}}, \delta \in \mathbb{R}} & \bar{u}_v^T \mathcal{A}_v^T \mathcal{A}_v \bar{u}_v + 2(\mathcal{L}_{f,v})^T \mathcal{A}_v \bar{u}_v + p\delta^2, \\ \text{s.t.} & A_v^{\text{CLF}}(q, \dot{q})\bar{u}_v \leq b_v^{\text{CLF}}(q, \dot{q}) + \delta, \quad (\text{CLF}) \\ & A_v^{\text{GRF}}\bar{u}_v \leq b_v^{\text{GRF}}, \quad (\text{GRF}) \\ & u_{lb} \leq u \leq u_{ub}, \quad (\text{Torque}) \\ & A_{h,v}(q, \dot{q})\bar{u}_v = b_{h,v}(q, \dot{q}). \quad (\text{Holonomic}) \end{aligned}$$

To increase the feasibility of the QP, we follow the method in [12] to relax the CLF constraints by introducing δ and penalizing the relaxation by adding $p\delta^2$ in the cost, with some large positive constant p .

V. SIMULATION AND EXPERIMENTAL RESULTS

To validate our proposed control methodology, we first implement this framework in simulation, and then apply the simulation results on the robot in experiment. The simulation starts from an initial static configuration q_0 of the robot. Given a desired apex height, we first optimize the jumping on the spring-mass model from the initial condition corresponding to q_0 . Then the leg length trajectory is used as the desired output in the CLF-QP. The QP is formulated and solved every at 0.5ms using qpOASES [19]. The dynamics is numerically integrated using MATLAB's ode113 function. At the transition from *Flight* to *Landing*, the post-impact state decide the initial condition for landing optimization on the spring-mass. The leg length trajectory of the landing planning is then used in the CLF-QP for controlling the landing of the robot.

The desired and actual output trajectories are shown in Fig. 6 (a) and (b). The simulation indicates fast convergence of output dynamics. Fig. 6 (c) shows the comparison on the system energies. Since the robot is not a point mass exactly, the kinetic energy and potential energy differ slightly from these of the spring-mass. Fig. 6 (d) and (f) illustrate the spring torques at joints and the ground reaction force (GRF) respectively. Note that the GRF of the robot model align closely with the GRF of the spring-mass model. More

importantly, the spring joint forces on the robot show the same profile as the GRF, which equals to the spring force in the spring-mass. This ratifies our hypothesis that the spring of the robot behaves similar to the spring in the spring-mass.

The QP is not yet implemented on the hardware but this is a subject of future work. To validate our method experimentally, we extract the motor joint positions from the simulation and apply position tracking by a PD+feedforward controller on the hardware at 2kHz. The feedforward term is the motor torque from the simulation. Under this setting, the QP is viewed to perform joint trajectory generation. Fig. 7 (a) (b) and (c) show the experiment results of a hopping motion with ground clearance of ~ 7 inches. Due to the model inaccuracy of the physical robot, the spring deflection difference is ineligious and the robot jumps forward consequently. Future work will try to address this issue by utilizing feedback. Experiment videos can be found at [1].

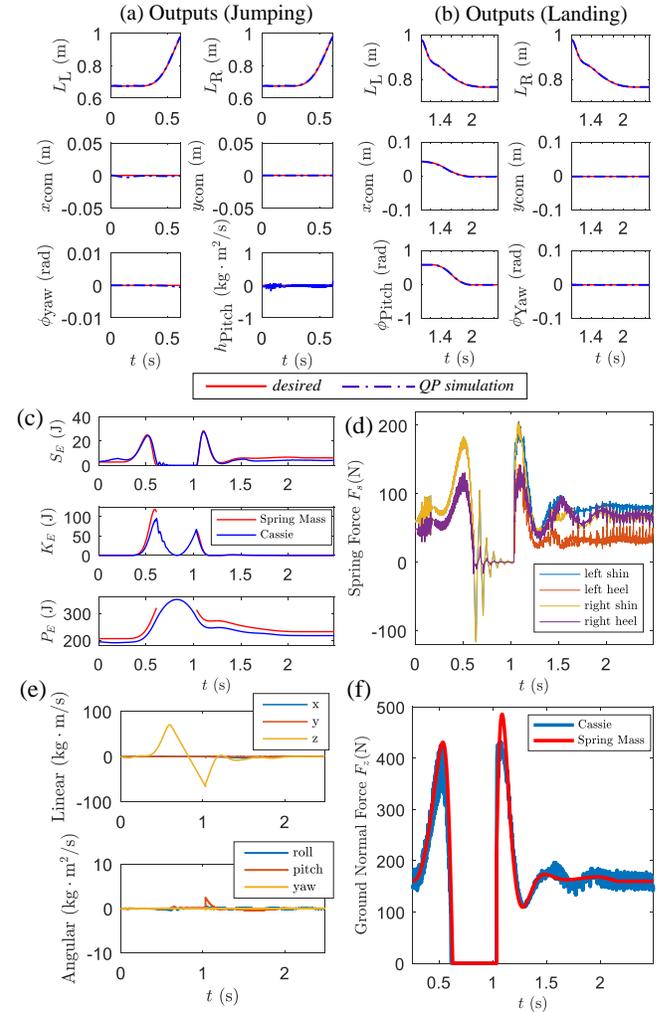


Fig. 6. Simulation results. (a,b) Outputs of jumping and landing. (c) Spring energy S_E , kinetic energy K_E and gravitational potential energy P_E during hopping. (d) The spring torques of Cassie during hopping. (e) Centroidal momentum of Cassie during hopping. (f) Comparison on the ground reaction normal force of the hopping of Cassie vs that of the spring-mass model.

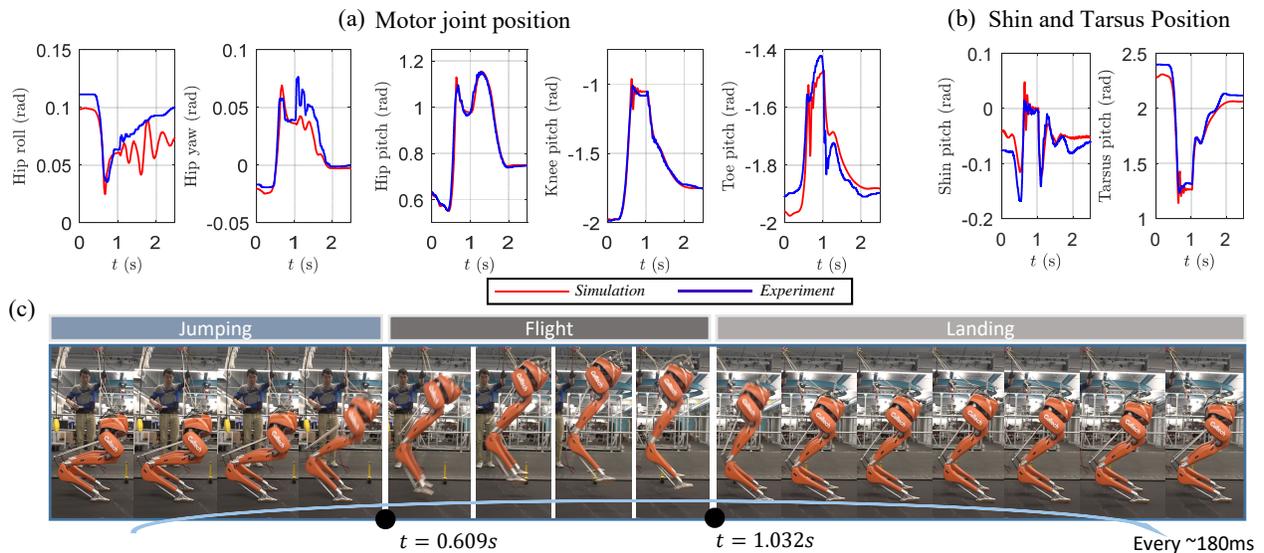


Fig. 7. Experiment results. (a) Comparison of motor joint positions on the left leg in simulation vs these in experiment. (b) Comparison of shin and tarsus joints on the left leg in simulation vs these in experiment. (c) Snapshot of the hopping of Cassie.

VI. CONCLUSION AND FUTURE WORK

This paper presented the planning and controller synthesis to achieve hopping of the bipedal robot Cassie via reduced-order model embedding. The spring-mass model is faithfully established from the mechanical (kinematic and compliant) properties of the robot. The planned trajectories of leg length and additional outputs are rigorously defined in the context of a CLF-QP formulation to facilitate the feedback control of hopping. The realized hopping on Cassie validates the proposed approach.

The future work will be devoted to extending the approach to create different dynamic behaviors on bipedal robots, likely including reduced-order modeling in two or three dimensions for walking and running behaviors. From a theoretical viewpoint, we want to establish stability conditions on how the reduced-order model quantitatively predicts dynamics behaviors of the full-order system. The hope is that this will inform rigorous controller synthesis for complex robots via simple models.

REFERENCES

- [1] <https://youtu.be/5s19IWqN4-c>.
- [2] J. Rummel, Y. Blum, H. M. Maus, C. Rode, and A. Seyfarth, "Stable and robust walking with compliant legs," in *Robotics and automation (ICRA), 2010 IEEE international conference on*, pp. 5250–5255.
- [3] X. Xiong, A. D. Ames, and D. I. Goldman, "A stability region criterion for flat-footed bipedal walking on deformable granular terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada*, pp. 4552–4559.
- [4] A. Hereid, M. J. Powell, and A. D. Ames, "Embedding of slip dynamics on underactuated bipedal robots through multi-objective quadratic program based control," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, pp. 2950–2957.
- [5] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.
- [6] I. Poulakakis and J. Grizzle, "The spring loaded inverted pendulum as the hybrid zero dynamics of an asymmetric hopper," *IEEE Transactions on Automatic Control*, vol. 54, no. 8, pp. 1779–1793, 2009.
- [7] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Robotics and Automation, IEEE International Conference on*, vol. 2, 2003, pp. 1620–1626.
- [8] J. Pratt, T. Koolen, T. De Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus, "Capturability-based analysis and control of legged locomotion, part 2: Application to m2v2, a lower-body humanoid," *The International Journal of Robotics Research*, vol. 31, no. 10, pp. 1117–1133, 2012.
- [9] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Sprowitz, A. Abate, and J. Hurst, "Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1497–1521, 2016.
- [10] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer Science & Business Media, 2008.
- [11] A. D. Ames and M. Powell, "Towards the unification of locomotion and manipulation through control lyapunov functions and quadratic programs," in *Control of Cyber-Physical Systems*. Springer, 2013, pp. 219–240.
- [12] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.
- [13] D. E. Orin and A. Goswami, "Centroidal momentum matrix of a humanoid robot: Structure and properties," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 653–659.
- [14] <http://www.agilityrobotics.com>.
- [15] A. Hereid, E. A. Cousineau, C. M. Hubicki, and A. D. Ames, "3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1447–1454.
- [16] C. M. Hubicki, J. J. Aguilar, D. I. Goldman, and A. D. Ames, "Tractable terrain-aware motion planning on granular media: an impulsive jumping study," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 3887–3892.
- [17] A. D. Ames, "Human-inspired control of bipedal walking robots," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1115–1130, 2014.
- [18] H. K. Khalil, "Nonlinear systems," *Prentice-Hall, New Jersey*, vol. 2, no. 5, pp. 5–1, 1996.
- [19] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.