Impact-Invariant Running on the Cassie Bipedal Robot

William Yang and Michael Posa

***Abstract*— Impact-invariant control is a general framework for adapting model-based controllers for robots undergoing impacts. The framework projects the tracking objectives into a subspace that is invariant to impact forces, thus resulting in a controller that is robust to uncertainties in the impact event while minimally sacrificing control authority. In this work, we apply impact-invariant control to a SLIP-inspired running controller for the bipedal robot Cassie. This, to our knowledge is the first example of a model-based running controller demonstrated on hardware for Cassie. We detail our controller framework and the effect of the impact-invariant projection on the stability of the controller.**
Paper Type – Original Work

I. Introduction

There has recently been considerable progress on walking controllers for bipedal robots demonstrating full use of the passive dynamics [1] and impressive robustness [2]. While walking is a suitable locomotive gait in most scenarios, exploration of more dynamic gaits is necessary to unlock the full locomotive potential of bipedal robots.

Running not only enables movement at higher speeds, but also generally results in more efficient motions at those speeds [3]. While running has been demonstrated on many bipedal robotic platforms including MABEL [4], ATRIAS [5], as well as Atlas from Boston Dynamics, running has not yet been demonstrated on the 3D bipedal robot Cassie except through reinforcement learning (RL) [6]. Cassie presents a particularly challenging control problem when the motion includes an aerial phase, because its relatively low inertia pelvis coupled with a high degree of underactuation makes it difficult to stabilize.

Model-based controllers should in principle be well suited for designing running motions because they have the ability to directly incorporate known model details and specify desired behaviors in an intuitive manner. However, model-based controllers are often brittle to model uncertainties as they assume perfect knowledge of the robot dynamics. One particularly sensitive source of model uncertainty is at the impact event, where the robot undergoes rapid changes to its velocity over a short period of time. Our recent work [9] has shown that projecting the robot state to an impact-invariant subspace enables controllers to avoid providing erroneous feedback commands caused by sensitivity to the impact event while minimally sacrificing control authority. In that previous work, we demonstrate the benefits of the projection on improving the tracking performance on walking and jumping motions.

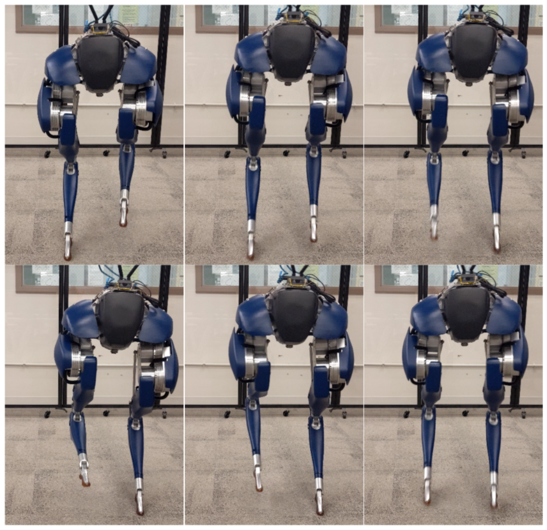
Fig. 1. Gait tiles of the bipedal robot Cassie executing a running gait in a controlled laboratory setting.

In this work, we incorporate the impact-invariant projection into an inverse dynamics running controller with a SLIP-inspired pelvis trajectory and Raibert footstep regulator as the footstep planner. In doing so, we showcase the benefits of the impact-invariant projection on a more dynamic gait.

The contributions of this paper are as follows:

- Demonstration of the first model-based running controller for the bipedal robot Cassie
- Analysis of the effect of impact-invariant projection on a dynamic running gait

We describe the robot model and give an overview of impact-invariant control in II. We discuss the model-based controller formulation, including the generation of the reference trajectories, in III. The running controller is then implemented on the physical robot and the effects of the projection are shown in IV. Finally, comments about the successes and limitations of this controller are discussed in V.

II. Background

A. Model Dynamics

Cassie is a 3D bipedal robot with built-in compliance located in the knee and ankle springs. We model Cassie's dynamics using conventional Lagrangian rigid body dynamics. The robot's state $x \in \mathbb{R}^{45} = [q; v]$ contains the robot's positions $q \in \mathbb{R}^{23}$ and velocities $v \in \mathbb{R}^{22}$ and is expressed in generalized floating-base coordinates. The dynamics are

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1845298

The authors are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA {yangwill, posa}@seas.upenn.edu

derived using the Euler-Lagrange equation and expressed in the manipulator equation:

$$M(q)\ddot{q} + C(q, \dot{q}) + g(q) = Bu + J_\lambda(q)^T \lambda, \tag{1}$$

where $M \in \mathbb{R}^{22\times22}$ is the mass matrix, C and g are the Coriolis and gravitational forces respectively, B is the actuator matrix that maps the vector of actuator inputs $u \in \mathbb{R}^{10}$ to generalized forces, and J_λ and $\lambda \in \mathbb{R}^{14}$ are the Jacobian of the holonomic constraints and corresponding constraint forces respectively. The Jacobian of the holonomic constraints includes the Jacobian for the loop closure constraint of the knee linkages present on each leg J_h as well as the Jacobian for the contact constraint of the current stance leg J_c.

B. Impact Invariant Control

Controllers for legged robots are often sensitive to impact events, due to the presence of large contact forces resolving in a relatively short period of time leading to highly uncertain velocity signals. Prior solutions often used heuristics-based methods such as blending or reducing feedback gains to avoid overreacting to these velocities. While these methods can be effective at reducing sensitivity, they sacrifice control authority to do so.

Impact-invariant control [9] is an extension applied to model-based controllers that enables robustness to uncertainty in both the timing and duration of impact events, while minimally sacrificing control authority. Prior work on walking and jumping has shown it improves tracking near impact.

The extension revolves around a projection of the robot state into the nullspace of $M^{-1}J_\lambda^T$, which maps contact forces into generalized accelerations. The basis $P(q) \in \mathbb{R}^{(n-c)\times n}$ for the impact-invariant space is defined as

$$P(\dot{q} - \dot{q}^-) = 0 = PM^{-1}J_\lambda^T \Lambda, \tag{2}$$

such that the projected state is constant to any unknown contact force or even absent contact force.

This projection can be applied directly to the robot state. Alternatively, for more complex inverse dynamics based controllers, the projection can be applied directly to the tracking objectives, provided they are a function of the robot state. This is accomplished by solving a least squares problem to find a correction that projects the output velocities discussed later in III-B.

III. Controller Formulation

A. Controller Formulation

We use an operational space controller (OSC) to track desired outputs that produce the running motion. An OSC is an inverse dynamics controller that tracks a set of task or output space accelerations by solving for dynamically consistent inputs, ground reaction forces, and generalized accelerations [10] [11].

Our formulation, described in detail in our previous work [9], is summarized here. For an output position $y(q) = \phi(q)$ and corresponding output velocity $\dot{y} = J_y(q)\dot{q}$, where $J_y(q) = \frac{\partial \phi}{\partial q}$, the commanded output accelerations $\ddot{y}_{cmd}$ are calculated from the feedforward reference accelerations $\ddot{y}_{des}$ with PD regulation:

$$\ddot{y}_{cmd} = \ddot{y}_{des} + K_p(y_{des} - y) - K_d(\dot{y}_{des} - \dot{y}) \tag{3}$$

The objective of the OSC is then to produce dynamically feasible output accelerations $\ddot{y}$ given by:

$$\ddot{y} = \dot{J}_y\dot{q} + J_y\ddot{q},$$

such that the instantaneous output accelerations of the robot are as close to the commanded output accelerations as possible. This controller objective is written as the following optimization problem:

$$\begin{aligned} \min_{u,\lambda,\ddot{q}} \quad & J_\theta(u, \lambda, \ddot{q}) && (4)\\ \text{subject to:} \quad & \text{Dynamic Constraint} && (5)\\ & \text{Holonomic Constraints} && (6)\\ & \text{Friction Cone Constraints,} && (7) \end{aligned}$$

where J_θ is a combination of the instantaneous tracking error and regularization terms

$$\begin{aligned} J_\theta(u, \lambda, \ddot{q}) = \sum_i^N (\ddot{y}_i - \ddot{y}_{i_{cmd}})^T W_i (\ddot{y}_i - \ddot{y}_{i_{cmd}}) \\ + \text{ regularization terms.} \end{aligned} \tag{8}$$

This optimization problem can be formulated as a quadratic program (QP) and be solved efficiently.

B. Projecting General Outputs to the Impact Invariant Space

To project general outputs to an impact-invariant space, we solve the following optimization problem:

$$\begin{aligned} \min_{\lambda} \quad & \left\| \dot{y}_{des} - J_y(\dot{q} + M^{-1}J_\lambda^T\lambda) \right\|_2 && (9)\\ \text{subject to:} \quad & J_h(\dot{q} + M^{-1}J_\lambda^T\lambda) = 0 && (10) \end{aligned}$$

This applies a correction to the generalized velocities $\dot{q}$ that minimizes the tracking error in the output velocities, under the condition that the correction lies within the set of feasible velocities that could result from a contact impulse λ. With some manipulation, this can be formulated as a least squares problem and the optimal λ can be solved for implicitly with the Moore-Penrose pseudo-inverse denoted by $(\cdot)^\dagger$. The projected output velocity error, $\dot{y}_{proj}$, can then be found as:

$$\dot{y}_{proj} = \dot{y}_{des} - J_y\dot{q} - J_yM^{-1}J_\lambda^T\lambda, \tag{11}$$

This projected error $\dot{y}_{proj}$ is then used in place of the original output velocity error $\dot{y}_{des} - \dot{y}$ in (3).

C. Finite State Machine

We use a time-based finite state machine (FSM) to specify the active contact mode and generate the clock signal for the reference trajectories. We use a fixed frequency finite state machine in our implementation; however, variable step timings can be prescribed in this framework as well. The set of finite states is {LS, LF, RS, RF}, L and R correspond to the left and right legs respectively and S and F correspond

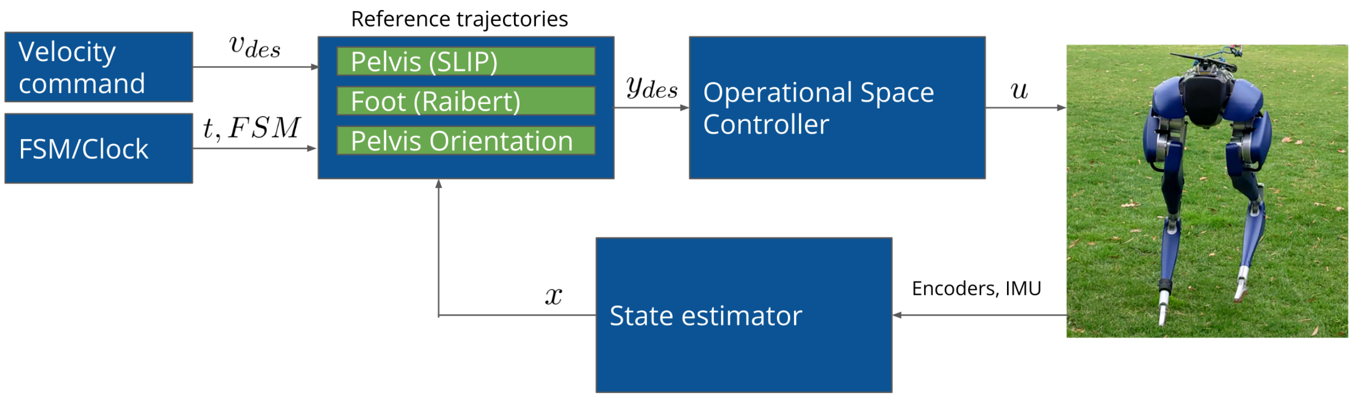

Fig. 2. Key elements of the running controller diagram.

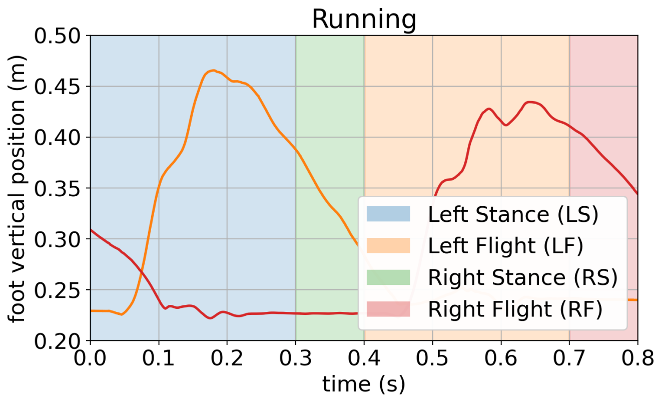

Fig. 3. Foot positions in the associated contact modes for a snapshot of an experiment on the physical robot. Note, due to state estimator drift, the vertical position of the foot when on the ground is not at 0.

to Stance and Flight. We distinguish between the two air phases {LF, RF} even though they are governed by the same dynamics in order to prescribe of different tracking priorities for the different legs.

This FSM is also used to determine when the impact-invariant projection is active. In a small window of duration T before and after the nominal impact is anticipated, i.e. during the transition from LF to RS and from RF to LS, we blend in the correction using a continuous scalar function $\alpha(t)$ to avoid introducing additional discontinuities. The scalar function is given by:

$$\alpha(t) = 1 - \exp(\frac{-(t - t_{switch} + T)}{\tau}), \quad (12)$$

where t_{switch} is the nominal impact time according to the FSM, and τ is the time constant.

D. Tracking Objectives

The vector of tracking objectives are the virtual SLIP length $L_{SLIP} \in \mathbb{R}$, the feet positions $y_L, y_R \in \mathbb{R}^3$, the pelvis orientation $\psi \in SO(3)$, the foot angles $\phi_L, \phi_R \in \mathbb{R}$, and finally the hip yaw joints $\beta_L, \beta_R \in \mathbb{R}$. L_{SLIP} is defined as the distance between the pelvis and current stance foot and the foot positions y_L, y_R are defined relative to the pelvis, expressed in the world frame.

During left stance (LS), the active vector of tracking objectives is $[L_{SLIP}, y_R, \psi, \phi_R, \beta_R]^T \in \mathbb{R}^9$. For right stance (RS), the tracking objectives are the same, just flipped for the other leg.

During the aerial modes LF, RF, the active tracking objectives are $[y_L, y_R, \psi, \beta_L, \beta_R, \phi_L, \phi_R] \in \mathbb{R}^{13}$. Note, the dimension of the tracking objectives in flight, 13, is greater than the total degrees of actuation, 10. Therefore, during flight, the control formulation is over specified and perfect tracking cannot be achieved. We chose to leave the problem overspecified as opposed to leaving out either the pelvis orientation ψ or one of the foot positions y because we found that trading off multiple tracking objectives led to better performance on hardware.

E. Reference Trajectory Generation

The OSC supports the tracking of any differentiable reference trajectory. For example, in prior work, a jumping controller was constructed using target trajectories computed through offline trajectory optimization. Similar methods of computing offline reference trajectories with trajectory optimization in order to construct a gait library were attempted in the process of developing this running controller.

Ultimately, we chose to use simple well-known models (SLIP) and heuristics (Raibert stepping [12]) to generate the reference trajectories. The trajectories generated with these simple heuristics were trivial to compute and resulted in stable and natural looking running motions.

1) SLIP-inspired pelvis trajectory: Inspired by the extensive literature on SLIP, we use a SLIP-like reference for the pelvis motion during stance {LS, RS}. We achieve this by anchoring the pelvis to a damped spring model using the OSC gains and setting the spring rest length L, defined as the distance between the pelvis and the current stance foot, as a constant target output.

$$\ddot{y}_{SLIP,cmd} = K_p(L - y_{SLIP}) + K_d(\dot{y}_{SLIP}) \quad (13)$$

Note, this is not formulated exactly SLIP, which has dynamics:

$$\ddot{y}_{SLIP} = \frac{k}{m}(L - y_{SLIP}) + r\dot{\gamma}^2 - g, \quad (14)$$

where L is the rest length, k/m is the spring constant, and γ is the angle of the leg relative to the ground. In particular the radial and gravitational components are missing from our formulation. We choose not to track the SLIP dynamics because that would result in two sets of gains (one for SLIP dynamics, and one for OSC to converge to the reference trajectory)

2) Footstep trajectories: Regulating the global velocity is achieved through foot placement. While there are possible variations for choosing where to place the foot [13] [2], the basic principle behind all the stepping strategies is stepping in the direction that you are falling. We choose to regulate the running velocity by planning footsteps with the widely recognized Raibert footstep control law [12]:

$$y_{ft} := \begin{bmatrix} y_{ft,x} \\ y_{ft,y} \\ y_{ft,z} \end{bmatrix} = \begin{bmatrix} K_x(v_{des,x} - v_x) \\ K_y(v_{des,y} - v_y) \\ -L \end{bmatrix}, \tag{15}$$

where K are the Raibert stepping gains, v_{des} are the desired velocities as commanded by the operator, and v is the current velocity computed by the state estimator. x and y in this context denote the sagittal and lateral directions respectively. y_{ft} then defines the target footstep location relative to the pelvis.

With the end footstep location specified, we can generate a trajectory for the swing foot given its initial position at liftoff to the final desired location. We specify all the reference trajectories as piecewise cubic polynomials, so with the additional degrees of freedom we add a waypoint so that the trajectory roughly resembles the swing leg retraction policy observed in both numerical optimization [14] [15] and biology [16].

3) Turning: The desired pelvis orientation is specified as the identity quaternion in the absence of operator commands. To implement turning, we simply offset the desired orientation from the current orientation in the direction of the command.

IV. Results

The optimization problem (4) is solved using OSQP. State estimation is implemented using a contact-aided invariant EKF [17], where contact is detected using the deflection of the physical leaf springs. The state estimator returns state feedback information at 2 kHz, whereas the OSC runs at 1 kHz. The controller gains were first tuned in the Drake [18] simulator and additional tuning was later done on hardware. In our preliminary formulation, we are able to successfully achieve stable running, albeit in a limited range of velocity commands. The controller gains used on hardware are given in Table I.

TABLE I
Relevant controller parameters

Controller Parameters	Value
Stance Duration (s)	0.30
Flight Duration (s)	0.10
Raibert Stepping Gains (K_x, K_y)	0.3, 0.6
SLIP (K_p, K_d)	65, 5
Projection duration (s)	0.05

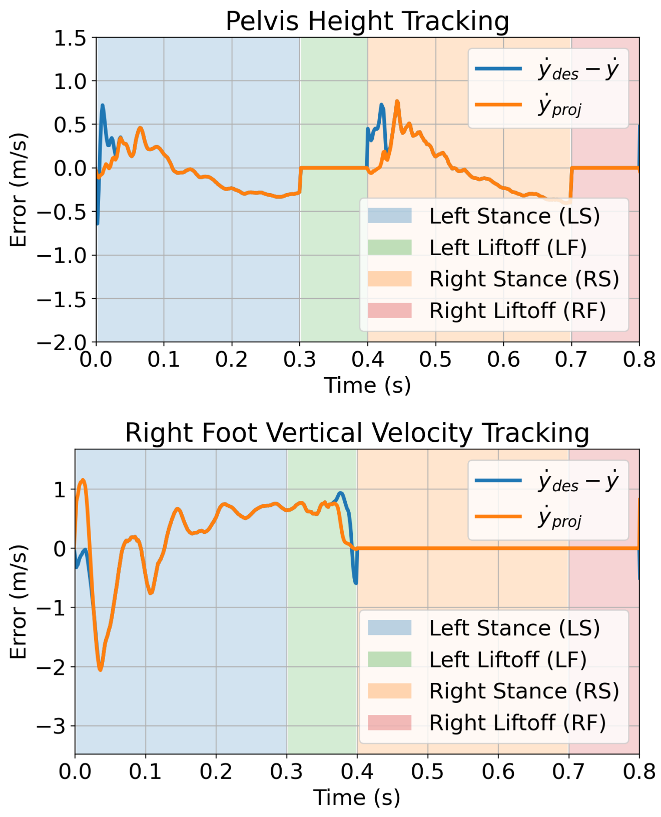

Fig. 4. A feedback controller that reacts the feedback error as typically used (blue) would react erroneously to the spikes in error near the impact events and apply efforts that may harm tracking performance. When the error is projected using the impact-invariant projection (yellow), the sensitivity to the impact events is greatly reduced.

A. Feedback signal with impact-invariant projection

To illustrate the benefits of the impact-invariant projection, we plot the velocity feedback terms in Figure 4. Due to uncertainty in the impact timing, duration, and magnitude, the error $\dot{y}_{des} - \dot{y}$ often spikes. These errors often dissipate quickly, and thus reacting only to the projected error allows for much more stable feedback.

V. Discussion

We successfully implemented running on the 3D bipedal robot Cassie, a task thus far only accomplished using reinforcement learning methods.

Our controller formulation was surprisingly simple, using methods and heuristics developed or observed decades prior. These results are still preliminary, with only a few iterations of hardware tuning, thus our current controller is only stable in a small range of commanded velocities. Oscillations from the physical springs are a obstacle to the tracking performance, and precise velocity regulation has not been achieved. However, we believe additional tuning can resolve these issues. Future work includes variable step timings and constraints on the footstep locations.

References

[1] J. Reher and A. D. Ames, "Inverse dynamics control of compliant hybrid zero dynamic walking," *arXiv preprint arXiv:2010.09047*, 2020.

[2] Y. Gong and J. Grizzle, "Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller," *arXiv preprint arXiv:2008.10763*, 2020.

[3] W. Xi, Y. Yesilevskiy, and C. D. Remy, "Selecting gaits for economical locomotion of legged robots," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1140–1154, 2016.

[4] K. Sreenath, H.-W. Park, I. Poulakakis, and J. W. Grizzle, "Embedding active force control within the compliant hybrid zero dynamics to achieve stable, fast running on mabel," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 324–345, 2013.

[5] C. Hubicki, A. Abate, P. Clary, S. Rezazadeh, M. Jones, A. Peekema, J. Van Why, R. Domres, A. Wu, W. Martin, *et al.*, "Walking and running with passive compliance: Lessons from engineering: A live demonstration of the atrias biped," *IEEE Robotics & Automation Magazine*, vol. 25, no. 3, pp. 23–39, 2018.

[6] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7309–7315.

[7] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," *arXiv preprint arXiv:2105.08328*, 2021.

[8] H. Duan, A. Malik, J. Dao, A. Saxena, K. Green, J. Siekmann, A. Fern, and J. Hurst, "Sim-to-real learning of footstep-constrained bipedal dynamic walking," *arXiv preprint arXiv:2203.07589*, 2022.

[9] W. Yang and M. Posa, "Impact invariant control with applications to bipedal locomotion," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5151–5158.

[10] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 3103–3109.

[11] L. Sentis and O. Khatib, "Control of free-floating humanoid robots through task prioritization," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 1718–1723.

[12] M. H. Raibert, H. B. Brown Jr, and M. Chepponis, "Experiments in balance with a 3d one-legged hopping machine," *The International Journal of Robotics Research*, vol. 3, no. 2, pp. 75–92, 1984.

[13] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2. IEEE, 2003, pp. 1620–1626.

[14] H. Dai and R. Tedrake, "Optimizing robust limit cycles for legged locomotion on unknown terrain," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 1207–1213.

[15] A. Seyfarth, H. Geyer, and H. Herr, "Swing-leg retraction: a simple control model for stable running," *Journal of Experimental Biology*, vol. 206, no. 15, pp. 2547–2555, 2003.

[16] M. A. Daley and A. A. Biewener, "Running over rough terrain reveals limb control for intrinsic stability," *Proceedings of the National Academy of Sciences*, vol. 103, no. 42, pp. 15 681–15 686, 2006.

[17] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[18] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu