Modeling a Monopod Hopper with Parallel Actuation

Benjamin Bokser[1]

***Abstract*— Maximal coordinate dynamics with variational integration is a promising method which can be applied to dynamically hopping robots. The dynamics and simulation of a monopod hopping robot with parallel leg actuation are presented and tested for both minimal and maximal coordinates, with the potential for in-depth quantitative comparison in the future.**

I. INTRODUCTION

Developing highly dynamic robots requires a balance between high-fidelity and high-efficiency dynamics models and integration methods for both simulation and control. The choice of methodology is an ongoing research question.

The majority of current state-of-the-art dynamic robots use minimal coordinate dynamics for control. For example, the MIT Mini Cheetah [1], Cassie [2], and Hume [3] have made extensive use of whole-body operational space control.

Simulation engines currently used in robotics research prioritize computational efficiency over accuracy. For example, PyBullet [4] utilizes a time-stepping method with forward Euler integration, which artificially injects energy into dynamical systems [5]. MuJoCo [6] implements contact smoothing with classic Runge-Kutta (RK4) integration. While convenient for machine learning, smooth contact models are not accurate to real-world physics [7], [5]. Both of these examples operate in minimal coordinates.

In the past, maximal coordinate methods have seen little use due to greater computational cost and accuracy issues such as constraint drift [8]. However, recent innovations made by Brudigam and Manchester [9] solve many of these issues through a special combination of maximal coordinates and variational integration. This method promises improved fidelity at a reasonable computational cost without constraint drift and can be used for both simulation and control. However, until now it has not been demonstrated with a real dynamic legged robot. To that end, we present the Fly-Hopper (Fig. 1).

This paper lays down the groundwork for the dynamics and simulation of the Fly-Hopper robot. Section II provides a brief description of the robot system. Section III provides methods for simulation of the robot in minimal coordinates. Section IV explores dynamics and simulation of the robot in maximal coordinates with variational integration. And Section V contains a discussion of the results and future work.

[1]B. Bokser is with the Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213 USA bbokser at andrew.cmu.edu

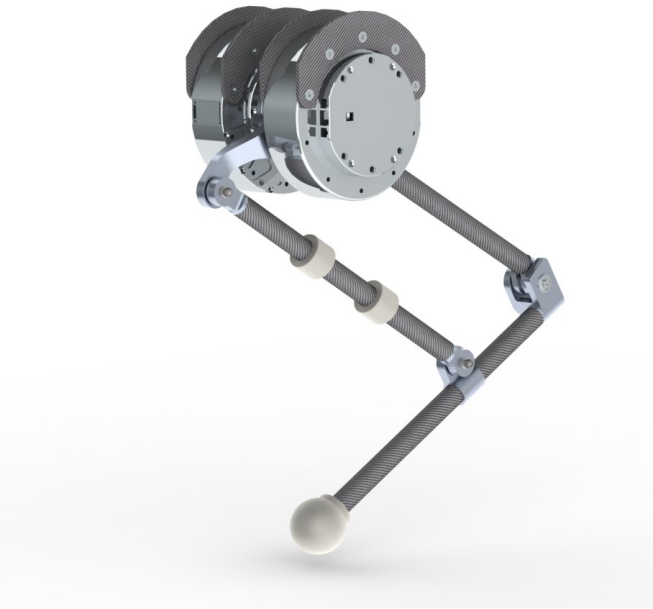

Fig. 1. Rendering of the hopper (moment control system not shown).

II. SYSTEM OVERVIEW

The Fly-Hopper has been designed with the specific intent of performing highly dynamic leaping maneuvers. As such, the design prioritizes maximum flight time per unit mass, mid-air maneuverability, impact mitigation, and force proprioception. Reaction wheel actuators (RWA) (not shown) will be used for both balance and mid-air attitude adjustment.

In its default state, the Fly-Hopper's Center of Mass (CoM) rests approximately 0.3 m off the ground. As a general rule, smaller robots perform better than larger robots in terms of relative jump height and impact mitigation due to significantly reduced mass. However, design requirements for this project bounded the Fly-Hopper's minimum size to future-proof the robot for larger-scale applications. In addition, available electronics, actuators, and other components, as well as design complexity, limit the practical size of the robot.

A. Leg Design

The Fly-Hopper will weigh approximately 6 kg in total. The vast majority of its mass is concentrated in the body, as minimizing distal leg mass provides optimal actuation bandwidth and impact mitigation [10]. Minimizing distal leg

mass requires distal actuation, which narrowed the mechanism design selection of the leg to three main types:

1) Two-actuator linkage-driven or belt-driven serial actuation as seen in [11] and [12]
2) Two-actuator linkage-driven or belt-driven parallel actuation as seen in [13]
3) Single-actuator belt-driven parallel actuation.

For the purposes of quick analysis, PyBullet simulation was performed early in the design process to select the best mechanism type. All three mechanism types were compared on a basis of vertical jump height for a given body mass, and actuator specifications using the actuator model developed in Section II were taken into account. Choice 2 was found to provide the best ratio of jump-height-to-actuator-torque, an important factor due to the torque limitations of quasi-direct drive electric actuators.

For distal actuation, a four-bar linkage was chosen over a belt in order to utilize a parallel spring mechanism, which improves efficiency, impact mitigation, and flight time through energy storage. However, the dynamics of the spring are not investigated in this paper.

To optimize link lengths, a simple iterative analysis of flight-time versus scale was performed using PyBullet, as shown in Fig. 2.

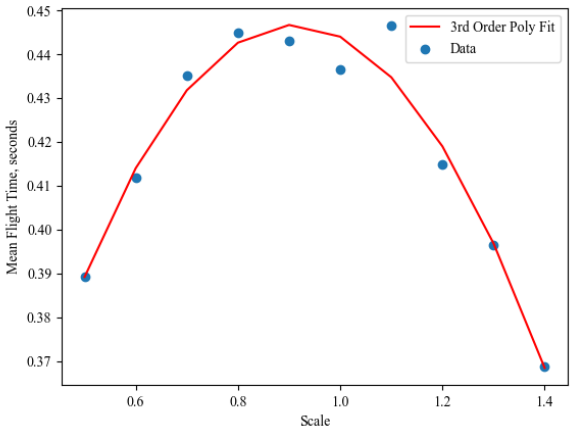

Fig. 2. Iterative analysis of flight time versus scale.

B. Actuator Model

High torque density electric motors with quasi-direct drive (QDD) actuation have taken the forefront in dynamic robotic actuation over the past decade due to their transparency, backdrivability, and force proprioception capabilities [14]. The Fly-Hopper takes full advantage of this design paradigm for leg actuation. Both leg actuators are composed of permanent magnet synchronous outrunner motors with large air gap radii connected to single stage planetary gearing with a 1:7 reduction. Depending on the motor's specifications, an electric motor's torque saturates as a function of speed, and speed saturates as a function of torque. This has been accounted for in PyBullet simulation as described in [15].

$$\tau_k = k_t * i_k \tag{1}$$

$$\tau_{max_k} = -\omega_k(k_t^2)/R + V_{max} * k_t/R \tag{2}$$

Where i is the current input, τ is motor torque, k_t is the torque constant, R is armature resistance, τ_{max_k} is the maximum torque at the current timestep, ω is motor speed in radians, and V_{max} is maximum voltage. The motor torque is then saturated by the maximum torque at each timestep.

$$-\tau_{max_k} \leq \tau_k \leq \tau_{max_k} \tag{3}$$

C. Moment Control System

The Fly-Hopper requires a moment control system for both balance and adjustment of attitude in mid-air. There are two main design options in consideration for this moment control system: 1. reaction wheel actuators (RWA) and 2. control moment gyroscopes (CMG). Currently, the Fly-Hopper will use RWAs for the purpose of simplicity. In the future, tradeoffs between RWAs and CMGs will be analyzed in depth. However, that analysis will not be covered in this paper, nor will control or simulation of the RWAs.

III. Minimal Coordinates Simulation

The dynamics of a five-bar parallel mechanism in minimal coordinates is similar to that of two double pendulums, but with the addition of a constraint to create a joint between the two distal links (See Fig. 3). The dynamics are explained below.

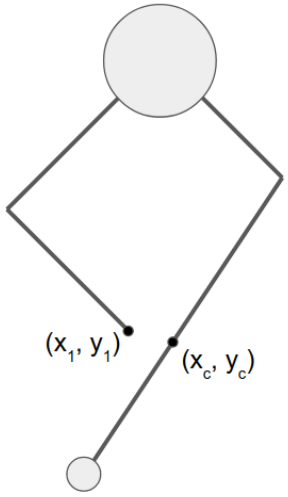

Fig. 3. Point (x_1, y_1) on the first double pendulum is constrained to point (x_c, y_c) on the second double pendulum. Image created in Google Slides.

A. Joint Constraints in Minimal Coordinates

The parallel joint constraint, based on the combination of the two points shown in Figure 3 above, is found simply by taking the difference between them as shown in (4) through (6). Ideally, joint constraint $c(q)$ should be equal to zero at all times.

$$c_x = x_1 - x_c \tag{4}$$

$$c_y = y_1 - y_c \tag{5}$$

$$c(q) = \begin{bmatrix} c_x \\ c_y \end{bmatrix} \tag{6}$$

It is desired that the constraint function's first and second derivatives are also equal to zero. They are found as shown in (7) through (9) below. The Jacobian of the constraint function is denoted as D(q) here.

$$D(q) = jacobian(c) = \frac{\partial c}{\partial q} \tag{7}$$

$$\dot{c} = \frac{\partial c}{\partial q}\dot{q} = D(q)\dot{q} \tag{8}$$

$$\ddot{c} = \frac{d}{dt}(D(q)\dot{q}) = D(q)\ddot{q} + \frac{\partial}{\partial q}(D(q)\dot{q})\dot{q} = 0 \tag{9}$$

Finally, the dynamics are rearranged into the KKT system as shown. As this is a Differential Algebraic Equation (DAE), stabilization is necessary. The simplest stabilization method is Baumgarte stabilization[5], which is used here. Once (12) is attained, it is a simple matter of solving for $\ddot{q}$.

$$d = \frac{\partial}{\partial q}(D(q)\dot{q})\dot{q} \tag{10}$$

$$e = D(q) * (M(q) \backslash (D(q)^\top * (\alpha * c + \beta * \dot{c}))) \tag{11}$$

$$\begin{bmatrix} M(q)_{4\times4} & -D(q)^\top_{4\times2} \\ D(q)_{2\times4} & 0_{2\times2} \end{bmatrix} \begin{bmatrix} \ddot{q}_{4\times1} \\ \lambda_{2\times1} \end{bmatrix} = \begin{bmatrix} (-C(q,\dot{q}) - G(q))_{4\times1} \\ -(d+e)_{2\times1} \end{bmatrix} \tag{12}$$

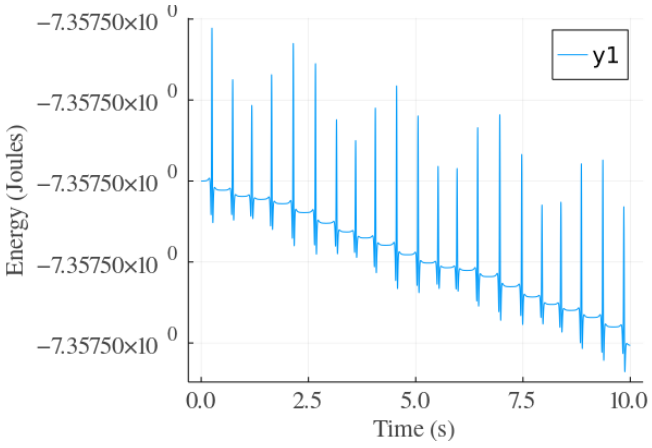

Fig. 4. Total energy vs. time for the idealistic model.

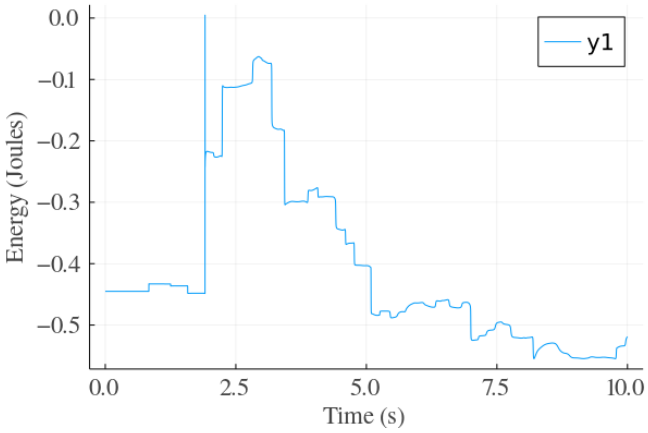

Fig. 5. Total energy vs. time for the CAD-based model.

B. Fixed-Base Minimal Coordinates Simulation

Simulation of the parallel leg was performed using fixed-base minimal coordinates dynamics in several ways. First, an idealized model of a five-bar parallel mechanism, assuming massless links with point masses at each link's center, was tested with symbolic differentiation. Next, a more accurate model of the robot leg was used, including link masses and inertia tensors from the CAD. In addition, a version using autodiffing was tested. The difference in computational speed between autodiffing and symbolic differentiation (not including precomputation) was negligible. All simulations used the RK4 integration. Figures 4 and 5 show total energy as a function of time for the idealized and design-based models, respectively. While total energy should be equal to zero at all timesteps for a closed system, it is likely that energy is being injected not just through numerical inaccuracies in RK4, but also through the Baumgarte stabilization. The accuracy can be improved through further tuning of the Baumgarte stabilization gains.

IV. Maximal Coordinates Simulation

In maximal coordinates, the state vector q is composed of the quaternion Q and position r of each body's CoM in the world frame. The Fly-Hopper's state vector is defined as shown:

$$q = \begin{bmatrix} r_b & Q_b & r_0 & Q_0 & r_1 & Q_1 & r_2 & Q_2 & r_3 & Q_3 \end{bmatrix}^\top \tag{13}$$

Where b denotes the base link of the robot and the numbered states correspond to links 0 through 3.

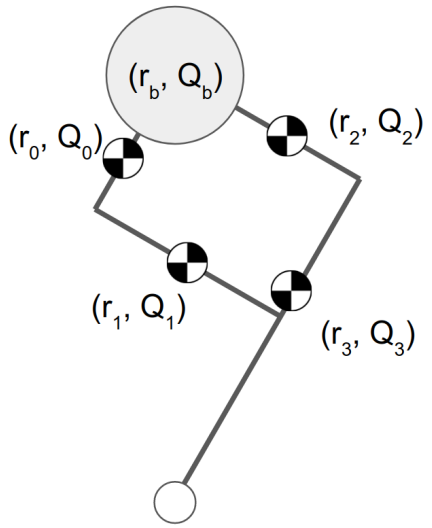

Fig. 6. Diagram identifying the components of the state vector on the robot. Image created in Inkscape.

A. Joint Constraints in Maximal Coordinates

In maximal coordinates, all joints between rigid bodies are modeled as constraints. As such, there is no difference between the "parallel constraint" joint and any other joint. These joint constraints make heavy use of quaternion rotations. For the sake of readability, a function for quaternion rotation of a vector is defined here:

$$Z(Q, r) = H^\top L(Q) R(Q)^\top H r \tag{14}$$

Where Q is the desired quaternion in the world frame and r is a position vector in $\mathbb{R}^3$ in the body frame. $L(Q)$, $R(Q)$, and H are defined in [16].

For the purpose of calculation of joint constraints, this function can be used to rotate a position r_B in the body frame by a quaternion Q_W in the world frame to get the position r_W in the world frame.

$$r_W = Z(Q_W, r_B) \tag{15}$$

A constraint between two joints can now be defined, as with minimal coordinates, by calculating the distance between the two desired points for connecting. As an example, the constraint between the base and Link 0 is defined as follows:

$$r_b + Z(Q_b, l_{cb}) - r_0 - Z(Q_0, -l_{c0}) = 0 \tag{16}$$

Where l_{cb} and l_{c0} are CoM positions relative to the joints in the link body frames.

However, (15) merely defines a spherical joint, whereas the Fly-Hopper's joints rotate in one axis. To constrain rotation between the base and link 0 to the y-axis (in the body frame) only, the following equation can be used:

$$YL(Q_b)' Q_0 = 0 \tag{17}$$

Where

$$Y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{18}$$

The full joint constraint vector can now be defined as follows:

$$c(q) = \begin{bmatrix} r_b + Z(Q_b, l_{cb}) - r_0 - Z(Q_0, -l_{c0}) \\ YL(Q_b)^\top Q_0 \\ r_0 + Z(Q_0, l_0 - l_{c0}) - r_1 - Z(Q_1, -l_{c1}) \\ YL(Q_0)^\top Q_1 \\ r_b + Z(Q_b, l_{cb}) - r_2 - Z(Q_2, -l_{c2}) \\ YL(Q_b)^\top Q_2 \\ r_2 + Z(Q_2, l_2 - l_{c2}) - r_3 - Z(Q_3, -l_{c3}) \\ YL(Q_2)^\top Q_3 \\ r_1 + Z(Q_1, l_1 - l_{c1}) - r_3 - Z(Q_3, l_c - l_{c3}) \end{bmatrix}_{23\times 1} \tag{19}$$

It is notable that the last joint (connecting links 1 and 3) does not have the y-axis constraint, as it would be redundant due to the parallel nature of the mechanism. The choice of which joint to leave spherical is arbitrary.

B. Control Force and Torque

Control inputs are characterized as a wrench applied to each body, where torques τ are applied in the body frame and the corresponding forces due to those torques on the body's CoM are applied in the world frame. Given a control input u to joints 0 and 2:

$$\tau_b = \begin{bmatrix} 0 & (-u_0 - u_2) & 0 \end{bmatrix}^\top \tag{20}$$

$$\tau_0 = \begin{bmatrix} 0 & u_0 & 0 \end{bmatrix}^\top \tag{21}$$

$$\tau_2 = \begin{bmatrix} 0 & u_2 & 0 \end{bmatrix}^\top \tag{22}$$

$$f_b = \frac{\tau_b \times l_{cb}}{||l_{cb}||^2} \tag{23}$$

$$f_0 = \frac{\tau_0 \times l_{c0}}{||l_{c0}||^2} \tag{24}$$

$$f_2 = \frac{\tau_2 \times l_{c2}}{||l_{c2}||^2} \tag{25}$$

Force and torque vectors are concatenated into the full input wrench matrix for the Fly-Hopper as shown in (26).

$$F_k = \begin{bmatrix} Z(Q_b, f_b) \\ \tau_b \\ Z(Q_0, f_0) \\ \tau_0 \\ 0_{3\times 1} \\ 0_{3\times 1} \\ Z(Q_2, f_2) \\ \tau_2 \\ 0_{3\times 1} \\ 0_{3\times 1} \end{bmatrix}_{30\times 1} \tag{26}$$

C. The Discrete Euler-Lagrange Equation

A special interpretation of the Discrete Euler-Lagrange (DEL) equation is used for variational integration. The main components are the "left" and "right" Discrete Legendre Transforms (DLT), the derivation for which can be found in [17]. For a floating rigid body, the "right" and "left" DLTs are shown in (27) and (28), respectively.

$$\frac{\partial L_d(q_k, q_{k+1})}{\partial q_k} = D_1 L_d \tag{27}$$

$$\frac{\partial L_d(q_{k-1}, q_k)}{\partial q_k} = D_2 L_d \tag{28}$$

$$D_1 L_d = \begin{bmatrix} -\frac{1}{h} m (r_{k+1} - r_k) - \frac{h}{2} mg \\ (2.0/h) G(Q_k)' T R(Q_{k+1})' H \bar{H}' L(Q_k)' Q_{k+1} \end{bmatrix} \tag{29}$$

$$D_2 L_d = \begin{bmatrix} \frac{1}{h} m (r_k - r_{k-1}) - \frac{h}{2} mg \\ (2.0/h) G(Q_k)' L(Q_{k-1}) H \bar{H}' L(Q_{k-1})' Q_k \end{bmatrix} \tag{30}$$

Where $G(Q)$ is the attitude Jacobian defined in [16], m is mass, g is the gravity vector, and h is the timestep size.

The DEL is then as shown in (31).

$$D_2 L_d(q_{k-1}, q_k) + D_1 L_d(q_k, q_{k+1}) = 0 \tag{31}$$

D. Contact Dynamics and Friction

To add contact and friction to the model, time-stepping is implemented with IPOPT [18]. The full optimization problem is as described below.

$$\min_{q_{k+1}, \lambda_k, s_k, b_k, \psi_k, n_k} \quad \epsilon (\lambda_k)^\top \lambda_k + \sum_{i=1}^{N_s} s_{i_k} \tag{32}$$

$$\text{s.t.} \quad DEL = 0 \tag{33}$$

$$(Q_{b_{k+1}})^2 - 1 = 0 \tag{34}$$

$$(Q_{0_{k+1}})^2 - 1 = 0 \tag{35}$$

$$(Q_{1_{k+1}})^2 - 1 = 0 \tag{36}$$

$$(Q_{2_{k+1}})^2 - 1 = 0 \tag{37}$$

$$(Q_{3_{k+1}})^2 - 1 = 0 \tag{38}$$

$$c(q_{k+1}) = 0 \tag{39}$$

$$J(q_{k+1}) * v_m + \psi_k \frac{b_k}{||b_k|| + \epsilon} \geq 0 \tag{40}$$

$$\phi(q_{k+1}) \geq 0 \tag{41}$$

$$s_{1_k} - n_k \phi(q_{k+1}) \geq 0 \tag{42}$$

$$\mu n_k - ||b_k|| \geq 0 \tag{43}$$

$$s_{2_k} - \psi_k (\mu n_k - ||b_k||) \geq 0 \tag{44}$$

$$s_k \geq 0 \tag{45}$$

$$\psi_k \geq 0 \tag{46}$$

$$n_k \geq 0 \tag{47}$$

Where:

- λ is the Lagrange multiplier, i.e dual variable for the joint constraint forces.
- s is an array of slack variables.
- ψ is the dual variable representing velocity reduction due to friction.
- b is a 2×1 vector representing friction force in the x and y directions.
- μ is the friction coefficient.
- n is the dual variable representing normal force.
- ϵ is equal to or less than the solver tolerance, e.g. 10^{-6}.
- $\epsilon(\lambda_k)^\top \lambda_k$ is a Tikhonov regularization for kinematic singularities.
- $J(q)$ is a 2×1 vector of the end-effector Jacobian's x and y components.
- v_m is a finite difference approximation of the velocity. $v_m = (q_{k+1} - q_k)/2$.
- (33) is the forced DEL equation described in (49).
- (34) through (38) are used for quaternion normalization.
- (40) minimizes dissipation of kinetic energy to prevent creep.
- (41) is the signed distance function described in (48) below.
- (42) is a relaxed complementarity constraint for contact.
- (43) is the friction cone constraint.
- (44) is a relaxed complementarity constraint for friction.
- $||b||$ is calculated as follows: $||b|| = \sqrt{b^\top b + \epsilon^2} - \epsilon$. This helps to prevent numerical issues and allows IPOPT to converge to $tol = \epsilon$.

The signed distance constraint function ϕ returns the z-axis distance from the lowest point of the foot to the ground plane, which is assumed to be flat and level.

$$\phi(q) = r_{3_z} + Z(Q_3, l_{ee} - l_{c3})_z - \frac{d_f}{2} \tag{48}$$

Where l_{ee} is the distance vector in the body frame between Joint 3 and the end effector (foot), and d_f is simply the diameter of the spherical foot. The output is a scalar.

The signed distance constraint force is then added to the DEL equation in the same way that the joint constraint function is, with its own dual variable (which is equivalent to the normal force).

E. The Forced Discrete Euler-Lagrange Equation

The Fly-Hopper leg is composed of five rigid bodies, so the translation and rotation components of (31) are repeated for each body. The end result is a 30x1 vector. This vector is combined with the input wrench, joint constraint forces, contact force, and friction wrench from the previous sections to form the full forced DEL.

$$D_2 L_d + D_1 L_d + \frac{h}{2} F_{k-1} + \frac{h}{2} F_k + h \frac{\partial c(q_k)}{\partial q}^\top \lambda + h \frac{\partial \phi(q_k)}{\partial q}^\top n + h B_k = 0 \tag{49}$$

The friction wrench B_k is calculated in a similar manner to the input wrench in Section IV-B, but body frame torques are

calculated from the world frame translational friction forces b_k rather than the other way around.

F. Simulation Results

The maximum coordinates simulation was implemented in Julia. A timestep size of 0.01 s was used, and visualizations were generated using both simple geometric shapes and using the URDF. Fig. 7 shows position of Joint 0 following a simple joint space PD command in zero gravity.

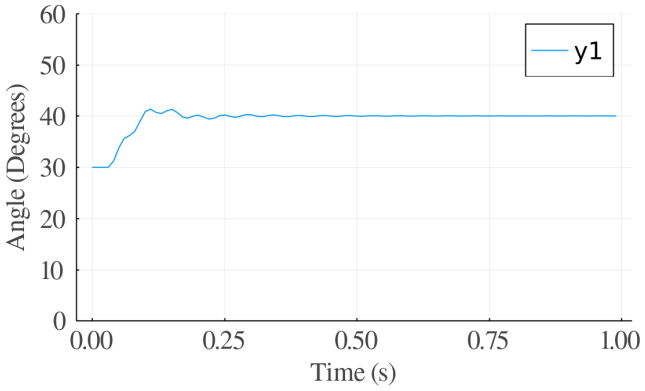

Fig. 7. PD control of Joint 0 of the simulated robot.

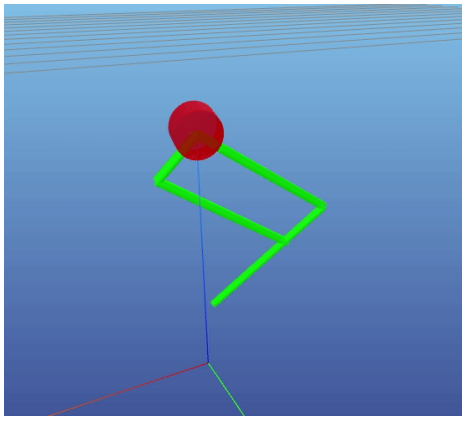

Fig. 8. Geometric visualization of the simulation.

The full implementation of Section IV in code including contact and friction is not yet complete.

V. Discussion and Future Work

In this paper, we presented the dynamics and simulation of a parallel linkage robot in both minimal and maximal coordinates. The immediate next steps in this work are to finish the code for the full maximal coordinates simulation with contact and friction, and add constraints for x- and y-axis translation and rotation of the robot base so that a simple vertical hopping controller can be tested.

It is notable that while the dynamics methods discussed in this paper were applied for simulation only, they can also be used for control methods such as trajectory optimization and model predictive control. This is a subject of interest for future work. Another important objective for future work is a quantitative comparison between traditional dynamics methods and the maximal coordinate dynamics explored here for the Fly-Hopper.

While the methods shown here are useful tools, significant improvements can still be made across a variety of aspects. For example, the code written for this project is far from optimized for computational efficiency.

Furthermore, the actuator model from Section II-B has not been included in the simulation methods presented in Sections III and IV. Neither has a model for the leg's parallel spring. These will be included in the future.

In addition, while minimal coordinates simulation with fixed-base dynamics was covered in Section III, one method in minimal coordinates that was not covered was floating-base dynamics with constraints. In fact, floating-base dynamics was tested using Newton's method, but the joint constraint, contact, and friction were not included. We may return to this subject in the future for comparison purposes.

Finally, one of the most immediate tasks for Fly-Hopper is to include the moment control system in the dynamics, control, and simulation of the robot so that iterative design and analysis can be performed on it.

The source code for this project is available at (https://github.com/bbokser/hopper-sim).

Acknowledgment

I would like to thank Dr. Zachary Manchester (Carnegie Mellon University), Kevin Tracy (Carnegie Mellon University), and Simon Le Cleac'h (Stanford University) for the generous advice they provided on this project.

References

[1] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[2] T. Apgar, P. Clary, K. Green, A. Fern, and J. W. Hurst, "Fast online trajectory optimization for the bipedal robot cassie.," in *Robotics: Science and Systems*, vol. 101, p. 14, 2018.

[3] D. Kim, Y. Zhao, G. Thomas, and L. Sentis, "Assessing whole-body operational space control in a point-foot series elastic biped: Balance on split terrain and undirected walking," *arXiv preprint arXiv:1501.02855*, 2015.

[4] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[5] Z. Manchester, "Advanced robot dynamics lecture notes," Fall 2021.

[6] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, IEEE, 2012.

[7] M. Parmar, M. Halm, and M. Posa, "Fundamental challenges in deep learning for stiff contact dynamics," *arXiv preprint arXiv:2103.15406*, 2021.

[8] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Computer methods in applied mechanics and engineering*, vol. 1, no. 1, pp. 1–16, 1972.

[9] J. Brüdigam and Z. Manchester, "Linear-time variational integrators in maximal coordinates," *arXiv preprint arXiv:2002.11245*, 2020.

[10] A. Abate, J. W. Hurst, and R. L. Hatton, "Mechanical antagonism in legged robots.," in *Robotics: Science and Systems*, vol. 6, Ann Arbor, MI, 2016.

[11] Y. Ding and H.-W. Park, "Design and experimental implementation of a quasi-direct-drive leg for optimized jumping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 300–305, IEEE, 2017.

[12] B. G. Katz, *A low cost modular actuator for dynamic robots*. PhD thesis, Massachusetts Institute of Technology, 2018.
[13] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Spröwitz, 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.
[14] P. M. Wensing, A. Wang, S. Seok, D. Otten, J. Lang, and S. Kim, "Proprioceptive actuator design in the mit cheetah: Impact mitigation and high-bandwidth physical interaction for dynamic legged robots," *Ieee transactions on robotics*, vol. 33, no. 3, pp. 509–522, 2017.
[15] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.
[16] B. E. Jackson, K. Tracy, and Z. Manchester, "Planning with attitude," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5658–5664, 2021.
[17] Z. R. Manchester and M. A. Peck, "Quaternion variational integrators for spacecraft dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 1, pp. 69–76, 2016.
[18] V. M. Zavala, C. D. Laird, and L. T. Biegler, "Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems," *Chemical Engineering Science*, vol. 63, no. 19, pp. 4834–4845, 2008.