2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Design and Comparative Analysis of 1D Hopping Robots

Eric Ambrose[1], Noel Csomay-Shanklin[2], Yizhar Or[3], and Aaron Ames[1]

Abstract— **Hopping is a highly dynamic motion requiring precise input over brief moments of ground contact in order to achieve desired performance. While this problem has been approached from multiple perspectives, this work provides a comparative analysis of two robot models. The first model uses an actuator to store energy in a spring and release it during the ground phase, while the second uses an actuator to move an additional mass vertically to generate force on the spring. In the first model, analytic expressions are used to find the desired controllers, while trajectory optimization is used in the latter. Orbital stability of each model under the conditions of uncertain damping and poor estimation of the hop height is examined. To this end, Poincaré analysis is used to give a metric of stability in the presence of different initial conditions and parameter uncertainty. Simulations show that the first model converges quickly to a point near the desired height determined by the amount of uncertain damping present. The second model is less robust to uncertainty, but is be made to converge to a desired height with the addition of PD control around the optimal trajectory. This robustness is improved with different gains in the controller. In experiments performed on hardware for the second model, stability is observed through convergence to a periodic orbit within several hops.**

I. Introduction

The hopping motion is fundamentally hybrid in nature, consisting of multiple phases of differing governing dynamics as well as discrete events [1], [2]. Similar to the case of bipedal running [3], [4], there are two different domains of continuous dynamics separated by very dramatic discrete impacts. These impacts cause energy to be lost beyond what would normally be lost in a robot due to friction alone. In order to make up for this disruption and reach a desired hopping height, the robot must add energy back in through actuation. The easiest way to do this is by applying a force on the world while the robot is in contact with the ground, and then spend the time in the air preparing for that action. However, the duration of the ground phase tends to be much smaller than that of the aerial phase, leading to a need for very brief and powerful actuation during this time.

Robotic hopping is not a new problem, but one that has been studied for decades [5], [6], [7], with many approaches to the problems of actuation and control. Two methods of overcoming the issue of inputting energy during the short ground phase have been followed. The first is the idea of decreasing the amount of energy lost at each impact, to therefore reduce the amount that must be input back to the system by the actuator. This has been accomplished by decreasing the mass of the robot's foot [8], or by taking shorter and higher frequency hops [8], [9]. The second way of improving actuation is to increase the amount of time spent on the ground to allow for longer durations of energy input. This is typically done by using a leg with a large range of motion and an elastic element [10], [11]. By increasing the duration of the ground phase, the on-board actuators can be less powerful and lighter while still having sufficient time to provide the required energy to the system. Work has also been done to increase the capabilities of robots to jump higher [12], [13] or over difficult terrain [14]. For the case of the Urban Hopper and the Sand Flea robot, extreme jump heights were achieved using compressed energy that could be released at a chosen moment. However, storing the amount of energy needed to jump to these heights required more time than the robot was in the air, preventing the immediate succession of another jump. This type of energy release could also be unsafe due to the high impulse force.

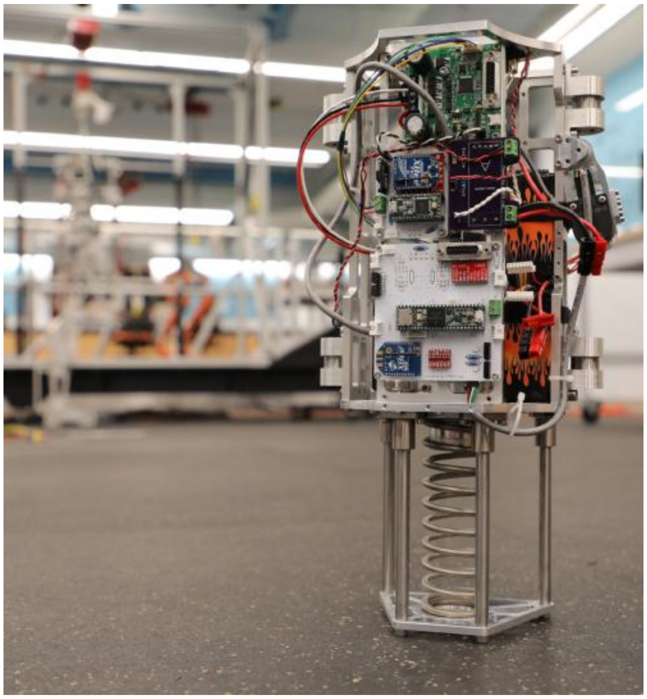


Fig. 1: A 1-Dimensional, moving-mass hopping robot used to experimentally demonstrate the results of this paper

This work was supported by Disney Research and Development

[1] E. Ambrose and A. Ames are with the Department of Mechanical and Civil Engineering at the California Institute of Technology in Pasadena, CA (eambrose, ames)@caltech.edu

[2] N. Csomay-Shanklin is with the Department of Mechanical Engineering at the Georgia Institute of Technology in Atlanta, GA noelcs@gatech.edu

[3] Y. Or is with the Department of Mechanical Engineering, Technion - Israel Institute of Technology in Haifa 3200003, Israel izi@technion.ac.il

978-1-7281-4004-9/19/$31.00 ©2019 IEEE

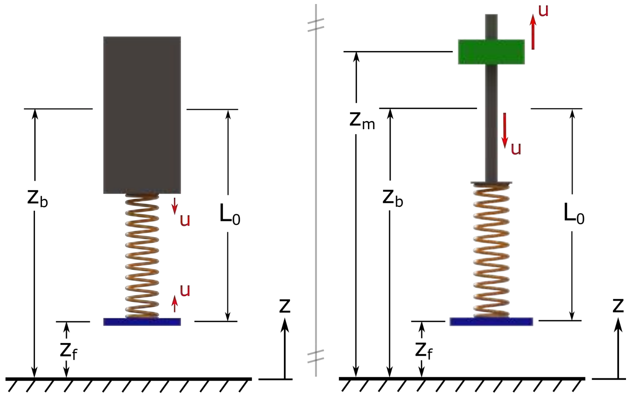


Fig. 2: Model coordinates for the Compress-Release (left) and Moving-Mass (right) hopping robots.

With the end goal of developing robots which can freely hop around in safe ways, the group at Disney Research and Development began working towards hopping with safe actuation via their LEAP robot [15]. The idea of safe actuation entails placing an elastic element between the robot and the world and, in doing so, creating a limit on the impulse that can be transmitted between the two. This can be best seen in the patent [16], detailing a robotic bouncing ball consisting of an elastic shell and internal actuators. Given the size constraints of this elastic element and the robot itself, further limits on the duration of the ground phase, the range of motion for the robot, and total amount of energy storage are placed on the system. Such constraints prevent the use of certain energy input strategies during the ground phase.

The goal of this work is to begin examining methods of hopping which will work in such a scenario while still achieving sufficient stability in the process. The two methods analyzed here for storing energy in the spring are a clutch-release method, similar to the style of the Bow Leg hopper [6], and a moving-mass method, as in [17], to generate force during the ground phase. The model dynamics and hybrid domain structure for each of these robots will be described in Section II, along with the exact method of actuation used. Analysis of each system will be performed through simulation using Poincaré mapping in Section III, with the goal of observing the effects of model uncertainty and poor estimation of hop height. Experiments were performed for the moving-mass hopping model in order to validate the results of that analysis, and as a proof of concept for that model. These results are presented in Section IV, followed by discussion and conclusions in Section V.

II. Mathematical Models of 1D Hoppers

Two hopping robots are examined in this work, with different designs and methods of actuation. The coordinates, q, of each model represent the heights of the individual entities from the ground, as shown in Fig. 2. The corresponding configuration space is given by $q \in Q \subset \mathbb{R}^n$, where n is the number of coordinates for the given model. For the tangent bundle with coordinates $(q,\dot{q}) \in TQ \subset \mathbb{R}^{2n}$, the hybrid control system is defined as the tuple,

$$\mathscr{HC} = (\Gamma, \mathscr{D}, \mathscr{U}, S, \Delta, FG) \tag{1}$$

- $\Gamma = \{V, E\}$ is a *directed cycle* containing vertices V and edges E
- $\mathscr{D} = \{\mathscr{D}_v\}_{v \in V}$ is the set of admissible domains
- $\mathscr{U} \in \mathbb{R}$ is the set of admissible control inputs
- $S = \{S_e\}_{e \in E}$ is the set of guards for domains, which represents the transition point between domains
- $\Delta = \{\Delta_e\}_{e \in E}$ is the set of reset maps between domains
- FG is the set of vector fields representing continuous dynamics of the domains

Fig. 3 contains a visual representation of the hybrid domain cycle for each model.

The dynamics during each domain can be written as

$$M\ddot{q} + H(q,\dot{q}) = Bu + J_v^T(q)F_v \tag{2}$$

where $M \in \mathbb{R}^{n \times n}$ is the mass matrix, $H \in \mathbb{R}^n$ is a matrix containing the gravity, damping and spring forces, $B \in \mathbb{R}^n$ is the actuation distribution matrix, and $u \in \mathbb{R}$ is the input force from the actuator. In addition, $J_v \in \mathbb{R}^n$ is the Jacobian of the holonomic constraint in D_v and $F_v \in \mathbb{R}$ is the magnitude of that holonomic constraint force. The method of defining these constraints and their forces is based on the work in [18]. During the ground phase this holonomic constraint represents the foot being held in place by ground forces. The beginning of the aerial domain is subject to another holonomic constraint in the form of a hardstop around the spring, preventing the spring from extending past its equilibrium length. As a consequence of this constraint, the foot and body of the robot will move as one entity together until a compressive force is placed on the spring again. The purpose of this hardstop is to allow for a simpler spring design and to cause the robot to leave the ground at a known moment, i.e. when the spring reaches its equilibrium length.

Some of the transitions between domains involve what are assumed to be perfectly plastic impacts, leading to reset maps governing sudden changes of velocity for some coordinates. In Fig. 3 the reset maps at landing and take-off are represented by Δ_L and Δ_T, respectively. The method of getting a reset map is based on the work in [19], where the velocity change is deemed a result of an instantaneous impulse force. The map for landing sets the velocity of the foot to zero instantly, while the reset map at take-off is a conservation of momentum equation between the foot and body given by

$$\dot{z}_{f+} = \dot{z}_{b+} = \Delta_T \dot{z}_{b-} = \frac{M_b}{M_b + M_f}\dot{z}_{b-} \tag{3}$$

where M_b and M_f are the masses of the body and foot, and the scripts $-$ and $+$ represent information from before and after the reset, respectively. This means that as the body of the robot rises up and the spring reaches the hardstop, then the foot and body velocities will instantaneously change to a new and equal value.

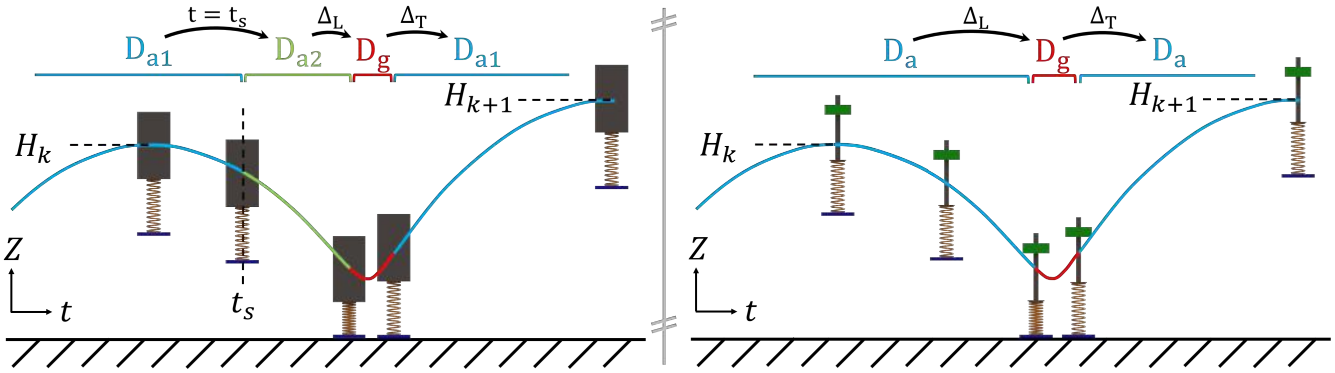


Fig. 3: Hybrid cycles for both the Compress-Release Hopper (left) and the Moving-Mass Hopper (right).

The following subsections will further detail each robot and their corresponding hybrid control system.

A. Compress-Release Hopper (CRH)

The CRH hopper uses an actuator within the body to compress a spring and store energy while in the air, and then releases that energy during the ground domain. In order to simplify analysis, a hardstop is placed around the spring which prevents extension past its equilibrium length. This results in the air phase being split between two parts. The first aerial domain has the hardstop active and the spring un-compressed, while the second aerial domain is the spring compression phase when the hardstop is no longer active. The end result is a hopping motion with three domains, as seen in the left half of Fig. 3. This cycle starts with the robot entering the first air domain, D_{a1}, with the hardstop in effect until a certain time, $t = t_s$. The robot then begins compressing the spring, removing contact with the hardstop and entering a second air domain, D_{a2}, where there is no internal constraint. This continues until the foot contacts the surface, beginning the ground domain, D_g. While on the surface, the robot reverses direction and leaves the ground when the body extends to the hardstop length, L_0. At this moment, time is reset and the cycle repeats. The definitions for these domains and their boundary guard surfaces are

$$D_{a1} = \{(t,q,\dot{q}) \in \mathbb{R} \times TQ : z_f \geq 0, t \leq t_s\} \tag{4a}$$

$$S_{a1} = \{(t,q,\dot{q}) \in D_{a1} : t = t_s\} \tag{4b}$$

$$D_{a2} = \{(t,q,\dot{q}) \in \mathbb{R} \times TQ : z_f \geq 0, t > t_s\} \tag{4c}$$

$$S_{a2} = \{(t,q,\dot{q}) \in D_{a2} : z_f = 0, \dot{z}_f < 0\} \tag{4d}$$

$$D_g = \{(q,\dot{q}) \in TQ : z_f = 0, z_b \leq L_0\} \tag{4e}$$

$$S_g = \{(q,\dot{q}) \in D_g : z_b = L_0, \dot{z}_b > 0\} \tag{4f}$$

where here, $q = [z_b, z_f]^T$ and $Q \subset \mathbb{R}^2$.

For the CRH model, there is a specific amount of pre-compression to generate in the spring during the aerial phase, δ^*, that will yield the correct return height of the robot after a full period of motion. Once δ^* has been calculated, as done in Section III, a cubic Bezier polynomial is used as the desired profile for the compression of the spring given by

$$B(\tau) = \delta^*(3(1-\tau)\tau^2 + \tau^3) \tag{5}$$

$$\tau = mod(t, t_s)/t_i \tag{6}$$

where t_s is the time after entering D_{a1} to begin spring compression, and t_i is the anticipated time to impact from t_s which can be calculated from height and velocity information. To minimize energy consumption of the motor, it is best to set t_s as large as possible; however, in order to ensure that the hardware is capable of following the given trajectory, $B'(\tau) \leq v_{max}$ and $B''(\tau) \leq a_{max}$ need to be enforced over the compression profile, where v_{max} and a_{max} are the maximum velocity and acceleration of the motor, respectively.

B. Moving-Mass Hopper (MMH)

The second hopper utilizes a moving mass within the body of the robot, which is actuated in series with the spring via a ball screw in order to apply a force to the spring during the ground phase, as seen in Fig. 2. The dynamics for this hopper are setup in the same way as in (2), but with a few minor changes. The inherent differences between this model and the CRH model revolve around this third mass: the mover. Firstly, there are now three coordinates used to represent the hopper as seen in Fig. 2, so $q = [z_b, z_f, z_m]^T \in \mathbb{R}^3$ and $Q \subset \mathbb{R}^3$. Secondly, the actuator is now a motor moving itself along a ball screw and is therefore acting between the body and mover, which will be reflected in the actuation matrix, B. Lastly, there is no actuator force across the spring so there is only one domain within the aerial phase leading to a simpler domain cycle, as seen in Fig. 3. There are still the same two impacts due to the ground and a hardstop around the spring, but the aerial domain and its guard are now defined as

$$D_a = \{(q,\dot{q}) \in TQ : z_f \geq 0\} \tag{7a}$$

$$S_a = \{(q,\dot{q}) \in D_a : z_f = 0, \dot{z}_f < 0\}\} \tag{7b}$$

III. Analysis

In this section, orbital stability analysis will be performed for each hopper model using a Poincaré map [20]. The Poincaré section is placed at the hop apex, given by

$$\Sigma = \left\{(q,\dot{q}) \in TQ : z_f > 0, \dot{z}_b = 0\right\} \tag{8}$$

Four cases of damping and control action will be examined in simulation. The first is with actuation assuming there is no damping in the robot joints and the simulation also includes no damping. The second case will take this same action but in simulation containing damping to see how it performs. The third trial will use actuation based on a robot model with damping during the ground phase and in a simulation with that exact same amount of damping present. The final case will show how the actuation from the third case performs with a larger amount of damping in the simulation.

A. *Compress-Release Model*

The value of compression needed in the spring, δ^*, is found through calculating the amount of spring energy that will make up for the losses from friction and impacts, as well as any desired height change. The complexity of determining this compression value gets increasingly difficult when including the damping terms of the dynamics. In order to analyze the stability of the hopper and its control methods, we will begin with the case where damping is not considered and then work towards the case with damping during the ground phase. By not considering these damping terms, we can determine whether or not the robot is able to converge to the goal height with corresponding model uncertainty.

Due to the spring hardstop, the foot coordinates are fixed relative to the body during flight, which leads to the Poincaré section of this model being a one-dimensional curve in the z_b coordinate. The simplicity of this section allows for the Poincaré map to be easily plotted as a scalar map, as seen in the subplots of Fig. 4.

The control input for this system is the amount of potential energy to generate through spring compression during the second aerial domain. In order to calculate these controllers, we also define the following height characteristics:

- H_k, the actual apex height of the current hop
- H^*, the desired height to reach at the apex of the next hop
- H_{k+1}, the actual apex height reached on the next hop
- H_c, the value given to the controller as the apex height of the current hop

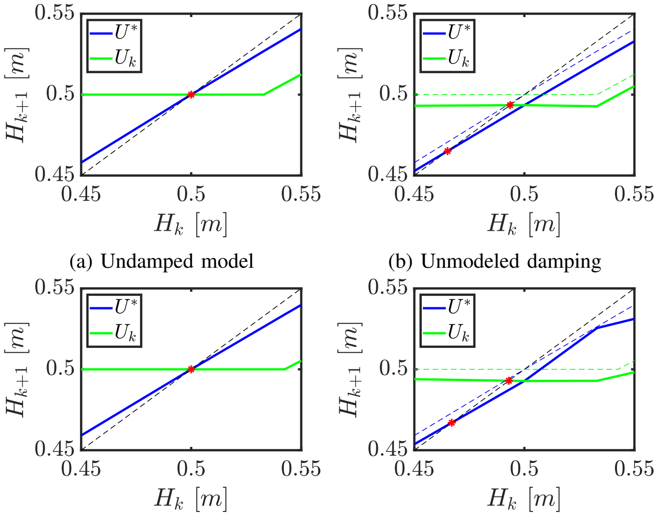
Note that this H_c variable is used as a means of simulating uncertainty in the state measurement for the body height. By giving incorrect information to the system, we can observe any bounds on the allowable uncertainty.

1) Undamped Cases: In order to find the amount of energy needed to input, we must first find how much energy is expected to be lost during our next hop if we were to reach the desired height. We will do this by calculating the energy lost in each impact and also the energy required to make any anticipated height change. In the case without damping, these are given by

$$E_L(H_c) = M_f g(H_c - L_s) \tag{9a}$$

$$E_T(H^*) = (M_b + M_f)g(H^* - L_s)\left(\frac{M_f}{M_b}\right) \tag{9b}$$

$$E_H(H_c, H^*) = (M_b + M_f)g(H^* - H_c) \tag{9c}$$



(a) Undamped model (b) Unmodeled damping

(c) Damped model (d) Additional damping

Fig. 4: Poincaré maps for the CRH.

where E_L is the energy lost in the landing impact due to the foot contacting the ground, E_T is the energy lost in the take-off impact due to the foot being picked back up off the ground, and E_H is the potential energy required to change height from the current hop height to the next. In order to reach the desired hop height, the energy we input in the spring, $u(H_c, H^*)$, must be equal to the sum of these three energy values

$$u(H_c, H^*) = E_L(H_c) + E_T(H^*) + E_H(H_c, H^*) \tag{10}$$

and from this, we get the desired compression:

$$\delta^* = \sqrt{\frac{2u(H_c, H^*)}{k_s}} \tag{11}$$

where k_s is the stiffness of the spring.

With this controller in mind, we can assess the stability around a desired hop height. A simple closed-form solution exists for this undamped case. We are able to calculate the total amount of energy lost from impacts for a hop using the given input spring energy, and then determine the final hop height. The total amount of energy added into the system after impact losses is

$$\Delta E = H^* g(M_b + M_f) - \frac{H_c M_b^2 g}{M_b + M_f} - \frac{H_k g(2M_b M_f + M_f^2)}{M_b + M_f} \tag{12}$$

Using this added energy value, we can find the corresponding height change

$$\Delta H = \frac{\Delta E}{(M_b + M_f)g} \tag{13}$$

With this, we can find the actual final height for the system

$$H_{k+1} = P(H_k, H_c, H^*) = H_k + \Delta H \tag{14}$$

which is also the Poincaré map of the system under this controller. Substituting (12)-(13) into (14) yields the full form of the Poincaré map

$$P(H_k,H_c,H^*) = H^* - \frac{M_b^2(H_c - H_k)}{(M_b + M_f)^2} \tag{15}$$

Illustrated in Fig. 4a is the Poincaré mapping for control action taken based on two different choices of H_c: either $H_c = H^*$, labeled as U^*, or $H_c = H_k$, labeled as U_k. In the case of the former, we would not be measuring the actual height of the robot, but instead always choose the control input that is associated with our theoretical periodic solution. On the other hand, when we set $H_c = H_k$ we are using the exact information of the current hop height. The convergence of each of these cases, i.e. the stability of the Poincaré map, can be seen in the figure, or found by taking the derivative of the Poincaré map with respect to the current hop height:

$$\frac{\partial P}{\partial H_k} = \frac{M_b^2}{(M_b + M_f)^2}(1 - \frac{\partial H_c}{\partial H_k}) \tag{16}$$

When $H_c = H_k$, this derivative evaluates to 0, which shows that our controller would drive the robot to the desired height in a single hop as a deadbeat controller. For the case where $H_c = H^*$, the derivative is slightly less than 1, which still yields asymptotic convergence, albeit slowly.

Due to the damping present in the hardware, the plots in Fig. 4a do not show an accurate representation of the return height for a hop starting from each H_k. In order to see the performance of the controlled system with damped equations of motion, these same controllers were used in simulation which integrated the dynamics through each domain to give the return height. The same controllers from before were used, and yet still gave convergence as seen in Fig. 4b. Here the solid lines are the curves from the damped dynamics, while the dashed lines are that from the previous case. It can be seen that both controllers still lead to convergence, just to a different point then our desired height. Now, using the actual previous height yields not only faster convergence, but convergence to a height that is closer to our desired height. As the damping value increases away from the ideal un-damped model, the convergence point will start to move away from the desired height.

2) Ground Damping Cases: We now consider the case when damping during the ground phase of hopping is added in the form of spring material damping. Since the air domains here remain deterministic, the key calculation only relates to the dynamics of the ground domain. To this end, only the height of the body needs to be used, as the foot is always fixed to the ground during this time. To simplify notation in this section, the body height coordinate is referred to as z. The dynamics of the ground phase become that of a single degree of freedom spring-mass-damper system, which has a known solution of

$$z(t) = e^{at}(z(0)\cos(bt) + \frac{-az(0) + \dot{z}(0)}{b}\sin(bt)) \tag{17}$$

$$a = -c_s/(2M_b)$$
$$b = (k_s/M_b - c_s^2/(4M_b^2))^{1/2}$$

where $z(0)$ and $\dot{z}(0)$ are the starting position and velocity of the body, and c_s is the damping coefficient of the joint. Note that this is the solution around the equilibrium length of the spring. Therefore, the L_0 term will be excluded until later since this will only shift the curve. For our case, (17) represents the equation of motion during the time between impact and take-off. For a given hop height, H_k, and spring compression, δ_k, the height of the next hop can be calculated from the known initial conditions at the landing impact, $z_L, \dot{z}_L$, and moving forward in time to the known boundary condition at take-off, $z_T = 0$. With this information, the duration of the ground domain can be found analytically through the expression

$$t_g = -\frac{2}{b}\arctan\left[\frac{c}{b} + \frac{a}{b} - \frac{\dot{z}_L}{bz_L}\right] \tag{18}$$
$$c = \sqrt{a^2 z_L^2 - 2az_L\dot{z}_L + b^2 z_L^2 + \dot{z}_L^2}$$

Substituting t_g into the derivative of (17), the velocity of the robot before take-off, $\dot{z}_{T-}$, can be found. The height of the next hop is then found using this velocity and the impact map of take-off yields

$$H_{k+1} = \frac{(\Delta_T \dot{z}_{T-})^2}{2g} + L_0 \tag{19}$$

Conversely, the method of determining the required amount of compression to reach a certain hop height involves using the desired states at take-off, and then working backwards towards the initial conditions at landing. The take-off states are found by using the same boundary conditions as before. The landing states are unknown, but each are a function of the desired spring compression, δ^*. Taking these conditions along with (17) and its derivative, and simplifying yields the equation

$$e^{2at_g}(\frac{a^2}{b^2}\sin^2(bt_g) + \cos^2(bt_g)) = \frac{2g}{v_0}(\frac{e^{at_g}}{a}\sin(bt_g) - \frac{H_k - L_0}{v_0}) \tag{20}$$

which is only a function of the ground phase duration, t_g, that is now a negative value. Solving this transcendental equation for t_g must be done numerically. Substituting that value into (17) along with the take-off conditions, will yield the desired compression needed to overcome even the energy lost from damping. Using the fsolve function in MATLAB to find this value, a similar Poincaré plot is created and shown in Fig. 4c. Once again, exact convergence can be achieved in a range around the desired hop height with proper height measurement, and quick convergence elsewhere. Even in the case where steady-state control is used, the convergence is slightly faster than that of the un-damped case which is shown by dashed lines in the figure.

Simulation was also performed for the case with damping set 50% higher than the controller model. The results of this in Fig. 4d, show that once again the point of convergence

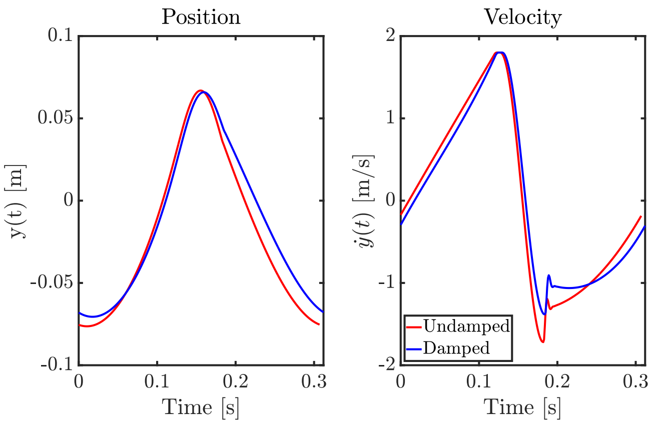


Fig. 5: Example trajectories for the y coordinate for both the damped and undamped models.

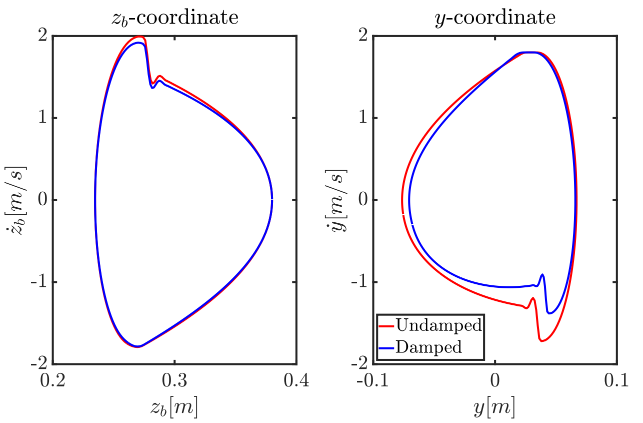


Fig. 6: Phase space representation for the z_b and y coordinates for both damped and undamped models.

is lowered from the desired point. The dashed lines here represent the curves from the ideal damped case. The exact convergence height relative to the desired is based on the level of uncertainty for both the height measurement and model damping. These results show that the robot should be stable under this control method, even with poor measurement of the hop height and some uncertainty in the damping estimation. It is clear that having a better estimate of the actual damping in the system will allow for a better choice of compression actuation.

B. Moving-Mass Model

This model uses an additional moving mass inside the body to input energy, which gives a third coordinate in the dynamics, z_m, and another source of internal friction. The presence of this moving mass inherently allows for many possible motions that will lead to a desired hop height change, since this is determined by the energy propagated to the spring during the ground phase. However, due to the indirect method of adding energy into the system and the extra friction, it becomes non-trivial to find those solutions. These issues point towards using trajectory optimization to generate the desired motions, and then tracking those trajectories during simulation or in practice. Consequently, characterizing the stability of these motions will be done numerically in MATLAB.

1) Generating Periodic Motions: The trajectory optimization package used in this project is *GPOPS-II* [21], which utilizes the *ipopt* solver. This software package allows users to specify the continuous and discrete dynamics, constraints, initial conditions, and the cost function in a convenient way within MATLAB and then maps those setup criteria into the form needed by the solver. An initial guess for the solution can be given in this framework as well. For this problem, it was found that the initial guess needed to be feasible, but not accurate in order to yield a solution.

The general optimization formulation is

minimize:

$$J(t) = \int_{t_0}^{t_f} U^2(t)dt \tag{21}$$

subject to:

$$\begin{aligned} \dot{x} = f(x) + g(x)U(t) \\ x_{min} \leq x \leq x_{max} \\ y_{min} \leq y \leq y_{max} \\ \dot{y}_{min} \leq \dot{y} \leq \dot{y}_{max} \\ -u_{max} \leq U(t) \leq u_{max} \\ x(t_0) = x(t_f) \end{aligned}$$

where $U(t)$ is the computed trajectory of the input force, $x = (z_b, z_m, z_f, \dot{z}_b, \dot{z}_m, \dot{z}_f)^T$ is the system state, and $y = z_m - z_b$ is the relative degree of freedom that is directly controlled by the actuator. For a given hop height, the initial conditions for the body and foot states are fixed, while the remaining mover states are only given bounds within the optimization framework. Note that the trajectory $U(t)$ is different from the control input u, so that the controllers may be formulated using this trajectory as a feed-forward term.

Using this method and the model parameters from hardware, trajectories were generated for the robot spanning a hop height range 0.32 m to 0.42 m. For this section, trajectories for a 0.38 m height with and without damping will be explored as an example. Fig. 5 shows the trajectory of the y coordinate for these example motions, while Fig. 6 shows the phase space orbits of both the body and relative coordinates of the robot.

2) Stability Analysis: Two controllers were run in simulation: open-loop (22), and open-loop with PD feedback on the y coordinate (23).

$$u(t) = U(T) \tag{22}$$

$$u(t) = U(T) + k_p(y_a - y_d) + k_d(\dot{y}_a - \dot{y}_d) \tag{23}$$

where $T = mod(t, t_f)$ and t_f is the duration of the generated hop motion. Once again, Poincaré analysis was used to assess

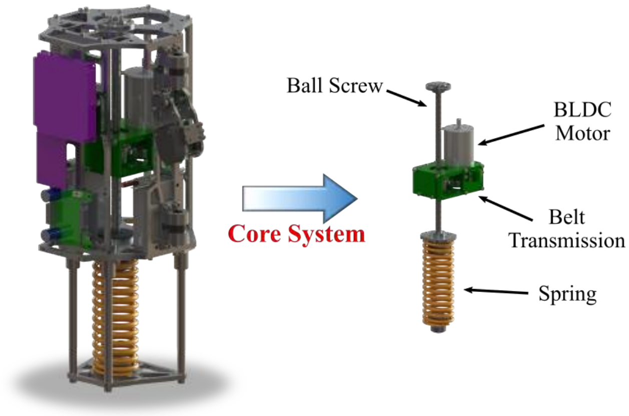


Fig. 7: CAD model of the MMH, with a clear view of the core components.

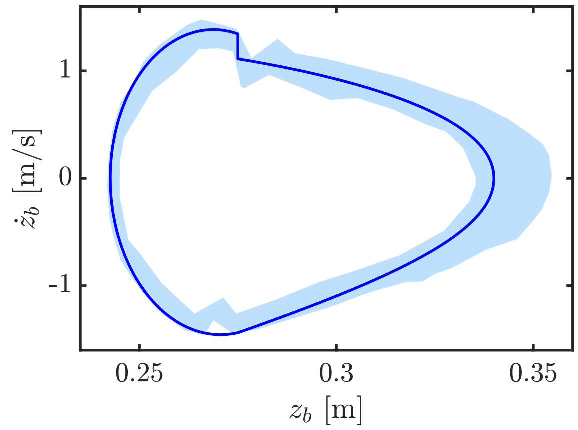


Fig. 8: Experiment hopping data (shaded blue) compared with simulated trajectory (dark curve), showing the existence of a periodic orbit.

the stability of the system under these controllers. With the additional coordinate, the Poincaré section for this model becomes a three dimensional surface along $(z_b, z_m, \dot{z}_m)^T$. Despite that the value of the mover coordinate is not critical for reaching the desired height, it still needs to be stable within the allowable range of motion due to hardware constraints. This necessitates achieving stability for all three dimensions of the Poincaré section, which can be determined by looking at the Jacobian of the section and its eigenvalues, λ_i, which are found using the MATLAB function *fsolve*. The system is stable if $|\lambda_i| < 1$ for all eigenvalues.

This was first done with the open-loop controller. The results in Table I show that the undamped system is not stable, while the optimal motion for the damped model is stable. However, the open-loop controller on this damped system had a small domain of attraction and was not robust to uncertainty in the damping terms. From here, a PD controller was added to improve stability and the domain of attraction. The gains of this controller were tuned to give optimal stability, while maintaining the input limits of the hardware. The right side of Table I shows that stability was dramatically improved with this controller. The domain of attraction also improved, now allowing for a wider range in initial conditions that lead to convergence.

TABLE I: Eigenvalues of Poincaré Map Jacobian

	OL Undamped	OL Damped$_1$	OL+PD Undamped	OL+PD Damped$_1$	OL+PD Damped$_2$
λ_1	1.4941	0.5177	0.0368	0.0757	0.5942
λ_2	0.8647	0.6395	0.0170	0.0312	0.3942
λ_3	1.4733	0.0630	0.0952	0.1277	0.0054

This stability analysis shows that (23) is robust to uncertainty in the hop height measurement, but it does not take model uncertainty of the damping terms into account. With simulation, it was shown that the chosen gains were not able to keep the system stable with uncertainty greater than 5% for the damping terms of the spring and the internal relative joint. The right-most column of Table I contains the stability analysis of the controller when gains were tuned to perform better with unmodeled damping. Compared with the ideal case, this controller took a larger number of hops to stabilize, but was able to do so with parameter uncertainty of up to 20% in the damping terms.

IV. Experiments

Running experiments with robots is valuable for showing the performance of these concepts in the real world. Currently, a robot for the MMH model has been custom built for this purpose. Fig. 7 shows the CAD model for the robot and a clear view of the inner workings. The force is input from a brushless DC motor connected to a ball screw via pulleys and a belt. This motor moves itself and some additional mass vertically along the ball screw, generating force onto the body of the robot, and therefore, onto one end of the spring. Using the known specifications of this motor and the ball screw transmission, max linear force and velocity of the actuator were found to be 100 N and 1.75 m/s, respectively. Due to the size of the body, the relative coordinate, y, is limited to $\pm$ 0.1 m, restricting the range that the mover can deviate from the body. The breakdown of mass within the robot is 55% in the body, 31% in the mover, and 14% in the foot.

These model parameters and constraints were given to the optimization described in Section III-B, resulting in hopping motions up to 0.42 m. The max force of the motor and mass of the foot prevent the robot from hopping any higher. These trajectories were taken and given to the controller on the robot in the form of desired input versus time. Experiments were run for each control method described previously, with the initial conditions being provided by the user dropping the robot from a chosen height. Data was collected from two onboard encoders monitoring the height of the body and the y coordinate. The hopper is constrained to be vertical at all times by linear bearings and rails on either side of the robot.

Experiments were run for at least 30 consecutive hops to show long term stability. It was seen in experiment that the initial condition of the drop did have to be fairly close to the

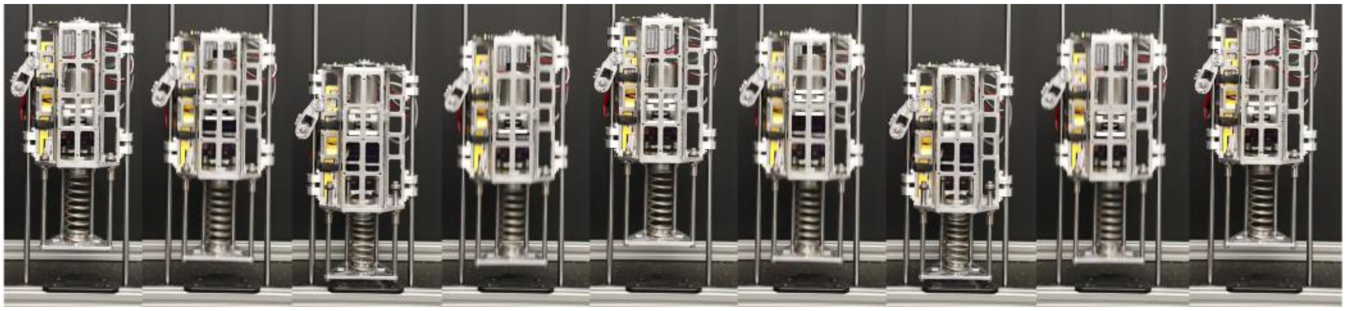


Fig. 9: Motion tiles of MMH robot over two consecutive hops.

ideal, but did not need to be perfect. This was apparent since the hopper needs to settle into the periodic orbit. This could take anywhere from 1 to 15 hops, depending on the timing and height of the initial drop. Fig. 8 shows data collected from an experiment for a hop height of 0.34 m, containing about 40 hops and an initial condition that was too high. The robot took about 8 hops to settle into its stable hop height. The data shows that the robot was able to remain around the simulated periodic orbit throughout the experiment. The two sections of data near the top and bottom of the plot show an oscillation in velocity of the robot that is not seen in simulation. This is likely due to the assumption of perfectly plastic impacts being not entirely true, showing that these impacts had a small bounce before coming to completion. Fig. 9 shows motion tiles of a two consecutive hops from the same experiment on our rail system. See submitted video.

V. Discussions and Conclusion

Analysis was performed for two different one-dimensional models of hopping robots for the purpose of understanding how to stabilize their vertical motion. Each of these mechanical models fit into the idea of safe actuation, which requires an elastic element to be placed between the robot and the world. It was shown that both cases were stable, even with non-ideal conditions and uncertainty. For the case of the moving-mass model, this stability was reached under the addition of PD control around the relative coordinate of the third mass within the body of the robot. This was further seen in experiments for the moving-mass model where the robot successfully stabilized itself using standard PD control along the nominal trajectory given from optimization.

The next steps of this work will be towards increasing the height of jumping, which will require more efficient mechanical designs. The changes will include lighter foot mass, stronger actuation, and higher energy storage capacity in the spring. Future work will also move towards 3-dimensional hopping with a similar emphasis on safe actuation.

References

[1] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek. Hybrid zero dynamics of planar biped walkers. *IEEE Transactions on Automatic Control*, 48(1):42–56, 2003.

[2] R. W. Sinnet, M. J. Powell, R. P. Shah, and A. D. Ames. A human-inspired hybrid control approach to bipedal robotic walking. *IFAC Proceedings Volumes*, 44(1):6904–6911, 2011.

[3] W. Ma, S. Kolathaya, E. R. Ambrose, C. M. Hubicki, and A. D Ames. Bipedal robotic running with DURUS-2D: Bridging the gap between theory and experiment. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 265–274. ACM, 2017.

[4] J. Park, J. Lee, J. Lee, K. Kim, and S. Kim. Raptor: Fast bipedal running and active tail stabilization. In *2014 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 215–215, Nov 2014.

[5] M. H. Raibert. Hopping in legged systems - modeling and simulation for the two-dimensional one-legged case. *IEEE Transactions on Systems, Man, and Cybernetics*, 14(3):451–463, May 1984.

[6] H. B. Brown and G. Zeglin. The bow leg hopping robot. In *Robotics and Automation (ICRA), IEEE International Conference*, 1998.

[7] P. Fiorini and J. Burdick. The development of hopping capabilities for small robots. *Autonomous Robots*, 14(2):239–254, 2003.

[8] G. Zeglin. The bow leg hopping robot. *Diss. Carnegie Mellon University*, 1999.

[9] M. H. Raibert, H. B. Brown, and M. Chepponis. Experiments in balance with a 3d one-legged hopping machine. *IJRR*, 3(2):75–92, 1984.

[10] D. W. Haldane, J. K. Yim, and R. S. Fearing. Repetitive extreme-acceleration (14-g) spatial jumping with salto-1p. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3345–3351, 2017.

[11] J. Hwangbo, V. Tsounis, H. Kolvenbach, and M. Hutter. Cable-driven actuation for highly dynamic robotic systems. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8543–8550. IEEE, 2018.

[12] J. R. Salton et al. Urban hopper. *SPIE*, 7692:7692 – 7692 – 9, 2010.

[13] Sand Flea Hopper. `https://www.bostondynamics.com/sandflea`.

[14] 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 *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3887–3892, Oct 2016.

[15] Untethered One-Legged Hopping in 3D Using Linear Elastic Actuator in Parallel (LEAP). `https://www.youtube.com/watch?v=M0ZXmGRCuts`.

[16] L. S. Smoot, G. D. Niemeyer, A. D. Ames, and D. L. Christensen. Robot bouncing ball. US Patent #: 10,092,850, Date of Patient: October 9 2018.

[17] J. Aguilar and D. I. Goldman. Robophysical study of jumping dynamics on granular media. *Nature Physics*, 12(3):278, 2016.

[18] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, Boca Raton, March 1994.

[19] J. W. Grizzle, C. Chevallereau, A. D. Ames, and R. W. Sinnet. 3D bipedal robotic walking: models, feedback control, and open problems. In *IFAC Symposium on Nonlinear Control Systems*, Bologna, September 2010.

[20] R. T. M'Closkey and J. W. Burdick. Periodic motions of a hopping robot with vertical and forward motion. *The International journal of robotics research*, 12(3):197–218, 1993.

[21] M. A. Patterson and A. V. Rao. Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Trans. Math. Softw.*, 41(1):1–37, Oct 2014.