Untethered One-Legged Hopping in 3D Using Linear Elastic Actuator in Parallel (LEAP)

Zachary Batts, Joohyung Kim, and Katsu Yamane

Disney Research,
4720 Forbes Ave, Lower Level Suite 110,
Pittsburgh, PA 15217
{zachary.batts,joohyung.kim,kyamane}@disneyresearch.com
http://www.disneyresearch.com/

Abstract. Current and previous single-legged hopping robots are energetically tethered and lack portability. Here, we present the design and control of an untethered, energetically autonomous single-legged hopping robot. The thrust-producing mechanism of the robot's leg is an actuated prismatic joint, called a linear elastic actuator in parallel (LEAP). The LEAP mechanism comprises a voice coil actuator in parallel with two compression springs, which gives our robot passive compliance. An actuated gimbal hip joint is realized by two standard servomotors. To control the robot, we adapt Raibert's hopping controller, and find we can maintain balance roughly in-place for up to approx. 7 seconds (19 hops) while continuously hopping.

Keywords: Hopper, legged-locomotion, spring-mass, parallel elastic actuator, voice coil actuator

1 Motivation, Problem Statement, Related Work

Legged robots are useful because, among other advantages [1], they can overcome uneven terrain, and can entertain an audience as they act out complex movements (e.g. different gaits). Single-legged robots have the simplest topology in the class of legged systems, and are limited to a hopping gait. Not only do single legged hopping robots provide a simplified testbed for locomotion control algorithms [2], they also demand high-speed, high-force actuation to achieve safe and robust ground-clearance and subject the actuator to greater mechanical stresses than do multi-legged systems. For these reasons, single-legged hopping robots provide an ideal benchmark for actuators used in legged locomotion.

The actuation requirements for a single-legged system are so great that to date, to the authors' best knowledge, no untethered single-legged hopper has achieved continuous hopping without using offboard power. Previous successful hopping robots (e.g. [3]) are tethered to stationary motors and/or power sources to avoid overburdening the robot. Sayyad et al. [4] provide a thorough review of single-legged hopping robots up to 2007, and note portability as a critical

stepping stone for commercial applications. The present authors have found no more recent examples of research that have achieved this goal.

Here, we attempt to "cut the tether" and create an untethered energetically-autonomous single-legged hopping robot. We employ a linear elastic actuator in parallel (LEAP) [5], previously developed by the present authors, which places a voice coil actuator (VCA) in parallel with compression springs, to act as the primary weight-bearing actuator for our single-legged hopping robot. The parallel configuration lessens the force requirements of the VCA by offloading weight to the spring, and allows the VCA to directly compress or extend the spring, independent of foot contact (in contrast to a series elastic topology). We chose a voice coil as our actuator because it is electrically-powered, is direct-drive, has low moving inertia, has little friction (the coil and body do not make contact), can produce force at high speeds, and has linear force output. These characteristics allow us to power, control, and actuate our robot with onboard batteries, microcontroller, and actuators, respectively. In section 2, we provide a hardware description of our robot, detail our method to estimate center-of-mass velocity, and present our locomotion controller. In section 3, we give an overview of our simulation environment and optimized controller, and present the results of our physical experiment. We discuss the results in section 4.

2 Technical Approach

2.1 Robot Description

We designed our robot to be kinematically similar to Raibert's 3D hopper [2] so that we might use his simple controller as an "off-the-shelf" algorithm to control our robot. Our hopper is an open kinematic chain composed of four links (Fig. 1). The first "torso" link (mass $m_1 = 1.41$ kg) contains the power source (seven 11.1V 1300mAh LiPo batteries), microcontroller (Texas Instruments LAUNCHXL-F28377S), power circuitry, an IMU sensor (Xsens MTi-3-8A7G6-DK) which outputs filtered orientation and velocity increment data at 100 Hz. The second "thigh" link (mass $m_2 = 0.31$ kg) is composed of two identical geared servomotors (Dynamixel MX-64T) whose axes intersect perpendicularly to realize a (gimbal) hip joint between the torso and third "shank" link (mass $m_3 = 0.52$ kg). The servomotor positions describe the configuration of the hip joint, which is defined by a roll angle ϕ_1 between the torso and thigh, and pitch angle ϕ_2 between the thigh and shank. The shank and fourth "foot" link (mass $m_4 = 0.23$ kg) compose the LEAP mechanism (see [5]), which is an actuated prismatic joint whose displacement is defined by a stroke length (ϕ_3). The IMU frame describes the configuration of the torso floating base, which is defined by a position vector $\mathbf{p} = [p_x, p_y, p_z]^T$ and quaternion vector $\mathbf{Q} = [q_w, q_i, q_j, q_k]^T$. The robot's configuration is fully defined by concatenating the configuration variables into the vector $\mathbf{q} = [p_x, p_y, p_z, q_w, q_i, q_j, q_k, \phi_1, \phi_2, \phi_3]^T$. We selected a voice coil model roughly by maximizing work density and stroke (maximum coil displacement) while minimizing price, and selected stock compression springs

with spring constants that roughly maximize steady-state hopping height in a simulated 1D environment (see [5] for details).

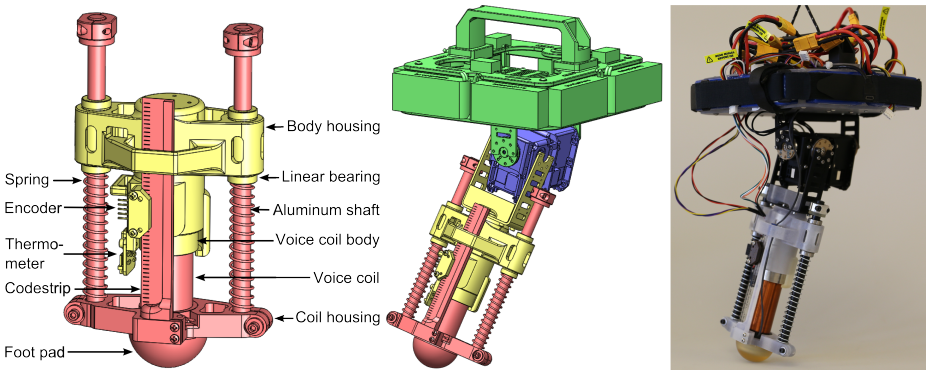

Fig. 1. (Left) CAD model of LEAP mechanism with component callouts. (Middle) CAD model of proposed hopping robot. (Right) Photo of assembled hopping robot.

2.2 Center of Mass Velocity Estimation

To perform proper foot placement, we must accurately estimate the horizontal components of the center of mass (COM) velocity of the entire robot. To do so, we first estimate the velocity of the IMU frame $\mathbf{v}_t^{imu}$ at time step t, then add the relative velocity of the COM with respect to the IMU. We define the predicted velocity $\mathbf{v}_t^p$ of the IMU by summing velocity increments with respect to the IMU velocity estimate of the previous time step $\mathbf{v}_{t-1}^{imu}$ as

$$\mathbf{v}_t^p = \mathbf{v}_{t-1}^{imu} + \mathbf{\Delta v}_t \tag{1}$$

where $\mathbf{\Delta v}_t$ are velocity increment measurements output from the IMU at time t. We define the update velocity $\mathbf{v}_t^u$ of the IMU by differentiating the forward kinematics of the IMU during stance. Specifically, we treat the IMU as an end-effector of a rooted open-link kinematic chain by assuming the tip of the foot maintains static contact with the ground through a spherical joint. Solving the forward kinematics gives the position of the IMU as a function of IMU orientation and joint angles, concatenated as $\mathbf{y} = [q_w, q_i, q_j, q_k, \phi_1, \phi_2, \phi_3]^T$, such that the IMU position with respect to the foot is a function of the sensor variables, $\mathbf{p}_t = f(\mathbf{y}_t)$. The update velocity is found by differentiating the IMU position,

$$\mathbf{v}_t^u = \frac{d}{dt}(\mathbf{p}_t) = \frac{\partial \mathbf{p}_t}{\partial \mathbf{y}}\dot{\mathbf{y}} = \mathbf{J}_1\dot{\mathbf{y}} \tag{2}$$

where $\mathbf{J}_1 = \mathbf{J}_1(\mathbf{y}_t)$ is a standard manipulator Jacobian. We estimate the IMU velocity as a weighted average of the update and predict velocities,

$$\mathbf{v}_t^{imu} = K_f\mathbf{v}_t^u + (1 - K_f)\mathbf{v}_t^p \tag{3}$$

where K_f is the IMU velocity filter gain. We only estimate velocity during stance phase, when the foot is in contact with the ground. During flight phase, we assume the horizontal (x and y) components of the COM velocity remain constant, and thus do not need to estimate IMU velocity.

To find the relative velocity of the COM, we first find the relative position of the COM with respect to the IMU, $\mathbf{r}^{com/imu} = \frac{1}{M}(m_1\mathbf{r}^{1/imu} + m_2\mathbf{r}^{2/imu} + m_3\mathbf{r}^{3/imu} + m_4\mathbf{r}^{4/imu})$ where $M = m_1 + m_2 + m_3 + m_4$ is the total mass of the robot, and $\mathbf{r}^{i/imu}$ is the relative position of the COM of link i with respect to the IMU. Noting that $\mathbf{r}^{com/imu} = \mathbf{r}^{com/imu}(\mathbf{y})$, we differentiate it to find the relative COM velocity,

$$\mathbf{v}^{com/imu} = \frac{d}{dt}(\mathbf{r}^{com/imu}) = \frac{\partial \mathbf{r}^{com/imu}}{\partial \mathbf{y}}\dot{\mathbf{y}} = \mathbf{J}_2\dot{\mathbf{y}} \tag{4}$$

Adding this result to the IMU velocity, we estimate the COM velocity at timestep t as

$$\mathbf{v}_t^{com} = \mathbf{v}_t^{imu} + \mathbf{v}_t^{com/imu} \tag{5}$$

2.3 Modified Raibert Controller

Raibert's 3D hopping controller [2] is intuitive, and comprises three independent components: 1) fixed thrust control during stance, 2) torso orientation control during stance, and 3) foot placement control during flight. Thrust and orientation control are active when contact is detected, which occurs when stroke falls below a set threshold. Foot placement control is active when contact is not detected. First, to provide a fixed thrust during stance, we implement a bang-bang controller, which works to inject energy into the system. The controller commands maximum negative voltage to the VCA during leg compression, and maximum positive voltage to the VCA during extension, which ensures the VCA always performs net positive work. Second, the controller servos global pitch and roll angles of the torso (θ^P and θ^R respectively) to zero during stance using proportional control, such that the commanded pitch and roll joint torques are $f_1^{st} = K_{p1}^{st}\theta_R$ and $f_2^{st} = K_{p2}^{st}\theta_P$, respectively, where K_{p1}^{st} and K_{p2}^{st} are proportional gains. We don't use a derivative term since the D-gain of the built-in PID control of the Dynamixel servomotors has no effect on the motion. Third, the foot placement controller calculates the desired foot placement with respect to the center of mass, which is tracked by an inverse kinematics (IK) controller.

Specifically, the desired horizontal foot placement with respect to the COM, $\mathbf{r}_{x,y}^{f/com,d} = [x^{f/com,d}, y^{f/com,d}]^T$ is a function of the expected stance time T_{st}, horizontal components of the COM velocity $\mathbf{v}_{x,y}^{com} = [\dot{x}^{com}, \dot{y}^{com}]^T$, and desired horizontal velocity $\mathbf{v}_{x,y}^{com,d} = [\dot{x}^{com,d}, \dot{y}^{com,d}]^T$,

$$x^{f/com,d} = \frac{\dot{x}^{com}T_{st}}{2} + K_x(\dot{x}^{com} - \dot{x}^{com,d}) \tag{6}$$

$$y^{f/com,d} = \frac{\dot{y}^{com}T_{st}}{2} + K_y(\dot{y}^{com} - \dot{y}^{com,d}) \tag{7}$$

where K_x and K_y are the foot placement gains. Both hip servomotors are used to track the desired foot placement using an IK tracker, which is derived as follows. The horizontal position of the foot with respect to the COM, $\mathbf{r}_{x,y}^{f/com} = [x^{f/com}, y^{f/com}]^T$ is differentiated as

$$\dot{\mathbf{r}}_{x,y}^{f/com} = \begin{bmatrix} \dot{x}^{f/com} \\ \dot{y}^{f/com} \end{bmatrix} = \frac{\partial \mathbf{r}_{x,y}^{f/com}}{\partial \mathbf{y}} \dot{\mathbf{y}} = \mathbf{J}_3 \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix} + \mathbf{J}_4 \begin{bmatrix} \dot{\mathbf{Q}} \\ \dot{\phi}_3 \end{bmatrix} \tag{8}$$

where $\mathbf{J}_3 = \frac{\partial \mathbf{r}_{x,y}^{f/com}}{\partial \mathbf{y}_1}$ and $\mathbf{y}_1 = \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix}$. We can solve for $\begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix}$ as

$$\begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix} = \mathbf{J}_3^{-1} \left[\begin{bmatrix} \dot{x}^{f/com} \\ \dot{y}^{f/com} \end{bmatrix} - \mathbf{J}_4 \begin{bmatrix} \dot{\mathbf{Q}} \\ \dot{\phi}_3 \end{bmatrix} \right] \tag{9}$$

We can use (9), assuming $\dot{\mathbf{Q}}$ and ϕ_3 are zero to simplify the controller and reduce compution time, to derive desired hip joint positions, $[\phi_1^d, \phi_2^d]^T$, given the foot placement tracking errors, $\Delta x^{fp} = x^{f/com,d} - x^{f/com}$ and $\Delta y^{fp} = y^{f/com,d} - y^{f/com}$, as

$$\begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix}^d = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} + \eta \mathbf{J}_3^{-1} \begin{bmatrix} \Delta x^{fp} \\ \Delta y^{fp} \end{bmatrix} \tag{10}$$

where η is an IK tracking gain (i.e. step size). The desired hip joint positions are tracked with a proportional servo $f_1^{fp} = K_{p1}^{fp}(\phi_1^d - \phi_1)$ and $f_2^{fp} = K_{p2}^{fp}(\phi_2^d - \phi_2)$, where K_{p1}^{fp} and K_{p2}^{fp} are proportional gains. The Jacobians $\mathbf{J}_1$, $\mathbf{J}_2$, and $\mathbf{J}_3$ are derived using Matlab Symbolic Toolbox.

3 Results

3.1 Simulation

We first developed a simulation to test, tune, and debug our state estimator and controller before implementing on hardware. The simulation was created using Matlab/Simulink/SimMechanics/SimScape software. The LEAP actuator and ground contact models were reused from our previous work [5]. We model Coulomb and viscous friction at the hip joints as well as torque-velocity constraints. We run the controller at 100 Hz (same as on hardware) and simulate the system with a variable time-step solver (ode45, relative error tolerance: 1e-4, absolute error tolerance 1e-5). The geometric and inertial parameters of the links were estimated from CAD (mass was measured on a scale). The spring constant of the LEAP mechanism was roughly optimized in a 1D simulation given our system mass (see [5]), and two stock springs of similar stiffness (2060 N/m total) were selected and installed in our leg. We avoid reporting all simulation parameters here due to space constraints.

Sensor signals include joint positions and velocities, IMU orientation quaternion and quaternion derivative (approximated discretely), and IMU acceleration.

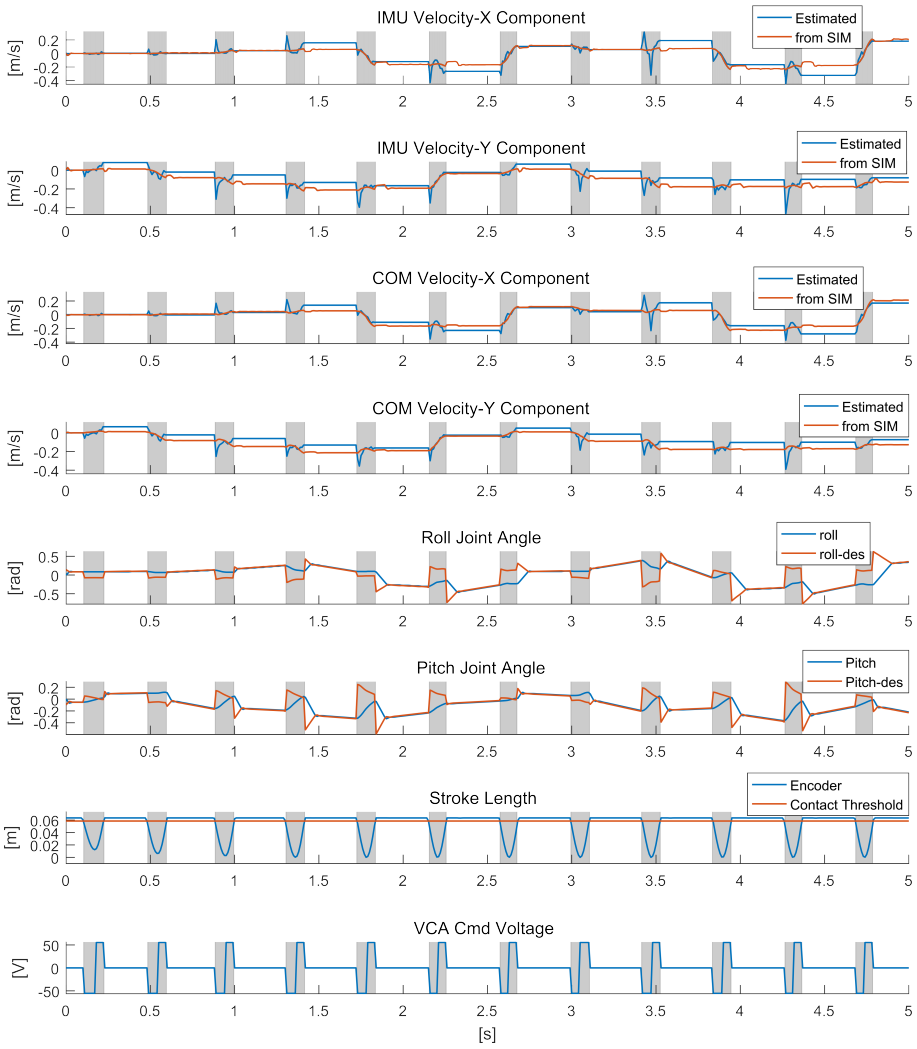

Fig. 2. Plots of simulated hopping data for 5 seconds. Gray represents positive contact detection, occurring when stroke drops below the contact threshold. From top to bottom are plots of: 1) X-component of IMU velocity, estimated and actual. All the velocity estimates are held constant during flight. 2) Y-component of IMU velocity, estimated and actual. 3) X component of COM velocity estimate, estimated and actual. 4) X component of COM velocity estimate, estimated and actual. 5) Roll joint angle and desired angle. 6) Pitch joint angle and desired angle. 7) Displacement of voice coil (stroke) and contact threshold. 8) Commanded voice coil voltage.

These signals are quantized and discretized to roughly match our hardware, and we inject Gaussian noise based on data reported by the sensor datasheets. We use the covariance matrix adaptation evolution strategy (CMA-ES) [6] to optimize the control parameters for maximum hopping time before fall. We present 5 seconds of simulated hopping data from the optimized controller in Figure 2. The resulting optimized controller could hop for roughly 60 seconds before falling.

3.2 Physical Experiment

In our physical setup, the robot was attached to a slack safety harness and maintained a serial connection to the host computer for data logging. No power was transmitted over these mechanical and data connections. A motion capture (MOCAP) system (Vicon MX series, 16 cameras, 120 fps) is used to record "ground truth" velocity estimates. MOCAP markers are placed in known locations on the torso (m_1) and foot (m_4) links. The MOCAP system provides position-time data for these markers, from which we can calculate positions and velocities of the tip of the foot and IMU. Here we present data from a hopping experiment for a single trial. For the same trial, we present plots (Fig. 3) of estimated and measured (from MOCAP) global-frame IMU velocity, estimated COM velocity, measured and desired pitch and roll servo angles (ϕ_1 and ϕ_2), stroke length (ϕ_3), commanded voice coil voltage, and contact detection. We captured data until an operator intervened to prevent an imminent fall. We recorded data for 50 trials, and found an average and maximum hopping time of approx. 3.3 and 6.5 seconds, respectively. A video camera captured snapshots of a separate experiment (Fig. 4) at 29 fps (shown every other frame).

4 Discussion

4.1 Velocity Estimation

Accurate velocity estimation is critical to the performance of our controller. Our current estimator performs rather poorly, as the IMU velocity plots in Figure 3 show. There are a couple reasons for this. First, our model assumes static foot contact with the ground, despite the existence of slip and deformation of our rubber foot. Second, our sensors are imperfect, and contain quantization error and noise, among other inaccuracies.

We could improve velocity estimation with better and/or redundant sensors, or with a better model. Our current model is purely kinematic. A better approach might comprise an unscented Kalman filter, using a forward dynamics model. Another approach might use a learning model to approximate our ground truth MOCAP velocity data.

4.2 Hopping Controller

Our current controller might be improved by relaxing the static torso/static q_3 joint assumptions. Alternatively, a momentum-based controller, which takes into

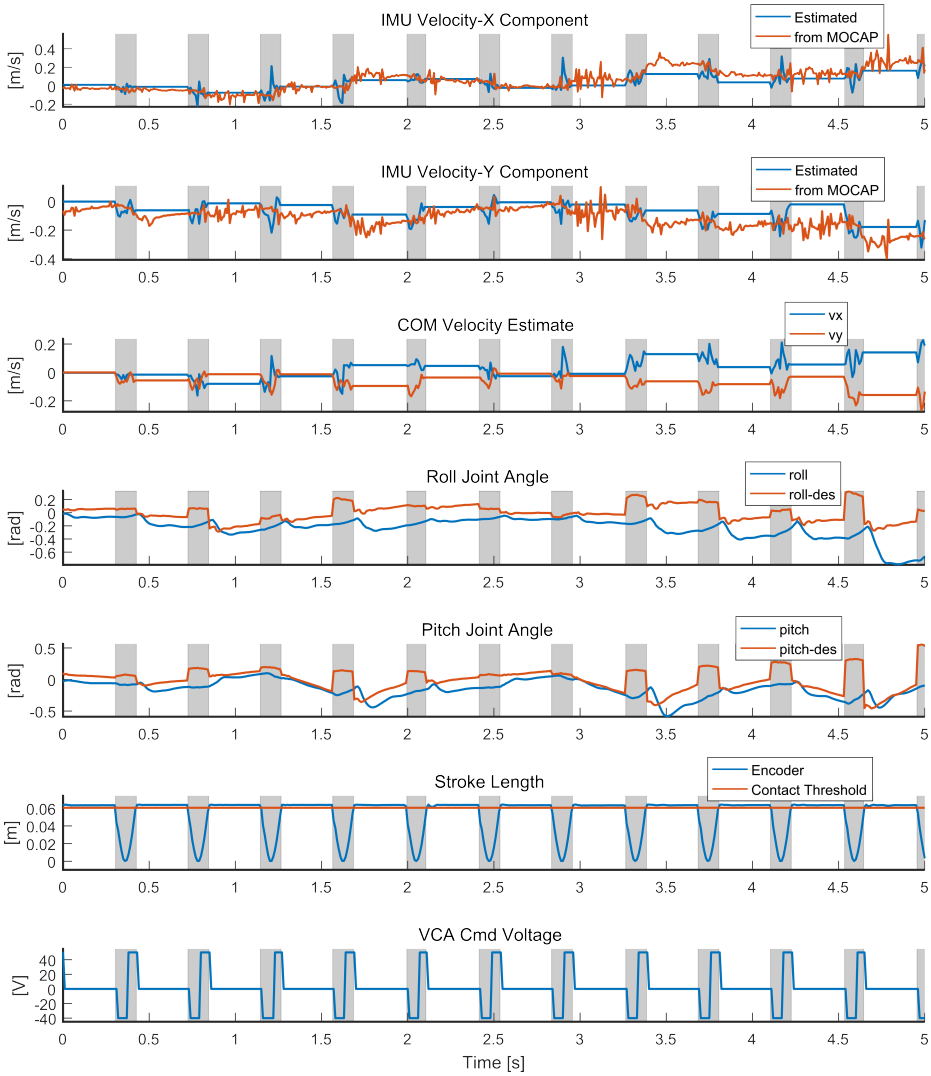

Fig. 3. Plots of hopping data for a single trial. Gray represents positive contact detection, occurring when stroke drops below the contact threshold. From top to bottom are plots of: 1) X-component of IMU velocity, estimated and from motion capture (MOCAP) data. The estimate is held constant during flight. 2) Y-component of IMU velocity, estimated and from MOCAP data. 3) horizontal (x and y) components of COM velocity estimate. 4) Roll joint angle and desired angle. 5) Pitch joint angle and desired angle. 6) Displacement of voice coil (stroke) and contact threshold. 7) Commanded voice coil voltage.

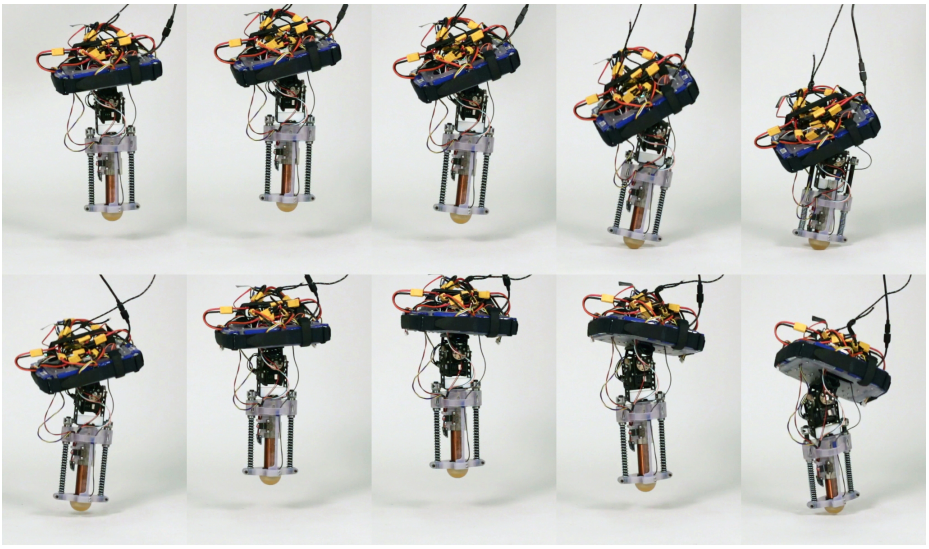

Fig. 4. Snapshots of a hopping experiment trial. Video was recorded at 29 fps, and presented every other frame. The shots are sequenced left to right, top to bottom.

account the robot's constant angular momentum during flight, might compensate for the large and unwieldy torso-to-leg inertia ratio. The bang-bang thrust controller is inefficient, and causes the voice coil to overheat if used for extended periods of time. A more efficient controller, which exploits velocity-efficiency characteristic of the VCA, might achieve the same performance with less energy consumption.

4.3 Hardware

There are many hardware improvements that would likely improve hopping performance of our robot. First, the most pressing issue is the relatively large leg inertia, which limits foot-placement control authority during flight. Increasing torso inertia would be the simplest way to overcome this issue, but would increase the load on the q_3 actuator. Leg inertia has already been minimized, and would be difficult to further decrease. Second, the servomotors of the hip joint might be replaced by direct drive motors. Our current servomotors cannot perform accurate torque control, lack an effective derivative term in their PID control loop, and contain gearbox backlash. Direct-drive motors may perform better. The authors note the Delta Hopper robot (mentioned briefly in [7]) as an alternative to our design. Its single leg is a parallel 3-dof mechanism with large-radius direct drive motors at the hip. Such a design allows for torque-based control methods, decreases leg inertia, and would likely be more successful in continuous hopping. Third, computational power could be improved by using a more powerful microcontroller, or a mini computer (e.g. Odroid or Raspberry Pi), which would

enable more complex estimation and control algorithms, as mentioned previously. Finally, active cooling of the voice coil, with a fan or other cooling system, may lessen the problem of overheating.

4.4 Future Work and Conclusions

In this paper, the design and control of an untethered singled-legged hopping robot was presented. Our simulations and experiments have shown that the LEAP mechanism can be employed as an actuator for the robot. It can likely be used in other robot designs as well, especially those that aren't as energetically demanding. While we fell short of our goal of continuous, indefinite hopping, we showed that such a gait is possible for an untethered robot for short periods of time. In the future, we plan to implement many of the previously proposed changes. Furthermore, we plan to redesign the LEAP mechanism to be more modular and compact for use in a multi-legged robot.

References

1. F. Hardarson, *Locomotion for difficult terrain.* Citeseer, 1998.
2. M. H. Raibert, H. B. Brown, 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.
3. G. Zeglin and B. Brown, "Control of a bow leg hopping robot," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 1. IEEE, 1998, pp. 793–798.
4. A. Sayyad, B. Seth, and P. Seshu, "Single-legged hopping robotics research-a review," *Robotica*, vol. 25, no. 05, pp. 587–613, 2007.
5. Z. Batts, J. Kim, and K. Yamane, "Design of a hopping mechanism using a voice coil actuator: Linear elastic actuator in parallel (leap)," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, May 2016.
6. N. Hansen, "The cma evolution strategy: A comparing review," *Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms. Springer*, p. 75.102, 2006.
7. G. Kenneally, A. De, and D. Koditschek, "Design principles for a family of direct-drive legged robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 900–907, 2016.