Precision Jumping Limits from Flight-phase Control in Salto-1P

Justin K. Yim[1] and Ronald S. Fearing[1]

Abstract— **We developed a deadbeat foot placement hopping controller for an untethered monopedal robot, Salto-1P. The controller uses a third order Taylor series approximation to an offline dynamic model and performs well on the physical platform. The robot demonstrated precise foot placement even on trajectories with aggressive changes in speed, direction, and height: in a random walk, its error standard deviation was 0.10 m. We establish how foot placement precision is tightly limited by attitude control accuracy, requiring attitude error less than 0.7 degrees for some tasks. We also show how foot placement precision degrades linearly as hopping height increases. These precision results apply to the large class of controllers that prescribe touchdown angle to control running velocity.**

I. Introduction

Legged robots offer unique capabilities and challenges when compared to other mobile platforms. Legged platforms can move over rough or even discontinuous surfaces by taking advantage of isolated footholds inaccessible to wheeled or tracked platforms. By jumping, a legged platform may traverse terrain in which useable footholds may be distributed at distances larger than the bodylength of the platform. Unlike heavier-than-air aerial platforms that must produce lift, legged platforms are not subject to ground effect and do not produce downwash that might kick up dust or blow away light objects.

Campana and Laumond developed a ballistic motion planning algorithm for saltatorial locomotion that respects takeoff velocity and friction cone limits in complicated terrain [3]. The planner is able to find trajectories across disparate stepping stones or even up channels between vertical walls. Using trajectories like these, jumping robots could traverse terrain that could be extremely difficult for other kinds of platforms. However, following these trajectories in complex terrain requires very accurate foot placement to avoid missing a foothold. A robot must be able to accurately redirect its ballistic trajectory towards the next foothold using only one stance phase. This single-stance control is akin to deadbeat control of a discrete-time system and we will likewise call a controller with this ability a deadbeat foot placement hopping controller. This work aims to improve the foot placement precision of a small jumping robot by developing deadbeat foot placement hopping control and examines the effect of attitude error on foot placement precision.

This material is based upon work supported by the United States Army Research Laboratory under the Micro Autonomous Science and Technology Collaborative Technology Alliance and the Army Research Office Grant No. W911NF-18-1-0038.

[1]J.K. Yim, and R.S. Fearing are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA yim@eecs.berkeley.edu, ronf@eecs.berkeley.edu

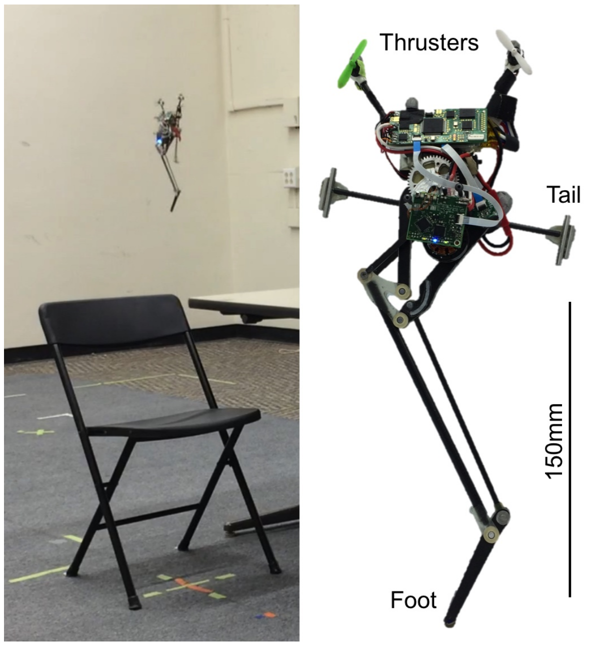


Fig. 1: Robot platform Salto-1P jumps on a chair.

One particularly useful model of legged locomotion both in robotics and biology is the Spring Loaded Inverted Pendulum (SLIP) [2]. In his seminal work on dynamic legged systems [14], Raibert laid out simple control policies that balance a SLIP-like monopedal hopping robot. Significant work has built upon this controller and proposed related control schemes like [19].

However, the SLIP dynamics cannot be integrated to produce a closed-form solution except for particular nonlinear spring forces without gravity. In [17] a closed-form integrable SLIP-like model is used to derive deadbeat control of hopping apex state. Many approximations to the SLIP dynamics like [18], [7], and [16] have also been proposed to sidestep the non-integrability.

Numerical simulation of the hopper dynamics is another approach. In [15], a lookup table and a polynomial approximation to the table selected control inputs. In [4], [22], and [20] deadbeat hopping of SLIP-like models is examined analytically and in simulation. In addition, sensitivity of a deadbeat controller to model error and control input error was examined in [22], finding that more aggressive turns produced larger errors. Without using SLIP dynamics, [1] and [6] investigated sequencing deadbeat control of monopedal jumping platforms. Clocked deabeat control of vertical hopping without horizontal dynamics was derived in [5]. A reachability approach controlled the next apex state during the stance phase in [12].

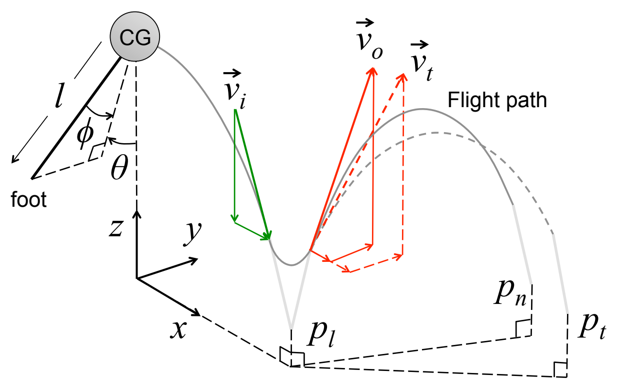


Fig. 2: Reference frames and variables.

Several works have demonstrated precise foot placement in physical experiments. Hodgins demonstrated precise foot placement to enable stair climbing with a boom-mounted planar robot by controlling the step length through variation in the horizontal velocity, flight time, and stance time [11]. In [21] partial feedback linearization demonstrated high precision foot placement also on a boom-mounted robot.

A. *Physical Platform*

Salto-1P shown in Fig. 1 is a small monopedal jumping robot developed in [8] [9] [10] [13]. It uses a series-elastic power modulating leg to power rapid high jumps. In the air, a balanced inertial tail controls the robot's pitch and two small laterally directed propellers atop the robot control its roll and yaw. The robot carries a 6-axis inertial measurement unit (IMU), encoders to measure the position of its leg, leg motor, and tail, memory for logging experimental data, and an XBee radio for communication with the ground station.

II. Methods

A. *Approach*

In developing a deadbeat foot placement hopping controller, we consider a SLIP-like hopping model shown in Fig. 2. Jumping motion can be divided into alternating stance phases (stances) and flight phases (flights). Touchdown is the transition between flight and stance when the foot strikes the ground at a foot placement point. Takeoff is the transition between stance and flight when the foot leaves the ground.

During flight, a jumping robot has no control over the motion of its center of gravity (CG) without specialized means to apply large forces in the air. Its CG trajectory in flight (flight path) is a ballistic parabola set by its previous foot placement and velocity at takeoff. The robot has only a little control over its next foot placement and velocity at touchdown through reorienting its foot relative to its CG.

To reach a desired foothold, the robot must set its velocity at takeoff to aim its flight path towards the foothold. Takeoff velocity can be changed either by control action during stance or by changing stance initial conditions like leg angles at the previous flight's touchdown. [10] demonstrated that this latter technique was useful for robots like Salto-1P that have stances too fast for effective closed-loop control. Thus to achieve deadbeat foot placement hopping control, a robot can set touchdown conditions from its previous flight to aim its next flight path towards the desired foothold.

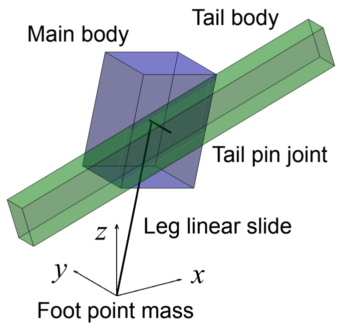


Fig. 3: Simulation model.

We split the deadbeat foot placement hopping control into two parts. The first part, a velocity planner, determines what takeoff velocity $\vec{v}_t$ will take the robot from its upcoming foot placement p_l and place its next foot placement p_n on the desired foothold at p_t. Since a family of ballistic parabolas pass through two points, the velocity planner has one free variable that it can set by selecting a parameter like vertical velocity at takeoff or apex height.

The second part, a velocity controller, determines appropriate touchdown conditions so that the stance phase will start from the touchdown velocity $\vec{v}_i$ and end at the takeoff velocity $\vec{v}_o$ so that $\vec{v}_o = \vec{v}_t$ computed by the velocity planner. Salto-1P's touchdown is parameterized by the roll angle (ϕ) pitch angle (θ), and leg length (l). Touchdown leg length changes the impulse delivered on the ground: a shorter initial length provides a larger net impulse. We parameterize the robot's attitude with ZXY Euler angles, but ignore the yaw angle (ψ) since the SLIP-like model is symmetric about Z.

We assume the robot's leg length is small relative to its jump height as is the case for Salto-1P. Therefore, the effects of ϕ, θ, and l on p_l and $\vec{v}_i$ can be ignored so that the velocity controller's actions do not affect the velocity planner.

B. *Simulation Model*

In order to predict $\vec{v}_o$ resulting from certain touchdown conditions, we simulated stance in a custom Matlab rigid body simulation matched to the physical parameters of Salto-1P. This model consists of three bodies shown in Fig. 3: a rigid body for the main body of the robot, a rigid body for the tail connected to the main body by a pin joint, and a point mass on a linear sliding constraint relative to the main body to serve as the foot and approximate the mass and inertia of the leg.

Leg force is computed with a leg friction model and nonlinear spring model from [9], the kinematic relationship between the crank angle and the foot extension produced by the 8-bar linkage, and a DC motor model.

The touchdown transition is modeled as an inelastic collision between the point mass foot and the rigid ground. Takeoff is another inelastic collision in which the stance leg length reaches its maximum extension and momentum transfers from the main body to the foot.

Salto-1P's moment of inertia about its lateral and longitudinal axes are both approximately 130×10^{-6} kg m^2. Since the robot weighs 0.103 kg and its CG is 0.10 m above its foot with the leg fully retracted, the robot's moment of inertia about its foot is dominated by the CG distance from the foot and is not significantly changed by the robot body's heading. Since the robot's foot moves along a straight line coincident with the CG and its moment of inertia is nearly the same at all headings, the robot's stance phase is insensitive to heading and its touchdown yaw angle can be neglected.

Furthermore, since the balanced inertial tail's angular velocity is kept low by braking during stance phase, the tail's angular momentum is small compared to the angular momentum due to the motion of the robot's CG. As with the robot's yaw heading, the tail angle and angular velocity are also neglected. With the above assumptions, the robot's behavior is similar to a SLIP-like point mass with a motor-controlled leg force and we can reduce the number of parameters that must be searched in simulation.

Adopting the convention of [22], $\vec{v}_i$ is measured aligned with the horizontal bearing of the robot's velocity. In this touchdown velocity-aligned frame, $\vec{v}_i$ has only two elements: longitudinal (v_{ix}) and vertical (v_{iz}). $\vec{v}_o$ has three elements: longitudinal (v_{ox}), lateral (v_{oy}), and vertical (v_{oz}). This simulation approximates a map from $\vec{v}_i$ to $\vec{v}_o$ controlled by ϕ, θ, and l, $\vec{v}_o = f_s(\vec{v}_i, \phi, \theta, l)$.

C. Velocity Controller

The velocity controller is composed of three functions whose inputs are the touchdown velocity and desired takeoff velocity $\phi = f_\phi(\vec{v}_i, \vec{v}_t)$, $\theta = f_\theta(\vec{v}_i, \vec{v}_t)$, $l = f_l(\vec{v}_i, \vec{v}_t)$.

From the SLIP-like model's rotational symmetry about the Z-axis and the coordinate frame selection, there are several symmetries in the velocity control functions:

- f_θ is odd in x ($f_\theta(v_{ix}, v_{iz}, v_{tx}, v_{ty}, v_{tz}) = -f_\theta(-v_{ix}, v_{iz}, -v_{tx}, v_{ty}, v_{tz})$) and even in v_{ty}.
- f_ϕ is odd in v_{ty} and even in x.
- f_l is even in x and v_{ty}.

We ran simulations of stance for a grid of 16,170 touchdown conditions. The baseline θ resolution was 0.05 rads and the ϕ resolution was 0.03 rads but these became finer and confined to a smaller deviation for larger v_{iz} since large deflections would have caused slipping. The horizontal and vertical velocities were swept at a constant step of 0.4 m/s from -2.2 to 2.2 m/s and -2.0 to -4.4 m/s respectively. l varied from 0.226 m to 0.25 m at a constant step of 0.004 m. Simulations for which the main body impacted the ground, reached velocities outside the range of the touchdown velocities, and those for which the foot would slip on a flat horizontal surface with coefficient of friction 1 were discarded.

Similarly to [15], we use a polynomial approximation for the velocity controller functions. We fit five-dimensional third-order Taylor series approximations for f_θ, f_ϕ, and f_l centered at the point $v_{ix} = v_{ox} = 0$ m/s, $-v_{iz} = v_{oz} = 3.3$ m/s and $v_{oy} = 0$ m/s using the simulation data. This point corresponds to hopping in place with an apex height of 0.55 m. Due to the symmetry constraints outlined above, there are 31 non-zero coefficients in total: five linear, eleven quadratic, and fifteen cubic terms. The final velocity controller f_θ function is given below as an example:

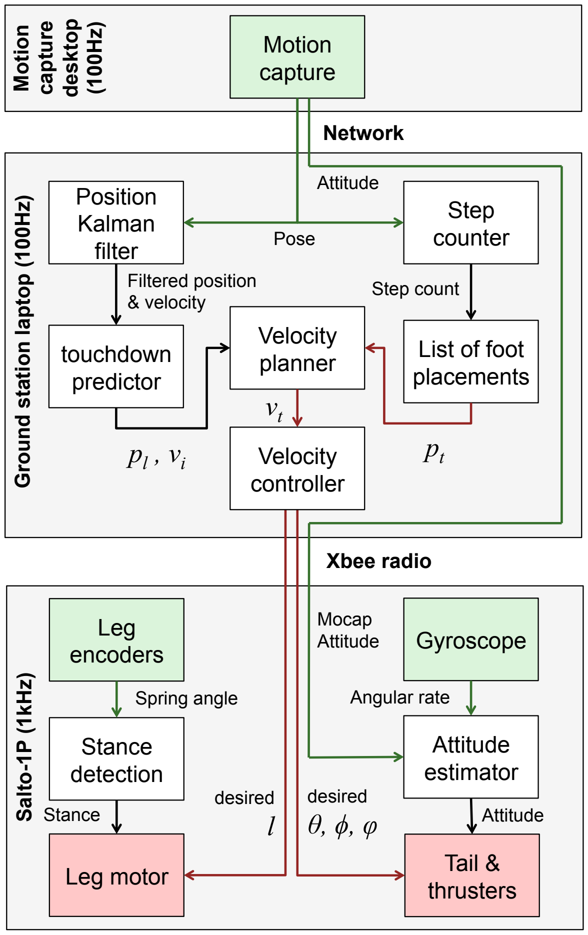


Fig. 4: Block diagram of experimental system, deadbeat velocity planning, and deadbeat velocity control. The Raibert controller replaced the "Velocity controller" block during comparison experiments.

$$\begin{aligned} f_\theta(v_{ix}, v_{iz}, &v_{tx}, v_{ty}, v_{tz}) = -0.2071 v_{ix} + 0.078 v_{tx} \\ &- 0.0385 v_{ix} v_{iz} + 0.0196 v_{tx} v_{iz} \\ &- 0.0052 v_{ix} v_{tz} - 0.04 v_{tx} v_{tz} \\ &+ 0.003 v_{ix}^3 - 0.036 v_{tx}^3 \\ &- 0.0024 v_{ix} v_{ty}^2 - 0.0046 v_{tx} v_{ty}^2 \end{aligned} \tag{1}$$

After the deadbeat velocity controller computes the desired θ and ϕ angles, they are rotated out of the touchdown velocity-aligned frame to the ZXY Euler angle frame.

D. Velocity Planner

The flight estimation and velocity planner determine the desired takeoff velocity v_t at the end of the upcoming stance so that the robot next lands at p_t. For convenience, the height of p_l is assumed to be the height of the previous desired foot placement. p_l is then predicted as the point at which the flight path reaches this height.

The velocity planner computes the desired flight time t_f from v_{tz} and the heights of the foot placements p_{tz}, and p_{lz}.

$$t_f = \frac{v_{tz}}{g} + \frac{\sqrt{v_{tz}^2 - 2g(p_{tz} - p_{lz})}}{g}$$

v_{tx} and v_{ty} are then the horizontal displacement of p_t from p_l divided by the desired flight time. These velocities are rotated into the touchdown velocity-aligned frame.

$$v_{tx} = \frac{p_{tx} - p_{lx}}{t_f},\ v_{ty} = \frac{p_{ty} - p_{ly}}{t_f}$$

Note that we ignore the foot deflection away from the CG trajectory as stated earlier. In the results we show that this error is small in comparison to other errors.

E. Physical experiments

The Salto-1P robot in Fig. 1 was used for the following experiments. It can jump 0.90 m high with a vertical jumping agility of 1.36 m/s [8]. The robot is 0.17 m tall with leg retracted and 0.32 m tall with leg fully extended.

Optitrack Prime 13 cameras and Motive software track retroreflective markers on the robot. As shown in Fig. 4, motion capture data stream at 100 Hz to a ground station laptop running ROS. The ground station uses a Kalman filter to estimate the robot's linear position and velocity in flight and uses these to predict p_l. Acceleration is estimated by double-differentiation of position. Touchdown is detected when acceleration exceeds a threshold. The Kalman filter is disabled in stance and reinitialized shortly after takeoff.

For each experiment, we specify a list of desired foot placements to define the desired hopping trajectory. Each desired foot placement has five parameters: x, y, and z position of p_t, desired yaw angle, and desired vertical velocity at takeoff.

The step count increments by one upon touchdown. The next desired foot placement in the list of desired foot placements is then selected as the new p_t. The velocity planner computes v_t from p_l and p_t. The velocity controller computes the desired ϕ, θ, and l.

The groundstation computer sends the attitude measured by the motion capture system, desired touchdown attitudes, and desired leg length to the robot via the XBee radio.

The robot's onboard microcontroller runs asynchronously from the ground station. Three 1 kHz proportional derivative (PD) controllers on yaw, roll, and pitch angles activate the thrusters and tail in flight to stabilize the robot's attitude to the desired touchdown leg angles. Another 1 kHz PD controller retracts the leg to the desired length. The microcontroller integrates rate gyroscope readings from the onboard IMU to estimate attitude. It fuses this estimate with the received motion capture attitude measurement to prevent drift due to accumulation of integrated gyroscope error.

The robot also detects its own touchdown by measuring the deflection of the series elastic spring. If this exceeds a threshold, the robot determines that it has hit the ground and it extends its leg. Takeoff is detected when the spring deflection reaches zero or the leg reaches its full extension.

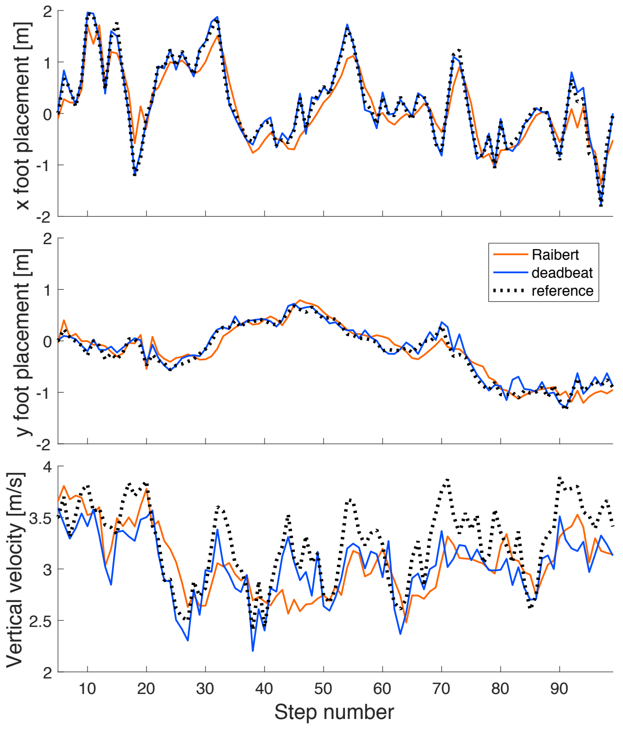


Fig. 5: Experimental comparison of deadbeat velocity controller and Raibert hopper controller foot placement error.

After each experiment, we manually synchronize recorded groundstation and robot data by aligning motion capture measurements of robot height with onboard measurements of leg deflection during stance.

III. Results

A. Foot Placement Accuracy

To evaluate foot placement accuracy, the robot was commanded to follow a series of 95 foot placements in a random walk on flat ground with a yaw heading of zero and random vertical velocity changes. The commanded and achieved foot placement trajectories are shown in Fig. 5. Longitudinal (x) displacements ranged up to 1.3 m and lateral (y) displacements ranged up to 0.36 m corresponding to longitudinal speeds up to 1.68 m/s and lateral speeds up to 0.43 m/s. The longest jumps were over four times the robot's body length. The largest change in horizontal velocity in one stance was 2.93 m/s. The longitudinal jumps are larger than the lateral because Salto-1P's inertial tail has greater control authority than its lateral thrusters. Horizontal jump direction angle change covered all angles from -180 degrees to +180 degrees. Desired vertical velocity at takeoff changed by up to 0.5 m/s higher or lower on each jump.

This experiment compared the performance of the deadbeat velocity controller and an optimally tuned Raibert controller. This Raibert controller replaced the "Velocity controller" block in Fig. 4. Since the Raibert horizontal velocity controller is parameterized only by one feedback gain, this parameter was optimized by hand tuning. Thrust control was selected by looking up l based on v_{tz} in a linearly

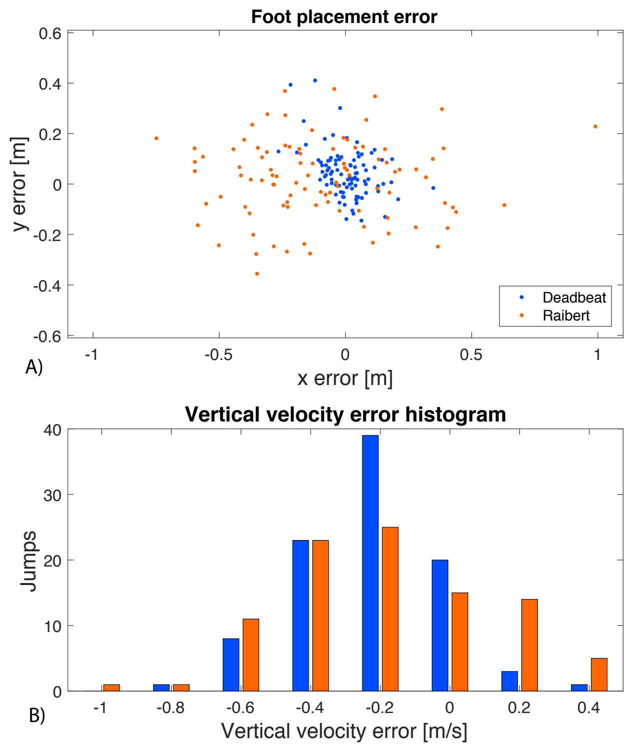


Fig. 6: Experimental deadbeat velocity controller and Raibert hopper controller foot placement error and vertical velocity error.

Longitudinal Error [m]	STD	95th percentile
Previous controller [10]	0.516	1.0
Optimal Raibert	0.306	0.6
Deadbeat	0.097	0.3

TABLE I: Experimental longitudinal foot placement accuracies for different controllers. For comparison, data from [10] were collected from a slightly less agressive run with only longitudinal jumps commanded.

interpolated lookup table of six experimentally measured pairs of leg length and steady-state hopping vertical velocity. In the canonical Raibert hopper controller, the stance duration is assumed to be the same as the last stance duration, but we used a second linearly interpolated lookup table of steady state hopping vertical velocity and stance duration since measuring stance duration was difficult for the motion capture system.

Fig. 6 compares the foot placement error $p_n - p_t$ and vertical velocity error $v_{oz} - v_{tz}$ of the Raibert controller and deadbeat velocity controller. The Raibert controller achieved a foot touchdown error standard deviation of 0.306 m longitudinally and 0.156 m laterally, while the deadbeat velocity controller achieved a foot touchdown error standard deviation of 0.092 m longitudinally and 0.097 m laterally.

Both controllers experienced decreased hopping height over time as the robot's battery voltage dropped during the run. The Raibert hopper controller was less able to follow quick changes from one v_{tz} to the next and rounded off the corners of v_{oz}. In comparison, though its v_{oz} had a steady offset below v_{tz} as the battery depleted, the deadbeat velocity controller better adjusted v_{oz} in a single stance phase. The Raibert hopper controller achieved a vertical velocity error standard deviation of 0.29 m/s while the deadbeat velocity controller achieved a vertical velocity error standard deviation of 0.20 m/s.

Foot placement precision is important since this defines the foothold size that the robot can reasonably hit. The Raibert controller placed 95 percent of its touchdowns within 0.6 m of the desired position, while the deadbeat velocity controller placed 95 percent of its touchdowns within 0.3 m of the desired position.

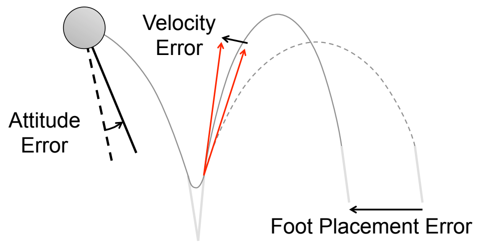


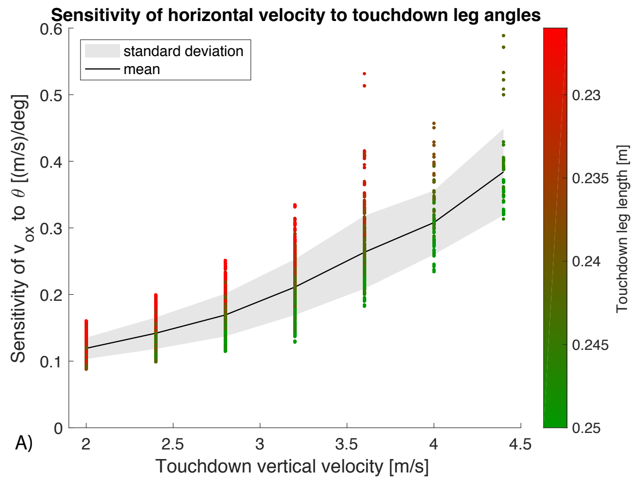
Fig. 7: Propagation of touchdown leg angle error to foot placement error.

B. Foot Placement Sensitivity to Touchdown Leg Angle

Like [14], [19], [22], [20] and others, our algorithm controls hopping by selecting touchdown leg angles and leg lengths during flight. Consequently, the foot placement accuracy is sensitive to the flight attitude control accuracy used to position the leg as shown in Fig. 7. To investigate how ϕ and θ accuracy would affect foot placement accuracy, we examined the sensitivity of $\vec{v}_o$ to the touchdown conditions. Using the simulation results that generated the deadbeat velocity controller, we calculated sensitivity by discrete differentiation of $\vec{v}_o$ with respect to $\vec{v}_i$, ϕ, θ, and l. This approximated the Jacobian of the stance velocity map f_s at each point in the grid of simulated initial conditions.

As expected, takeoff longitudinal velocity v_{ox} is highly sensitive to θ, and takeoff lateral velocity v_{oy} is highly sensitive to ϕ. v_{ox} is insensitive to ϕ and v_{oy} is insensitive to θ. These sensitivities are the coefficients by which leg angle errors will propagate into the takeoff horizontal velocities and foot placements.

Importantly, the sensitivity of takeoff horizontal velocity to touchdown leg angles increases as the robot's touchdown vertical velocity v_{iz} and takeoff vertical velocity v_{oz} become faster. Fig. 8A shows this sensitivity plotted against v_{iz}. The relationship is approximately linear with some upwards curvature. In steady forward hopping at $v_{ix} = v_{ox} = 1.0$ m/s and $v_{iz} = v_{oz} = 2.4$ m/s, v_{ox} varies at 0.13 m/s per degree of θ deflection and v_{oy} varies at -0.13 m/s per degree of ϕ deflection. For higher steady forward hopping at $v_{ix} = v_{ox} = 1.0$ m/s and $v_{iz} = v_{oz} = 4.0$ m/s, the horizontal velocity sensitivity magnitudes are 0.35 m/s per degree. The magnitudes of the v_{ox} sensitivity to θ and the v_{oy} sensitivity to ϕ remain very close to each other throughout the operating domain, so only the longitudinal direction is shown in Fig. 8A. Horizontal velocity sensitivity to touchdown leg angles also varies with touchdown leg length l but the effect is smaller than that of vertical velocity.



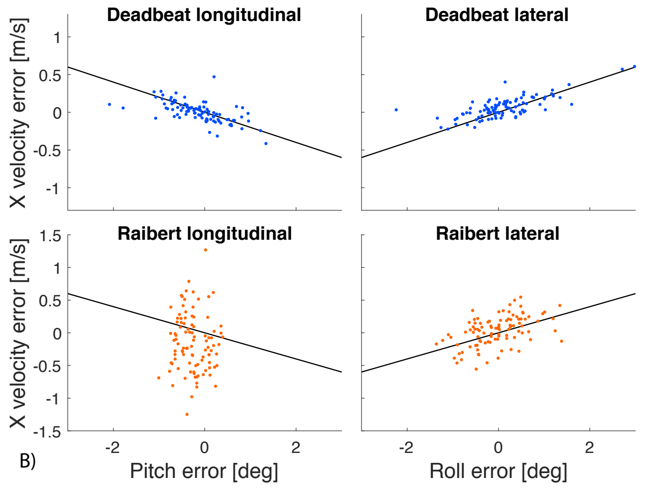


Fig. 8: A) Simulation data showing the sensitivity of v_{ox} to deviations in θ plotted against v_{iz} for the entire range of simulated initial conditions. Mean sensitivity is shown in black and one standard deviation is shown in grey.
B) Correlation between leg angle error and horizontal velocity error in the random walk experiments. The lines have slope 0.2, as predicted for this task by A).

To investigate how attitude control error propagates to foot placement error, we consider steady hopping over mostly flat terrain in which the vertical velocities change little from hop to hop. On horizontal surfaces, horizontal foot placement error is the product of the takeoff horizontal velocity error and the flight time plus errors due to flight time error. As shown in Fig. 8A, the horizontal velocity sensitivity to touchdown leg angles rises somewhat linearly with vertical velocity. Flight time rises approximately linearly with v_{zo} when the apex is significantly higher than the terrain height variation. Therefore the foot placement error rises approximately quadratically with vertical velocity for a given leg angle error. Apex height rises quadratically with increasing v_{zo}, thus foot placement error is approximately linearly related to apex height. Smaller jumps are significantly more precise than higher jumps for a given flight attitude control accuracy. From the numerical sensitivity data, to achieve a foot placement error lower than 0.15 m, the acceptable touchdown leg angle error is 2.8 degrees for $v_{zi} = v_{zo} = 2.4$ m/s but only 0.5 degrees for $v_{zi} = v_{zo} = 4.0$ m/s.

There is also a lower limit to effective hopping height since flight time decreases linearly with takeoff vertical velocity. If the flight time is too small, the angular velocity required for leg reorientation will exceed the flight attitude control authority and the touchdown attitude error will suffer. Therefore, there is a mid-to-low hop height for which the foot placement accuracy is optimal.

Attitude error propagation sets a foot placement precision limit for a given attitude control accuracy. During the deadbeat velocity controller's random walk task, the touchdown leg angle errors were approximately Gaussian with standard deviations of 0.58 degrees in θ and 0.75 degrees in ϕ. For this task, the numerical sensitivity data predict a takeoff horizontal velocity sensitivity to touchdown leg angle of about 0.2 (m/s)/deg, shown in Fig. 8B by black lines. This prediction matches the observed data relatively well. θ error was correlated to v_{ox} error with a coefficient of -0.61 and ϕ error was correlated to v_{oy} error with a coefficient of 0.75 as shown in Fig. 8B. This shows that a significant component of the foot placement errors can be attributed to the touchdown leg angle error and the deadbeat velocity controller operated relatively near the foot placement precision limit of the attitude control accuracy.

In comparison, the Raibert controller random walk θ error and v_{ox} error were correlated by a coefficient of -0.04 and the ϕ error and v_{oy} error were correlated by a coefficient of 0.49. The lower correlations, particularly in the longitudinal direction, are because the contribution of touchdown leg angle error to foot placement error is dwarfed by the contributions of other errors in the Raibert controller.

C. *Jumping on a chair and desk*

With the deadbeat foot placement hopping controller's superior accuracy, we demonstrated that the robot can jump up on top of a chair and desk as shown in Fig. 9. The jumping trajectory and foot placement errors are shown in Fig. 10. The plastic folding chair is 0.44 m tall at the front, 0.40 m tall at the back, 0.37 m deep, and 0.38 m wide. Jumping onto the chair requires foot placement accuracy better than the horizontal dimensions of the seat in order to avoid falling off. For $v_{iz} = v_{oz} = 3.9$ m/s, the stance initiating the jump onto the chair has an allowable touchdown leg angle error of 0.7 degrees. The desk is a standard 28 inches high (0.71 m), 0.7 m deep, and 1.2 m wide. Both the jump from the ground to the chair and from the chair to the desk are higher than the robot's full body length. The robot jumped 0.5 m longitudinally to mount the chair and then 0.3 m laterally to mount the desk.

This is a challenging task since the chair seat is small in comparison to the foot placement precision from attitude control accuracy; the robot did not always complete it successfully. This task also requires large changes in energy to mount and descend obstacles higher than the robot's size. The robot achieved desired vertical takeoff velocities by adding and removing energy when jumping higher and lower.

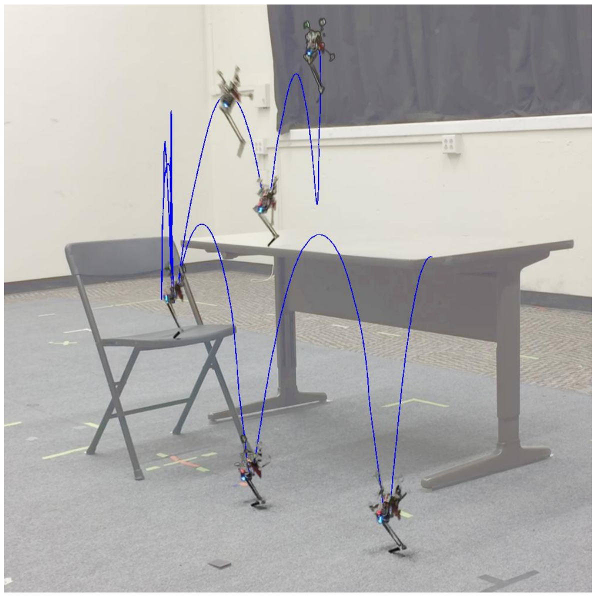


Fig. 9: The robot jumps up onto a chair and desk (trajectory in blue), then back down (not shown here, but shown in Fig. 10).

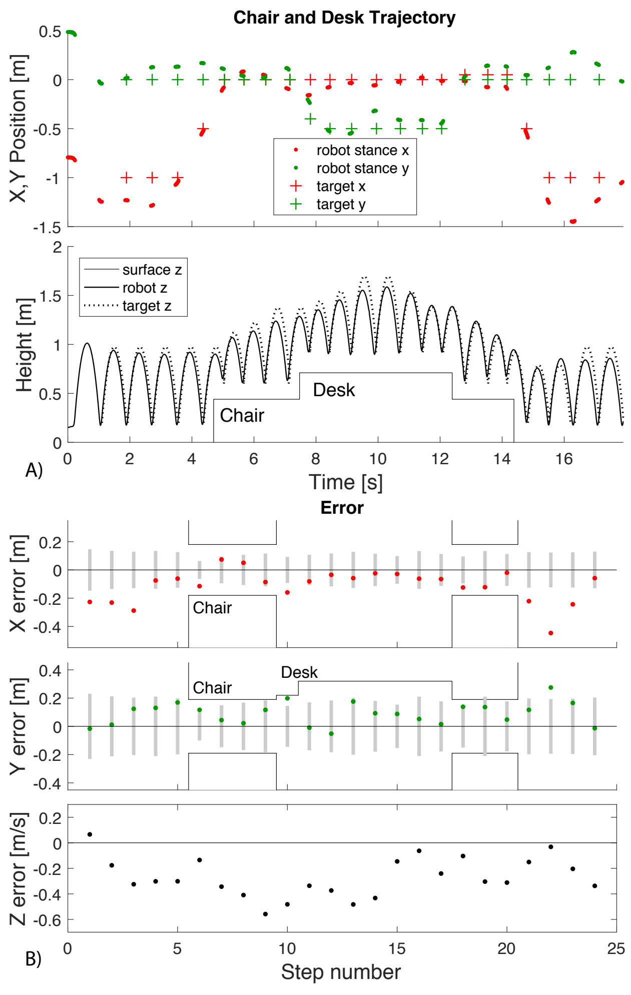
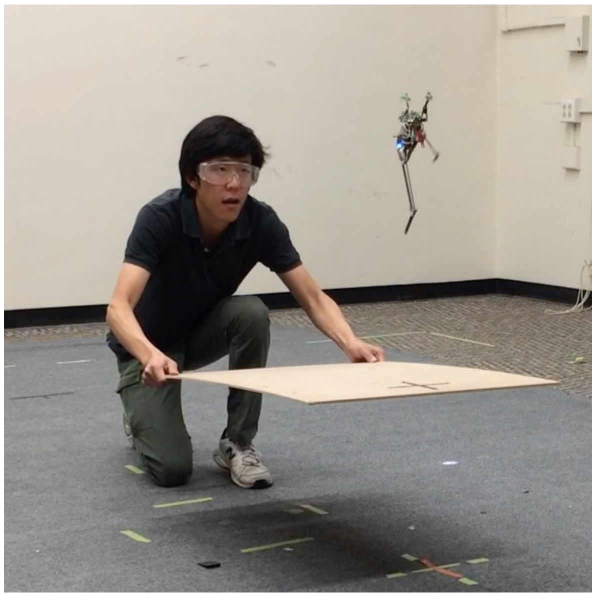


Fig. 10: A) Foot placements and height of the robot as it jumps onto the chair and desk and then back down.
B) Foot placement and v_{oz} error. Grey bars show foot placement error for the standard deviation of the random walk angle error. To avoid falling, foot placement error cannot cross furniture edges.



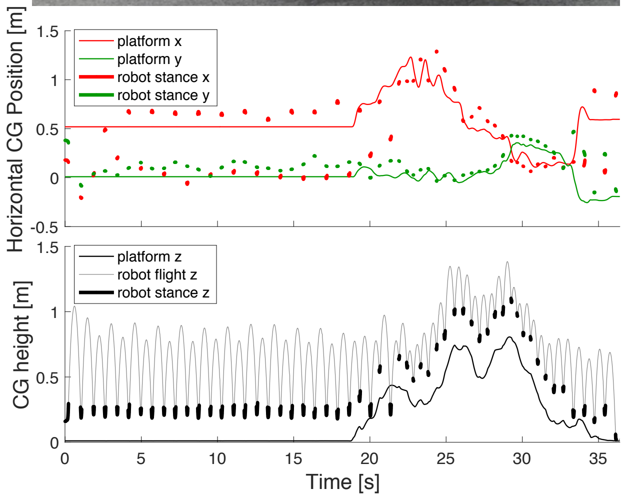


Fig. 11: Hopping on a moving platform.

D. *Jumping on a moving platform*

Finally, the deadbeat foot placement hopping controller is quick to compute and is robust to moderate ground disturbances. Due to its quick computation, it is able to quickly retarget to a new desired touchdown location. To demonstrate this, we outfitted a hand-carried wooden board with motion capture markers and commanded the robot to jump to a point on the board. The robot was commanded to alternate between the ground at (0,0,0) and the point on the board. The trajectories of the point on the board and the robot foot placements are plotted in Fig. 11.

The robot converged quickly to alternating between (0,0,0) and the board. At nineteen seconds, the platform was lifted off the ground. The robot completed two jumps from the ground to the board before being captured on top of the board at 22 seconds. The robot stabilized on top of the moving board until the board was placed on the ground and moved rapidly to the side at 33 seconds. The robot jumped between (0,0,0) and board twice before the aggressive horizontal jump length caused the robot to slip and fall over.

IV. Conclusion

We demonstrate that our deadbeat foot placement hopping controller achieves superior foot placement precision and jumping height adjustment when compared to an optimally tuned Raibert hopper controller for Salto-1P. This controller is capable of rapid changes in direction and velocity including reversing direction and turning at right angles. It is capable of controlling the energy in the system to achieve desired takeoff vertical velocities in a single jump.

We also analyze the sensitivity of the robot's takeoff velocity to touchdown leg angle deflections and show that flight-phase attitude error tolerance becomes tighter for higher hopping in which the vertical velocities are faster. This introduces an interesting tradeoff in jumping strategy. While larger jumps enable locomotion over footholds placed farther apart, foot placement precision degrades quadratically with increasing vertical velocity and linearly with increasing apex height. A larger number of smaller hops enables higher foot placement precision to avoid missing footholds. There is also a lower limit to the vertical velocity and height of small jumps to keep the flight-phase angular velocity within actuator limits. There is an optimal mid-to-low jump height at which the robot achieves its most precise foot placements.

The deadbeat foot placement hopping controller's high precision and good robustness to ground disturbances enabled it to hop on discontinuous surfaces like office furniture and to track a moving platform. The precision of the deadbeat foot placement hopping controller approaches the precision limit set by the accuracy of the flight-phase attitude control. Future improvements to the accuracy of the flight-phase attitude controller will enable more precise foot placement. Extensions to include the battery state of charge in the controller can improve the vertical velocity and hopping height control.

The third order polynomial deadbeat velocity controller is easy to compute. In future work it could easily be implemented onboard computation-limited embedded processors like the one on Salto-1P. Future work will also investigate a more theoretically founded understanding of hopping control sensitivity to errors based in simpler mathematical models like SLIP, stance-phase strategies that could correct touchdown errors, and the effects of non-rigid or sloped terrain on jumping performance and control strategy.

References

[1] O. Arslan and U. Saranli, "Reactive Planning and Control of Planar Spring-Mass Running on Rough Terrain," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 567–579, 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6112244

[2] R. Blickhan, "the Spring-Mass Model for Running and Hopping," *Journal of Biomechanics*, vol. 22, no. 1112, pp. 1217–1227, 1989.

[3] M. Campana and J.-p. Laumond, "Ballistic motion planning," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1410–1416, 2016.

[4] S. G. Carver, N. J. Cowan, and J. M. Guckenheimer, "Lateral stability of the spring-mass hopper suggests a two-step control strategy for running," *Chaos*, vol. 19, no. 2, 2009.

[5] G. Council, S. Yang, and S. Revzen, "Deadbeat control with (almost) no sensing in a hybrid model of legged locomotion," *International Conference on Advanced Mechatronic Systems, ICAMechS*, pp. 475–480, 2014.

[6] J. Englsberger, P. Kozlowski, C. Ott, and A. Albu-Schaffer, "Biologically Inspired Deadbeat Control for Running: From Human Analysis to Humanoid Control and Back," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 854–867, aug 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7527657/

[7] H. Geyer, A. Seyfarth, and R. Blickhan, "Spring-mass running: Simple approximate solution and application to gait stability," *Journal of Theoretical Biology*, vol. 232, no. 3, pp. 315–328, 2005.

[8] D. W. Haldane, M. M. Plecnik, J. K. Yim, and R. S. Fearing, "Robotic vertical jumping agility via series-elastic power modulation," *Science Robotics*, vol. 2048, no. 1, p. eaag2048, 2016. [Online]. Available: http://robotics.sciencemag.org/content/robotics/1/1/eaag2048.full.pdf

[9] D. W. Haldane, M. Plecnik, J. K. Yim, and R. S. Fearing, "A power modulating leg mechanism for monopedal hopping," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2016-Nov, 2016, pp. 4757–4764.

[10] D. W. Haldane, J. K. Yim, and R. S. Fearing, "Repetitive extreme-acceleration (14-g) spatial jumping with Salto-1P," *IEEE Int. Conf. Intell. Robots. Syst.*, pp. 3345–3351, 2017.

[11] J. K. Hodgins and M. H. Raibert, "Adjusting Step Length for Rough Terrain Locomotion," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 289–298, 1991.

[12] G. Piovan and K. Byl, "Reachability-based control for the active SLIP model," *Proceedings - IEEE International Conference on Robotics and Automation*, no. 1986, pp. 5656–5662, 2013.

[13] M. M. Plecnik, D. W. Haldane, J. K. Yim, and R. S. Fearing, "Design exploration and kinematic tuning of a power modulating jumping monopod," *Journal of Mechanisms and Robotics*, vol. 9, no. February, pp. 1–13, 2016.

[14] M. H. Raibert and J. H. B. Brown, "Experiments in Balance With a 3D One-Legged Hopping Machine," *Journal of Dynamic Systems, Measurement, and Control*, vol. 106, no. 1, pp. 75–81, 1984. [Online]. Available: http://dx.doi.org/10.1115/1.3149668

[15] M. H. Raibert and F. C. Wimberly, "Tabular Control of Balance in a Dynamic Legged System," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 2, pp. 334–339, 1984.

[16] U. Saranli, Ö. Arslan, M. M. Ankarali, and Ö. Morgül, "Approximate analytic solutions to non-symmetric stance trajectories of the passive Spring-Loaded Inverted Pendulum with damping," *Nonlinear Dynamics*, vol. 62, no. 4, pp. 729–742, 2010.

[17] U. Saranli, W. J. Schwind, and D. E. Koditschek, "Toward the control of a multi-jointed, monoped runner," *Proceedings of the 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146*, vol. 3, no. May, pp. 2676–2682, 1998. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=680750

[18] W. J. Schwind and D. E. Koditschek, "Approximating the stance map of a 2-DOF monoped runner," *Journal of Nonlinear Science*, vol. 10, no. 5, pp. 533–568, 2000.

[19] 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. [Online]. Available: http://jeb.biologists.org/cgi/doi/10.1242/jeb.00463

[20] N. Shemer and A. Degani, "A flight-phase terrain following control strategy for stable and robust hopping of a one-legged robot under large terrain variations," *Bioinspiration and Biomimetics*, vol. 12, no. 4, p. aa741f, 2017. [Online]. Available: https://doi.org/10.1088/1748-3190/aa741f

[21] P. Terry, G. Piovan, and K. Byl, "Towards precise control of hoppers: Using high order partial feedback linearization to control the hopping robot FRANK," *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, no. Cdc, pp. 6669–6675, 2016.

[22] A. Wu and H. Geyer, "The 3-D spring-mass model reveals a time-based deadbeat control for highly robust running and steering in uncertain environments," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1114–1124, 2013.