The 3-D Spring–Mass Model Reveals a Time-Based Deadbeat Control for Highly Robust Running and Steering in Uncertain Environments

Albert Wu and Hartmut Geyer, *Member, IEEE*

***Abstract*—Over the past three decades, the spring–mass model has developed into the basic behavior model to study running in animals and robots. In the planar version, this model has helped to reveal and understand the passive stabilization of running in the horizontal and sagittal planes, and to derive from this knowledge control strategies for running robots. However, only few attempts have been made to transfer the knowledge to 3-D locomotion. Here, we show that the 3-D spring–mass model reveals a deadbeat control that does not require feedback about the actual ground level to produce highly robust running and steering in uncertain environments. The control naturally extends the time-based control derived for the planar version of this model and allows it to navigate rough terrain, while stabilizing running and steering. Using this control strategy, we demonstrate in simulation that a human-like system running at 5 ms^{-1} tolerates frequent ground disturbances up to 30% of the leg length. Moreover, we find that the control outperforms a classical leg-placement strategy in terms of turning rate and disturbance rejection if the relative errors in system energy and the other model parameters stay small (<10%). Our results suggest that the time-based control can be a powerful alternative for leg-placement strategies in highly maneuverable running robots.**

***Index Terms*—Control theory, legged locomotion, nonlinear dynamical systems, robustness.**

I. Introduction

Over the past three decades, the spring–mass model has developed into the basic behavior model for studying the running gait in animals and robots. Established in the 1980s [1]–[4], the model helped early works develop locally stable feedback controls for hopping and running robots [2], [5] and to understand the leg function in animal and human runners [3], [4], [6]–[9]. More recently, the focus on this model has been shifted to developing approximate solutions [10]–[12] and to understanding global running stability by analyzing the model's hybrid dynamics with Poincaré maps. This analysis method has proved especially productive [13]. It has helped to reveal and understand the passive stabilization of insect running in the horizontal plane [14]–[16], and of robot and human running in the sagittal plane [17]–[20]. In addition, it has resulted in control strategies that promise to largely improve the stability in running robots [21]–[27], to the extent that they could one day outperform their animal competitors.

While the planar spring–mass model has greatly advanced the understanding of running dynamics and control in the horizontal and sagittal planes, few attempts have been made to transfer this knowledge to 3-D locomotion. Seipel and Holmes [28] investigated running stability with Poincaré analysis of the 3-D spring–mass model. They showed that symmetric gaits with a constant in-flight leg orientation defined with respect to the desired direction of motion are unstable. By introducing linear feedback control derived from a local stability criterion, they found stable running for motions sufficiently close to the sagittal plane. Peuker *et al.* [29] expanded these results in a recent numerical study. Inspired by insights from the work on insect running in the horizontal plane [16], they explored running stability using constant leg angle policies with respect to the direction of motion and observed locally stable gaits for a large range of leg angles. Beyond local stability analysis, Carver *et al.* [30] investigated deadbeat stepping policies that drive the 3-D spring–mass model into desired system states using the least number of steps. In particular, after deriving existence and uniqueness conditions, they proposed an iterative algorithm to compute these policies and demonstrated it in examples where the model's horizontal position is perturbed in flight.

Here, we show that identifying deadbeat policies for stability and steering in 3-D spring–mass running can be largely simplified and that the resulting controller can in part be embedded in a feedforward manner, generating highly robust locomotion over rough terrain. First, by parameterizing the full-model dynamics in a motion-aligned frame, we reduce the essential behaviors of stability and steering to a compact representation. We then use simple interpolation over this representation to precompute the deadbeat policies to attain all possible running behaviors. Finally, we map these policies into the time after apex and show in a simulation example that the resulting control enables navigation to waypoints over rough terrain with large, frequent, and unknown ground disturbances (up to 30% of the leg length in the example). Our results suggest that compact and efficiently computed deadbeat policies for the 3-D spring–mass model can be combined with time embedding to provide a powerful alternative for leg-placement strategies in highly maneuverable running robots.

Manuscript received September 25, 2012; revised February 18, 2013; accepted May 9, 2013. Date of publication May 30, 2013; date of current version September 30, 2013. This paper was recommended for publication by Associate Editor E. Yoshida and Editor J.-P. Laumond upon evaluation of the reviewers' comments. This work was supported by the DARPA Maximum Mobility and Manipulation Program under Contract W91CRB-11-1-0002 and by the National Science Foundation Dynamical Systems Program under Award CMMI-1100232.

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: albertwu87@gmail.com; hgeyer@cs.cmu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2013.2263718

II. Three-Dimensional Model and Problem Formulation

A. Dynamics

We model 3-D spring–mass running similar to previous work [18], [20], [28], [30], [31] (see Fig. 1). The flight phase is modeled as a point mass m following a ballistic trajectory with

$$m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix}=\begin{bmatrix}0\\ -mg\\ 0\end{bmatrix}\tag{1}$$

where the coordinates are given in the apex frame. The apex describes the highest point in flight and the apex frame aligns the x-axis with the direction of motion $\boldsymbol{v}$ at that point. In addition, the frame's origin is located at the ground level. After the apex event, the angle of attack α and the splay angle β parameterize the swing-leg orientation. The swing leg has a constant length l_0, and the model transitions to stance when the point mass reaches the touchdown height $y_{\mathrm{TD}} = l_0 \sin(\alpha)$.

During stance, the point mass rebounds on a massless spring with stiffness k and rest length l_0, assuming that the foot point stays fixed on the ground. The dynamics of the point mass are governed by the gravitational force $m\boldsymbol{g}$ and the spring force $\boldsymbol{F} = k(l_0/|\boldsymbol{l}| - 1)\boldsymbol{l}$ acting along the leg axis $\boldsymbol{l}$

$$m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix}=k(l_0/|\boldsymbol{l}|-1)\begin{bmatrix}x-x_f\\ y\\ z-z_f\end{bmatrix}+\begin{bmatrix}0\\ -mg\\ 0\end{bmatrix}\tag{2}$$

where $[x\ 0\ z]_f^T$ is the location of the foot point and $|\boldsymbol{l}|^2 = (x - x_f)^2 + y^2 + (z - z_f)^2$. The resulting trajectory of the point mass determines the spatial orientation and length of the spring. When the spring length returns to l_0 during rebound, the model transitions back to the flight phase.

The model describes running dynamics with alternating stance and flight phases of the left and right legs as long as four conditions are fulfilled. First, the model reaches an apex in flight. Second, at apex, the height of the point mass exceeds the landing height, $y_a > l_0 \sin(\alpha)$. Third, the point mass remains above the ground throughout stance, $y > 0$. And fourth, the horizontal leg forces stay within the friction limit, $\sqrt{F_x^2 + F_z^2} < \mu F_y$, where μ is the static friction coefficient. (We do not consider slipping.) The last condition translates into a minimum angle of attack $\alpha_{\min} = \mathrm{atan}(\mu^{-1})$. Smaller landing angles will always lead to a violation of the friction condition in stance.

B. Behavior Function and Control Objective

The behavior of the spring–mass model is captured by the apex return map, which is a discrete mapping of the system state at apex between two consecutive flight phases. The complete map tracks the state vector $[x\ \dot{x}\ y\ \dot{y}\ z\ \dot{z}]_a$, where the subscript a denotes the apex for each variable. However, to characterize the model behavior, it suffices to track only a pair of variables, the apex height y and the deflection $d\theta$ in heading. Similar to the planar case, $\dot{y}_a = 0$ by the definition of the apex. Furthermore, the stance dynamics depend on the relative position of the foot with respect to the point mass and its direction. This position is parameterized by α and β, making $[x\ z]_a$ unnecessary to compute the motion between apexes (just as x can be discarded given α in the planar case). Note, however, that these coordinates are required to track the absolute position of the system within the global frame. Among the remaining variables, the magnitude of the horizontal velocity is coupled to the height y_a by the constant system energy E_s with $\dot{x}_a^2 + \dot{z}_a^2 = \frac{2E_s}{m} - 2gy_a$. Therefore, the model behavior is captured by

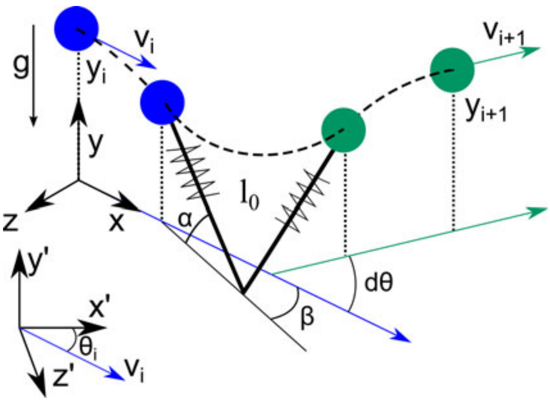

Fig. 1. Three-dimensional spring–mass running model. The model describes a sequence of flight and stance phases. The ith flight is characterized by a ballistic trajectory of a point mass m. The apex velocity vector $\boldsymbol{v}_i$ and the vertical axis define an apex frame (x, y, z) and a heading θ_i with respect to the world frame $(x, y, z)'$. The leg of length l_0 is positioned with an angle of attack α and a splay angle β, both measured in the apex frame. In stance, the point mass rebounds on a massless spring leg (stiffness k, rest length l_0). The model's net behavior can be captured by a behavior function f which maps the apex height $y_{i+1}(y_i, \alpha_i, \beta_i)$ and the change in heading $d\theta(y_i, \alpha_i, \beta_i)$. Because of the symmetry of the model, f equally captures left- and right-leg behavior of the alternating stance phases in running. g: gravitational acceleration, $\boldsymbol{v}_{i+1}$: velocity vector at apex $i + 1$.

$$(y_{i+1}, d\theta) = f(y_i, \boldsymbol{p})$$

where $d\theta$ is the horizontal deflection after the stance phase, f is a function of the model dynamics, and $\boldsymbol{p}$ is the vector of model parameters. That is, given $\boldsymbol{p}$, the previous apex height y_i is the only state needed to derive the *relative* motion of the next flight phase. From this representation, the apex return map R that characterizes running stability and global steering of the 3-D spring–mass model can be formalized as

$$(y, \theta)_{i+1} = R(y_i, \theta_i, \boldsymbol{p})$$

with θ_i describing the global heading in the ith step (see Fig. 1) and $\theta_{i+1} = \theta_i + d\theta$.

The model behavior can be controlled by manipulating the return map, for which a goal behavior is equivalent to a target state $(y, \theta)^*$. The control problem consists of finding ways to drive the model to the target state from arbitrary initial conditions, and deadbeat control is achieved if a parameter policy $\boldsymbol{p}(y_i, \theta_i, y^*, \theta^*)$ exists that enforces $(y, \theta)_{i+1} = (y, \theta)^*$ for all possible $(y, \theta)_i$. As a result, the general problem of controlling the model behavior resolves for each initial state i to the minimization problem

$$\boldsymbol{p}_i = \operatorname*{argmin}_{\boldsymbol{p}} d((y, \theta)^*, R(y_i, \theta_i, \boldsymbol{p}))\tag{3}$$

where $d(.,.)$ is an appropriate measure of distance (for example, the Euclidean norm). If a vector $\boldsymbol{p}_i$ exists that brings the minimum to zero, then deadbeat control is achieved; otherwise, the state $(y,\theta)_{i+1}$ will be as close to $(y,\theta)^*$ as possible.

In the 3-D spring–mass model, the vector

$$\boldsymbol{p} = [E_s\ m\ g\ k\ l_0\ \alpha\ \beta]$$

contains seven parameters, several of which are suitable for control. In particular, we choose the two angles that define the swing-leg orientation as the control inputs, $\boldsymbol{u} = [\alpha\ \beta]$, to demonstrate how leg placement in the flight phase generalizes the deadbeat control of planar spring–mass running to 3-D locomotion and steering. The remaining parameter vector $\hat{\boldsymbol{p}}$ is fixed to $E_s = 1785$ J, $m = 80$ kg, $g = 9.81$ ms^{-2}, $k = 15$ kNm^{-1} and $l_0 = 1$ m; these values fit a human-like system running at about 5 ms^{-1} [18]. The model behavior function f is, thus, written as

$$(y_{i+1}, d\theta) = f(y_i, \boldsymbol{u}, \hat{\boldsymbol{p}}) \tag{4}$$

and the control problem reduces to

$$\begin{aligned}[\alpha\ \beta]_i &= \underset{[\alpha\ \beta]}{\operatorname{argmin}}\, d((y,\theta)^*, R(y_i,\theta_i,\boldsymbol{u},\hat{\boldsymbol{p}})) \\ &= \underset{[\alpha\ \beta]}{\operatorname{argmin}}\, d((y^*, \theta^* - \theta_i), f(y_i,\boldsymbol{u},\hat{\boldsymbol{p}})).\end{aligned} \tag{5}$$

Note that more parameters can be included in solving (3). The added degrees of freedom in the control input can then be used to consider further performance metrics (compare [27] for a planar example), or to embed lower priority tasks as in redundant manipulators [32]. In this study, matching the degrees of freedom between the control targets $(y,\theta)^*$ and inputs (α,β) avoids this redundancy and simplifies the demonstration of deadbeat control.

III. Control Strategy Derivation

The behavior function $f(y_i, \boldsymbol{u}, \hat{\boldsymbol{p}})$ is nonlinear. To solve (5) for arbitrary target states, we first perform a numerical scan to generate a discrete representation of f. We then define a particular measure d and use interpolation to derive the control inputs $[\alpha\ \beta]_i$ that project from the initial states into arbitrary target states. In the final part of this section, we apply the transformation between apex height and falling time [21] to derive a control strategy that does not require knowledge about the actual ground level to produce deadbeat control of stability and steering.

A. Discrete Representation of Model Behavior

We scan the model behavior $f(y_i, \boldsymbol{u}, \hat{\boldsymbol{p}})$ over the inputs α and β for all possible initial apex heights y_i. As control input ranges, we choose $\alpha \in [45°, 90°]$ and $\beta \in [0°, 180°]$. The lower bound $\alpha_{\min}$ is governed by the static friction limit assuming $\mu = 1$. Because of the reflection symmetry of the model with respect to the sagittal plane, it suffices to scan β within 0° and 180° to capture the model behavior for all possible splay angles. The initial height is limited to $y_i \in [0.71, 2.27]$ m. The minimum corresponds to the landing height defined by $y_{\min} = l_0 \sin\alpha_{\min}$; the maximum $y_{\max} = E_s/(mg)$ results from converting all system energy into potential energy. Using these ranges, we generate the discrete functions $y_{i+1}[\alpha,\beta,y_i]$ and $d\theta[\alpha,\beta,y_i]$ that represent $f(y_i,\boldsymbol{u},\hat{\boldsymbol{p}})$ (4) with an α–β–y_i resolution of 22 × 45 × 150 by numerically integrating the model dynamics (1) and (2) between two apexes (computation time less than a day on a modern desktop PC).

Fig. 2 shows the discretized model behavior for two different initial heights y_i. The control inputs α and β are represented as foot point locations relative to the point mass and projected onto the *xz* plane. The projection forms a disk centered on the origin, which corresponds to $\alpha = 90°$ with the foot directly below the point mass. The distance from the origin is given by $l_0\cos(\alpha)$, and β describes the angular position relative to the direction of motion. The friction limit defines the maximum disk radius $r_{\max} = l_0\cos\alpha_{\min}$ [see Fig. 2(a)]. The discrete functions $y_{i+1}[\alpha,\beta,y_i]$ and $d\theta[\alpha,\beta,y_i]$ are well behaved and centered about a characteristic point (white circle) lying on the $\beta = 0°$ axis. This characteristic point is the spring–mass model's equivalent to the capture point of the linear inverted pendulum walking model [33]–[35]. Landing with the foot positioned at this point stops the motion in the *xz* plane as all the system energy is redirected into vertical motion with $y_{i+1} = y_{\max}$. Landing past this point leads to backward motion ($d\theta = 180°$). Foot positions that result in equal apex heights y_{i+1} form concentric rings about the stopping point with decreasing heights, eventually interrupted by the $r_{\max}$ given by the friction limit. By contrast, configurations with equal deflections $d\theta$ form rays that originate from the stopping point. The closeup in Fig. 2(b) shows that the inputs $[\alpha\ \beta]_i$ can reach a large range of target behaviors $(y, d\theta)^*$.

The range of target behaviors that can be reached diminishes when the initial height drops below the rest length $l_0 = 1$ m of the spring. Fig. 2(c) and (d) shows an example with $y_i = 0.85$ m. The disk of foot positions reduces to an annulus with an inner radius $r_{\min} = \sqrt{l_0^2 - y_i^2}$ that corresponds to a landing angle $\alpha_{\max} = \operatorname{asin}(y_i/l_0)$ [see Fig. 2(c)]. Steeper angles of attack cannot exist due to ground contact. As a result, individual behavior goals may not always be achievable simultaneously. For instance, a target height of $y^* = 0.95$ m cannot be achieved with a target deflection $d\theta^* = 0°$ from this initial apex height [see Fig. 2(d)].

B. Deadbeat Control Policy

To control the model behavior, we prioritize robustness against ground disturbances over heading directions and require that the apex height goal is achieved whenever possible. To realize this constraint, we split (5) into two stages. The first stage identifies the set of control inputs

$$U_{y^*}(y_i) = \{[\alpha\beta] : y_{i+1}(\alpha,\beta,y_i) = y^*\} \tag{6}$$

that enforce the desired apex height y^*. The second stage then solves the minimization problem

$$[\alpha\ \beta]_i = \underset{U_{y^*}(y_i)}{\operatorname{argmin}}\, |\theta^* - \theta_i - d\theta(\alpha,\beta,y_i)| \tag{7}$$

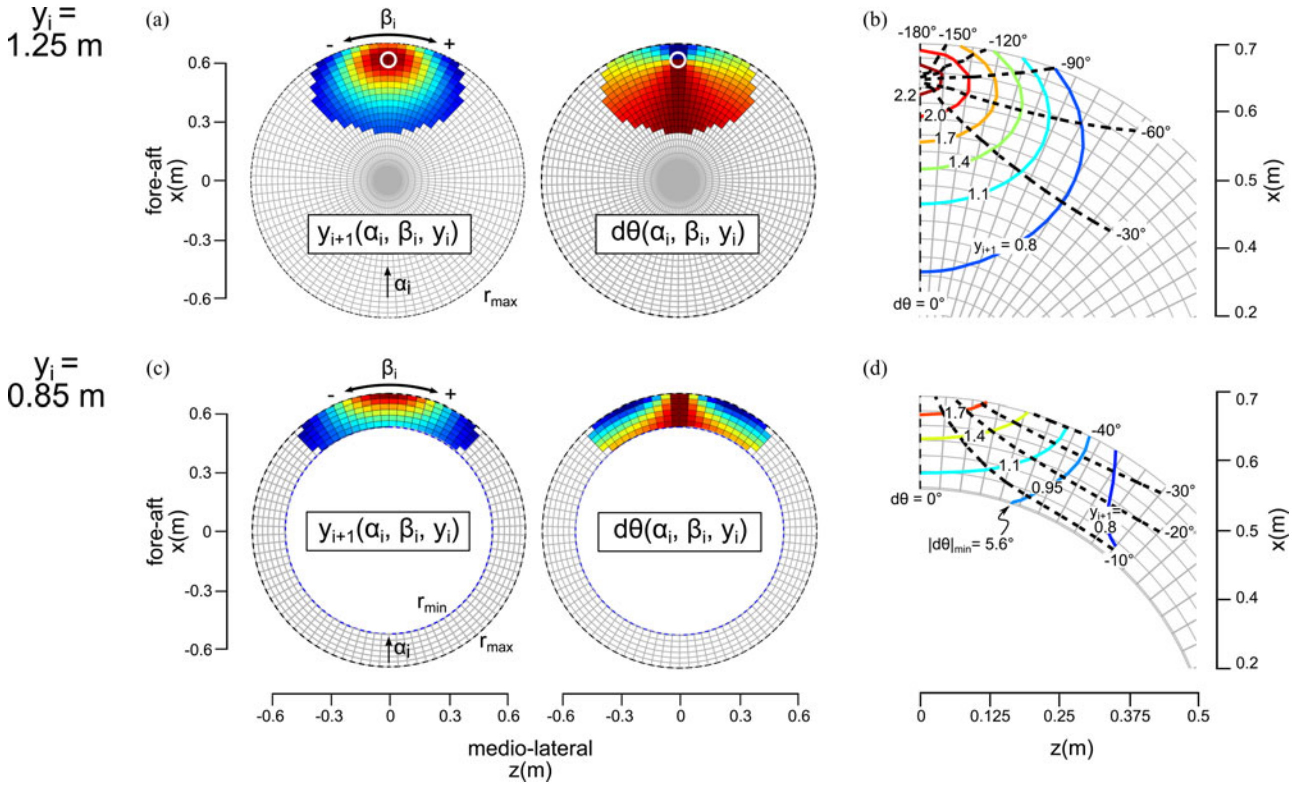

Fig. 2. Numerical representations of model behavior for two different initial apex heights $y_i = 1.25$ m (a and b) and $y_i = 0.85$ m (c and d). (a) Apex height $y_{i+1}[\alpha, \beta, y_i]$ and deflection $d\theta[\alpha, \beta, y_i]$ as functions of foot placement projected onto horizontal plane (see Fig. 1) for $y_i = 1.25$ m. The radius $r_{\max}$ describes the friction limit. Maps are color coded from low (blue) to high values (red). Empty regions indicate that the model does not return to an apex in the next flight phase. The white circle highlights the stopping point at which all energy is directed into vertical motion. (b) Closeup section with apex height and deflection behaviors overlaid as contour plots. Contour lines show inputs $[\alpha\ \beta]_i$ that produce the same apex height y_{i+1} (ring sections) or deflection $d\theta$ (rays). (c) and (d) For $y_i = 0.85$ m, the landing condition limits the maximum angle of attack to $\alpha_{\max} = \text{asin}(y_i/l_0)$. This constraint results in a minimum radius $r_{\min}$ for the feasible foot positions, reducing the domain of the behavior map projection to an annulus.

to find the control input $[\alpha\ \beta]_i \in U_{y^*}(y_i)$ that best aligns the heading with the target heading θ^*. Given a target apex height, the two stages are straightforward to implement using bilinear interpolation on the discrete functions $y_{i+1}[\alpha, \beta, y_i]$ and $d\theta[\alpha, \beta, y_i]$ for the range of initial apex heights y_i.

The constrained optimization on the behavior function yields the tables $\alpha[y_i, d\theta^*, y^*]$ and $\beta[y_i, d\theta^*, y^*]$, which contain the control policy that projects initial states $(y, \theta)_i$ into target states $(y, \theta)^*$. The domain of this policy is limited to the set of initial heights $\{y_i | U_{y^*}(y_i) \neq \emptyset\} \subset [l_0 \sin(\alpha_{\min}), E_s/mg]$ for which the apex constraint can be satisfied. If y_i falls outside of this domain, it is impossible to project to the target y^* in one step and the control policy picks the nearest neighbor height from the domain.

As an example, Fig. 3(a) shows, as contour plots, the resulting policy to achieve a target height of $y^* = 1.1$ m. For this target height, all possible initial heights $y_i \in [0.71, 2.27]$m can be projected to $y_{i+1} = y^*$ in a single step. The policy is characterized by a divider, which is defined by the friction limit and the touchdown condition (black curve). Above (and to the right of) the divider is the deadbeat region, where any target deflection can be achieved along with the target height, realizing deadbeat control for both behavior goals. Below the divider, however, target deflections cannot be reached and the control policy returns the closest alternative. For instance, the spring–mass model can achieve steady state running with $y_i = y^* = 1.1$ m for target deflections $d\theta^* = \theta^* - \theta_i$ up to ±74°, where the control input is $[\alpha\ \beta] = [45^\circ, \mp 20^\circ]$ (intersection of red line and black curve in each panel). For larger target deflections, the control input will remain at $[45^\circ, \mp 20^\circ]$, maintaining deadbeat control of the apex height but clipping the directional tracking.

C. *Transformation Into Time Policy*

Previous work on the planar spring–mass model has shown that deadbeat control using the angle of attack $\alpha(y_i)$ can be transformed into a time-based control $\alpha(t)$ during flight if the time is measured from the apex event, $t \geq t_{\text{apex}}$ [21], [22], [27]. This transformation fundamentally changes the control. Instead of selecting a single-leg placement based on a single height at apex, the time-indexed sequence of leg placements embeds the deadbeat behavior across a range of apex heights. In effect, ground height variations in the upcoming step are automatically accommodated as they correspond to different falling times. We generalize this transformation to include the splay angle β and generate a control policy that does not require knowledge about the actual ground level to produce a time-based deadbeat control of stability and steering.

The time transformation takes advantage of the free-fall dynamics after the apex to replace y_i with a function of time

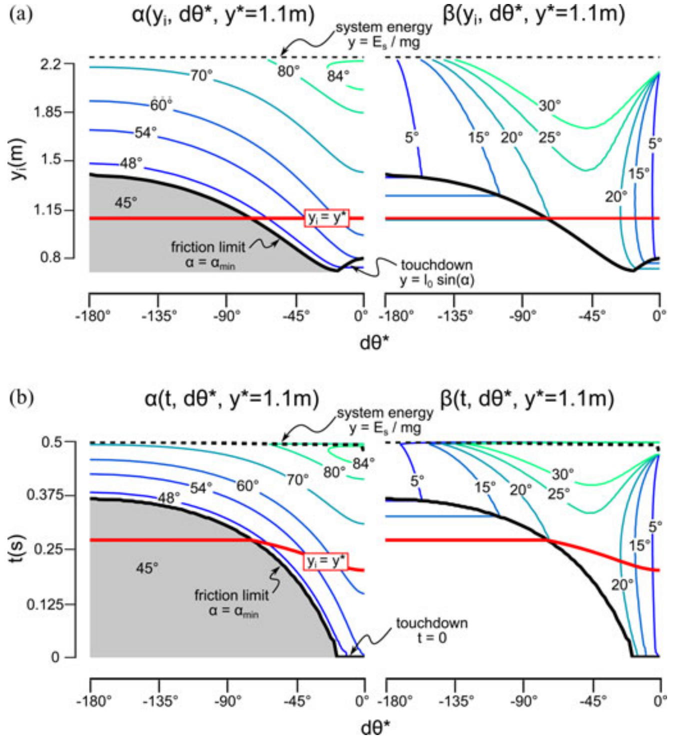

Fig. 3. Control policy to achieve $y^* = 1.1$ m. (a) Control tables $\alpha[y_i, d\theta^*, y^*]$ and $\beta[y_i, d\theta^*, y^*]$ are shown as contour plots for all possible initial heights y_i with ranges of target deflections $d\theta^*$. The black curve describes the friction and touchdown limits. The region of deadbeat control for stability and steering is above this divider and below the system energy limit. Below the deadbeat region, the control policy clips the parameters to the ones on the divider to minimize $|d\theta - d\theta^*|$ while enforcing $y_{i+1} = y^*$. The red line indicates the steady-state control $y_i = y_{i+1} = y^*$. (b) Same as (a) except that control tables are transformed to the time after apex.

in the control tables. The height of the point mass at touchdown is given by $y_{\mathrm{TD}} = l_0 \sin\alpha$. Given that time t has passed between apex and touchdown, this apex must have been at a height $y_{\mathrm{apex}} = l_0 \sin\alpha + \frac{g}{2}t^2$. The correspondence can be used to transform the table $\alpha[y_i, d\theta^*, y^*]$ into a time-based table

$$\alpha[y_i, d\theta^*, y^*] \rightarrow \alpha[t_i, d\theta^*, y^*] \tag{8}$$

which defines the angle-of-attack control as a time policy after the apex event. At the time

$$t_i = \sqrt{\frac{2}{g}(y_i - l_0 \sin\alpha[y_i, d\theta^*, y^*])}$$

the angle of attack assumes the value $\alpha[y_i, d\theta^*, y^*]$. If the model lands, the apex height must have been y_i; otherwise, the flight phase continues and the system automatically prepares for stance at a later time corresponding to a different pair of apex height and angle of attack. Since $\alpha[y_i, d\theta^*, y^*]$ belongs to a joint control input for α and β, the control table of the splay angle transforms in the same way

$$\beta[y_i, d\theta^*, y^*] \rightarrow \beta[t_i, d\theta^*, y^*]. \tag{9}$$

In effect, (8) and (9) realize deadbeat control of stability and steering without actually sensing the ground level.

Fig. 3(b) shows the resulting time policy for the previous example of achieving a target height of $y^* = 1.1$ m [see Fig. 3(a)]. The policy shares features with the original control policy. A divider (black curve) again separates between the deadbeat control region that achieves target deflections along with the target heights (above divider) and the region in which the deflections are clipped to the closest alternative (below divider). In the time policy, the divider is solely defined by the friction limit as the landing condition maps onto $t = 0$. At that time, only deflections that are smaller than $\pm 20^\circ$ can be realized. As time progresses, larger target deflections become possible with the same minimum landing angle ($\alpha_{\min} = 45^\circ$), allowing sharper turns that in steady state (red curve with $y_i = y^*$) can reach up to $\pm 74^\circ$ per step. Reducing the friction coefficient μ enforces larger maximum landing angles and leads to smaller maximum deflections.

For a given target deflection $d\theta^*$, the time policy describes an increasing angle of attack $\alpha(t)$ that retracts the leg in the present example. Although leg retraction has been associated with stabilizing running [36], [37], in general, it is not the cause of stability. Rather, deadbeat control requires a specific sequence of landing angles, which can appear as retraction for some model parameters but not for others.

IV. Control Performance in Simulation

We test the performance of the time-based control strategy (8) and (9) for running and steering in a 3-D simulation environment. Specifically, we focus on locomotion robustness when encountering large, frequent, and unknown changes in the ground height, and on the sensitivity of the control to changes in model parameters. In addition, we provide in these tests a comparison to the leg-placement strategy of the classical Raibert controller [2] as a baseline to assess the quality of the proposed strategy.

A. *Robustness to Changes in Ground Height or System Energy*

In the robustness test, the simulated environment consists of flat square tiles (1 m ×1 m), which define random height levels that are uniformly distributed between $\pm\frac{\Delta y_{\max}}{2}$ (see Fig. 4), with $\Delta y_{\max} = 25$ cm (25% of the model's nominal leg length) as the maximum step change that can be encountered going up or down. The spring–mass model is launched into this environment at an initial apex height of $y_0 = 1$ m with a velocity of $5\,\mathrm{ms}^{-1}$ (model parameters as defined in Section II). The target height is set to $y^* = 1.1$ m. In addition, we assume that waypoints define heading targets, where the target deflection $d\theta^*$ is computed as the change required to redirect the motion in flight toward the current waypoint. Once the model gets within 0.5 m of this waypoint, the target heading switches to the next one. (We forego advanced path planning algorithms for simplicity.)

In the simulation, the time-based control shows large robustness to changes in the ground level. The model navigates robustly through the 3-D environment, maintaining running stability, while making sharp turns toward the waypoints (see Fig. 4). The control performance is summarized in Fig. 5 for a random navigation trail with a total of 500 steps [see Fig. 5(a)]. The vertical steps encountered on the trail cover the full range from −25 to +25 cm [see Fig. 5(b)]. Whenever touchdown occurs

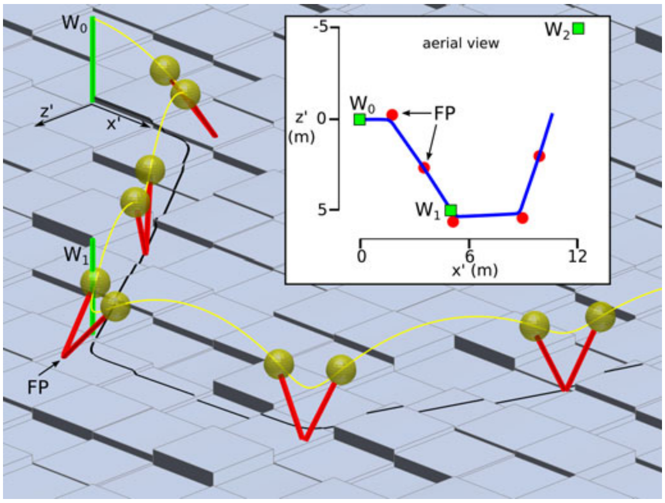

Fig. 4. Time-based control of stability and steering in 3-D environments with large, frequent, and unknown changes in the ground height. The spring–mass model is launched at 5 ms^{-1} over random ground with step changes up to 25 cm. In flight, the leg's attack and splay angles are updated after the apex event based on the time policy (8) and (9) with a target height of $y^* = 1.1$ m [see Fig. 3(b)]. Target deflections are generated by waypoints W$_i$ (green poles). The resulting trajectory of the point mass is shown (yellow, projection onto horizontal plane in black) with the leg sketched at touchdown and takeoff (red). The inset provides an aerial view with the point mass trajectory in blue and the resulting foot placements FP in stance marked as red circles.

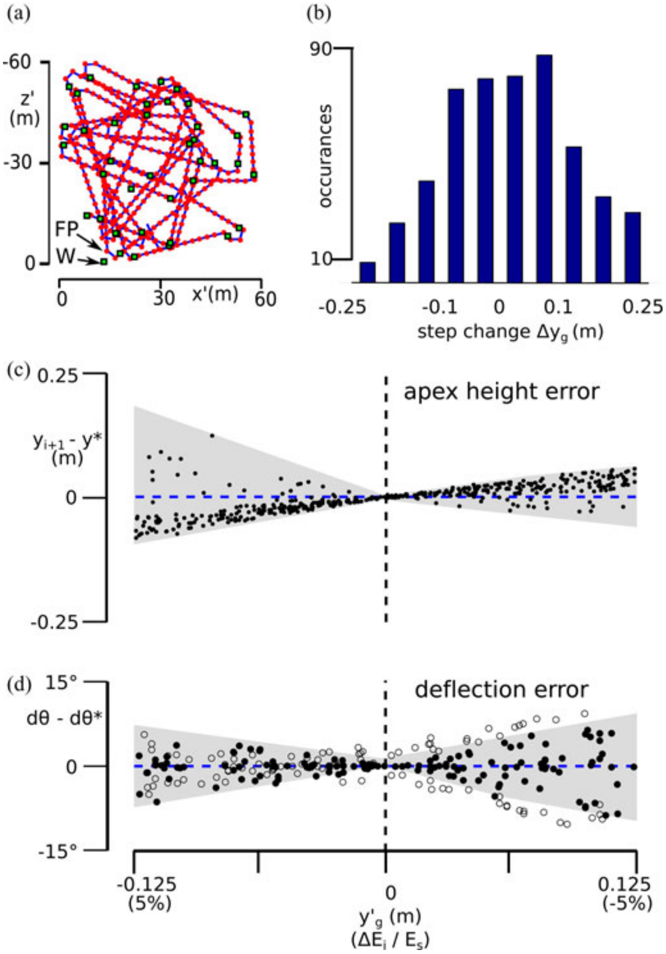

Fig. 5. Performance of time-based control with target height $y^* = 1.1$ m. (a) Aerial view of random navigation trail with 500 steps. Waypoints (W), resulting point mass trajectory and foot points (FP) shown as green squares, blue curve, and red circles, respectively. (b) Histogram of encountered step changes. (c) Error in apex height $\epsilon_y = y_{i+1} - y^*$ as functions of ground level y'_g and relative disturbance in system energy. The gray envelope is used in later figures for comparisons. (d) Corresponding deflection errors $\epsilon_{d\theta} = d\theta - d\theta^*$. If target deflections fall outside of the friction limit divider line [see Fig. 3(b)], $\epsilon_{d\theta}$ is measured with respect to the clipped target deflections. In contrast with the nonclipped cases (filled circles), the clipped ones (open circles) can have deflection errors at $y'_g = 0$ due to interpolating the divider line.

at a ground height $y'_{g,i} = 0$, the target height is met exactly [$y_{i+1} - y^* = 0$, see Fig. 5(c)]. Ground deviations $y'_{g,i} \neq 0$, on the other hand, can lead to subsequent apex heights y_{i+1} with errors $\epsilon_y = y_{i+1} - y^*$ up to -10/$+20$ cm and to errors in deflection $\epsilon_{d\theta} = d\theta_{i+1} - d\theta^*$ of up to ±10°. These errors fall within the deadbeat region of the control (see Section III-B and Fig. 3) such that they are corrected in the next step and do not accumulate. In effect, the system keeps robustly navigating through the environment despite the large ground disturbances.

The tracking errors are caused by changes in system energy, which are induced by the changing ground levels. We derived the time-based deadbeat control from the behavior function f assuming a particular system energy $E_{\text{sys}} = \frac{m\|\boldsymbol{v}_0\|^2}{2} + mgy_0$ given by the initial conditions $\|\boldsymbol{v}_0\| = 5$ ms^{-1} and $y_0 = 1$ m. In the environment with nonzero landing heights, however, the effective system energy changes from step to step by $\Delta E_i = -mgy'_{g,i}$ (between ±5% of the total energy), which alters the system behavior and reduces the precision of the applied deadbeat control. (Note that changes in ground level or system energy, thus, express the same disturbance to the system behavior as indicated by the horizontal axis in Fig. 5(c).

In general, we observe similar ranges of errors when encountering disturbances of similar *relative* changes in energy. For instance, at system energies corresponding to slower running speeds of 2.5 and 3.75 ms^{-1} (at apex height $y_a = 1$ m), we find similar error ranges to the ones we observed for running at 5 ms^{-1} if we limit the ground changes to levels of 6.5 and 8.5 cm, which correspond to the same 5% changes in energy.

Although energetic disturbances can be corrected by stabilizing the system energy in stance, such corrections reduce the energetic efficiency of locomotion when performed from step to step. Because the spring–mass model is conservative, it cannot correct for changes in system energy. To overcome this limitation, several extensions of this model have been proposed in theory and application [2], [5], [23], [38]–[40]. These extended systems can effectively cope with energetic disturbances ranging from dissipative losses to sloped terrain by regulating system energy during stance. However, the control proposed here demonstrates that neither the tight regulation of system energy from step to step is required on rough terrain that is flat on average, nor would such a regulation be energetically effective, since it would brake in one step only to reaccelerate in the next. In contrast, the proposed control strategy tolerates local, step-to-step changes in system energy or ground level without consuming power. An extension of this control with an explicit method for stabilizing system energy over a longer horizon of

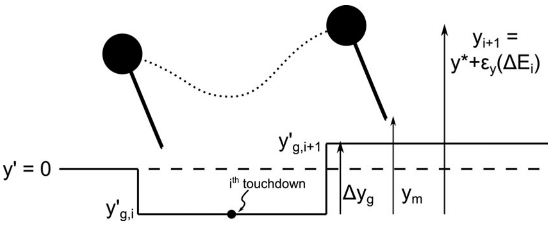

Fig. 6. Stumbling margin. Variable ground levels $y'_{g,i}$ introduce energetic disturbances $\Delta E_i = -mgy'_{g,i}$ resulting in the tracking error $\epsilon_y(\Delta E)$ that is measured with respect to the ground at the previous touchdown. The stumbling margin y_m describes the maximum height of the ground change Δy_g that the system can tolerate without stumbling in the next step.

multiple steps could thus combine robust and energy efficient running and steering in sloped environments.

B. Margin of Stability Against Stumbling

While the model did not fail, the margin of stability against stumbling was only $\Delta y_{\max} = \pm 19$ cm in this example. The stability margin y_m is given by the vertical distance

$$y_m = y_{i+1} - l_0 \sin\alpha[0, d\theta^*, y^*]$$

between the foot and the ground at apex (see Fig. 6). Stumbling occurs if this margin vanishes and the leg protrudes into the ground ($y_m < \Delta y_g$). In steady-state locomotion with $y_{i+1} = y^* = 1.1$ m, the time-based control has a margin of at least 29 cm [at $t = 0$, $\alpha \leq 54^\circ$, see Fig. 3(b)]. Since the errors in system energy produce apex height tracking errors, the actual margin deviates from the steady-state value depending on the disturbance encountered in the previous step. In the worst case, the previous step causes the maximum observed undershoot of $\epsilon_y = -10$ cm [see Fig. 5(c)], reducing the margin for the next step to

$$y_m = y^* + \epsilon_y - l_0 \sin\alpha[0, d\theta^*, y^*] \approx 19\,\text{cm}. \qquad (10)$$

The model would have stumbled in this next step if there had been a rise in ground level of $\Delta y_g > 19$ cm. Thus, (10) defines the actual stability margin in the example.

Besides increasing the target height to create a bouncier gait, the stability margin can also be improved by decreasing the initial angle of attack. In fact, at $t = 0$, the control tables allow a minimum of $\alpha = 45^\circ$ that aligns with the friction limit (see Fig. 3). As $l_0 \sin 45^\circ = 0.71$ m, a modified time policy that starts with this angle will, at the expense of an unavoidable deflection in the largest upward steps, increase the stability margin by 10 cm to $\Delta y_{\max} \sim \pm 29$ cm, or about 30% of the leg length.

C. Sensitivity to Changes in Dynamic and Control Parameters

In further simulation experiments, we test the sensitivity of the time-based control strategy to systematic errors in model parameters. Like the system energy E_{sys}, a change in any other element of the parameter vector $\hat{\boldsymbol{p}}$ causes a deviation in the behavior function $f(y_i, \boldsymbol{u}, \hat{\boldsymbol{p}})$, producing height and deflection errors. Although $\hat{\boldsymbol{p}}$ includes five parameters, we take advantage of the fact that they collapse to only two independent parameter groups in dimensional analysis (dimensionless system energy $\tilde{E}_{\text{sys}} = \frac{E_{\text{sys}}}{mgl_0}$ and stiffness $\tilde{k} = \frac{kl_0}{mg}$) [11], and we only consider changes in system energy ΔE and leg stiffness Δk to characterize the sensitivity of the leg-placement strategy to systematic errors in the dynamic parameters. In addition, we consider systematic errors in the control with regard to the positioning accuracy of the leg ($\Delta\alpha$ and $\Delta\beta$) and the update frequency of the control loop (time delay Δt). For each parameter, we first repeat the random waypoint navigation experiment over flat ground to isolate its effect on the control performance and then investigate its influence in rough-terrain locomotion.

Table I summarizes the results of the sensitivity analysis on flat terrain. With no parameter disturbances, very small tracking errors in height and deflection persist due to the discrete representation of the behavior function f (maximum errors $\leq$0.5 cm and $<3.5^\circ$ for the sharpest turns). With parameter disturbances, the model is most sensitive to systematic changes in system energy and the angle of attack, producing maximum errors of 36 cm and 20° ($\Delta E = 10\%$) and 17 cm and 12° ($\Delta\alpha = 2^\circ$). On average, however, tracking errors stay much smaller. Moreover, if the controller is constrained to make wider turns, the maximum errors are reduced substantially (commanded deflections of $|d\theta^*| > 20^\circ$ are truncated to a maximum turning rate of $d\theta^* = \pm 20^\circ$ per step for "constrained deadbeat" in Table I), demonstrating that accuracy in modeling and tracking matter the most for aggressive turning maneuvers.

As a characteristic example of the influence of systematic parameter changes on control performance during rough-terrain locomotion, Fig. 7 shows the combined effects of systematic errors in energy estimation and energy disturbances from rough ground. As mentioned in Section IV-A, changing the initial system energy has the same effect as changes in ground height; it recenters the tracking errors about a new ground height where the step change ΔE_i in energy cancels the systematic error, $\Delta E_i = -\Delta E$. For example, with a systematic change of $\Delta E = \pm 3\%$, the control is perfect at a ground height of $y'_{g,i} = \pm 7$ cm and the energetic errors encountered now range in [-1.5%, 7.5%] [blue dots in Fig. 7(a) and (b)] and [-7.5%, 1.5%] (red dots) instead of [-4.5%, 4.5%] [shaded gray area in (a) and black dots in (b)]. Apart from this horizontal shift along the height or energy axis, the shape of the original tracking error function (gray regions in Figs. 5(c) and 7) remains unchanged. (Other systematic parameter disturbances have similar effects on rough-terrain locomotion, systematically deforming the original error function. Their plots are omitted here.)

The results of the sensitivity analysis show that for an implementation in running robots, automatic model calibration may be required as demonstrated by Arslan *et al.* [24] for a planar hopper that derives its control from the deadbeat control of the planar spring–mass model. The clearly systematic trends in the steady-state errors for each disturbance suggest that such tuning will be effective.

D. Comparison With the Classical Leg-Placement Strategy

Classical approaches to the control of hopping or running robots often rely on the Raibert controller [2], [5]. This controller has three components including a thrust law for energy

TABLE I
SENSITIVITY OF TRACKING ON FLAT GROUND

Systematic Error	Deadbeat / Constrained Deadbeat / Classical														
	steady state ϵ_y (cm)			RMS ϵ_y (cm)			max ϵ_y (cm)			RMS $\epsilon_{d\theta}$ (deg)			max $\epsilon_{d\theta}$ (deg)		
0	-0.3	**-0.3**	-1.2	0.3	**0.3**	6.7	0.5	**0.5**	19.8	0.9	**0.0**	3.8	3.3	**0.0**	11.2
$\Delta E = -10\%$	+15.2	+15.2	**-0.6**	15.7	15.1	**5.2**	27.4	16.0	**15.6**	4.0	3.6	**2.4**	15.4	**6.9**	8.8
-5%	+5.9	+5.9	**-1.0**	5.7	**5.8**	6.0	6.8	**6.1**	17.6	3.2	**1.5**	3.2	6.0	**2.6**	9.1
+5%	-4.2	-4.2	**-1.5**	4.0	**4.0**	7.6	4.9	**5.3**	23.1	2.9	**1.2**	4.4	9.8	**1.9**	12.9
+10%	-7.4	-7.4	**-2.2**	7.0	**7.1**	8.6	35.9	**12.9**	25.3	4.6	**2.3**	5.2	19.3	**3.5**	15.8
$\Delta k = -10\%$	-7.5	-7.5	**-1.5**	7.8	7.5	**6.9**	9.8	**7.9**	26.4	1.5	**0.6**	5.1	4.8	**1.0**	14.6
-5%	-3.7	-3.7	**-1.4**	3.9	**3.7**	6.8	4.8	**3.9**	22.8	1.1	**0.2**	4.3	5.2	**0.5**	13.0
+5%	+3.6	+3.6	**-1.3**	3.7	**3.9**	6.8	4.6	**4.1**	17.7	0.3	**0.3**	3.3	1.4	**0.5**	9.9
+10%	+7.2	+7.2	**-1.2**	7.4	7.7	**6.9**	8.4	**8.4**	16.4	0.6	**0.4**	2.9	2.7	**0.8**	8.5
$\Delta\alpha = -2^\circ$	+10.9	+10.9	**-1.5**	10.1	9.8	**7.1**	16.8	**11.0**	16.8	2.4	**2.3**	4.2	7.6	**3.9**	13.7
-1°	+5.0	+5.0	**-1.4**	4.9	**4.5**	6.8	6.2	**5.1**	24.2	0.9	**1.0**	4.0	3.1	**1.8**	12.4
$+1^\circ$	-4.4	-4.4	**-1.3**	4.2	**3.9**	6.7	5.0	**4.4**	17.2	2.6	**1.0**	3.6	8.9	**1.6**	9.9
$+2^\circ$	-8.1	-8.1	**-1.3**	7.6	6.9	**6.8**	9.8	**8.0**	15.3	3.9	**2.2**	3.3	12.2	**3.1**	8.7
$\Delta\beta = \pm1^\circ$	-0.1	**-0.1**	-1.3	2.8	**1.7**	6.3	7.1	**2.7**	22.2	1.7	**0.8**	4.2	5.2	**1.0**	11.5
$\pm2^\circ$	+0.3	**+0.3**	-1.4	5.0	**3.3**	7.8	14.4	**4.9**	27.7	2.3	**1.6**	4.2	5.5	**1.9**	15.4
$\Delta t = 15$ ms	+3.9	+3.9	-	3.9	4.9	-	4.9	6.4	-	1.5	1.5	-	5.1	2.8	-
30 ms	+9.3	+9.3	-	9.2	9.7	-	13.3	10.9	-	2.2	3.2	-	10.1	5.3	-

Maximum and Root-Mean-Square Errors in Tracking Height (ϵ_y) and Deflection ($\epsilon_{d\theta}$) are Shown for the Deadbeat, Turn-Rate Constrained Deadbeat (Section IV-C), and Classical Controllers (Section IV-D). “Steady State” Refers to Straight-Line Motion With $y_i = y_{i+1}$. Smaller Errors Between Constrained Deadbeat and Classical Control are in Bold.

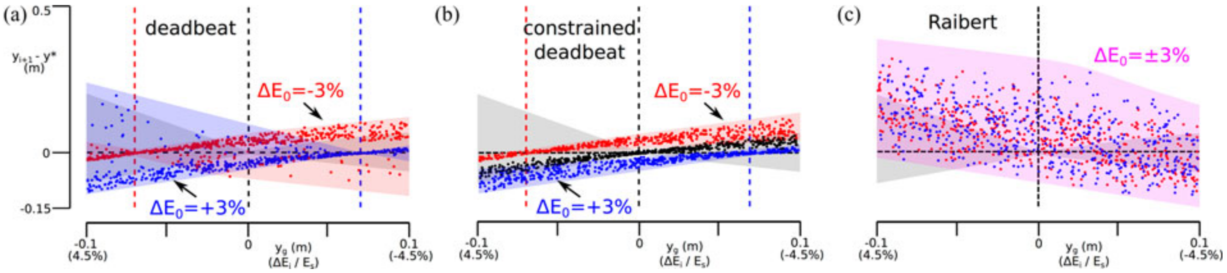

Fig. 7. Sensitivity of tracking apex height on rough ground to systematic changes in energy $\Delta E = \pm 3\%$ (blue/red dots). Tracking errors are shown as a function of ground level. (a) Time-embedded deadbeat control. The gray envelope outlines the tracking error of the unperturbed model [see Fig. 5(c)]. Systematic model errors produce systematic shifts in tracking error including the zero error height (vertical lines). (b) Same control limited to maximum turning rate of 20° per step. The black dots show the tracking error of the corresponding unperturbed model. (c) Classical Raibert-style leg-placement control (limited to 20° per step to avoid falling). Tracking errors remain large but nearly unaffected by systematic modeling errors (compare with red dots in Fig. 8(c). Similar effects are observed in the deflection errors $\epsilon_{d\theta}$ (not shown).

regulation in stance, a stance-phase attitude control for the robot center body, and a leg-placement control in flight for the generation of gait and turning. To compare the quality of the control proposed here with existing control approaches used in legged robotics, we focus on the leg-placement component of the Raibert controller (the spring–mass model is energetically conservative and has a point mass body without pitch) and redo the robustness and sensitivity tests.

The leg placement of the classical controller uses the horizontal velocity $\dot{\boldsymbol{x}} = [\dot{x}\ \dot{z}]^T$ and the duration T_s of the previous stance to estimate a “neutral foot point” for a symmetric stance phase, and then applies a linear correction about this point based on the desired velocity $\dot{\boldsymbol{x}}_d = [\dot{x}\ \dot{z}]_d^T$. The resulting leg placement is given by

$$\boldsymbol{x}_f = \frac{\dot{\boldsymbol{x}} T_s}{2} + k_{\dot{x}}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d) \qquad (11)$$

where $\boldsymbol{x}_f = [x\ z]_f^T$ is the horizontal distance vector of the foot point from the hip joint, and $k_{\dot{x}}$ is a proportional gain. Within the theoretical framework developed in this paper, $\boldsymbol{x}_f$ is equivalent to the control input $\boldsymbol{u} = [\alpha\ \beta]$, $\dot{\boldsymbol{x}}$ describes the x-direction of the apex frame and an initial apex height $y_i = \frac{E_{\text{sys}}}{mg} - \frac{|\dot{\boldsymbol{x}}|}{2g}$, and the pair $(|\dot{\boldsymbol{x}}_d|, \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}})$ translates into the desired state $(y^*, d\theta^*)$. Thus, (11) can be interpreted as a policy for approximating the behavior function f by linearizing it about the points where $y_{i+1} = y_i$ along the zero turning axis ($\beta = d\theta = 0$, see Fig. 2). While the neutral point adapts based on the previous stance phase, the slope $k_{\dot{x}}$ of the linearization remains constant, requiring a single compromised approximation at different locations of f.

The comparison of the two leg-placement strategies shows that the time-embedded leg-placement strategy enables much sharper turns and tracks the target state faster and more accurately (see Fig. 8). By design, the time-embedded deadbeat control (black dots) generates virtually no deviation from the target state for running and turning on flat ground [see Fig. 8(b)]. In contrast, the classical approach (red dots) needs to be constrained to a maximum turning rate of 20° per step to avoid falling [compare Figs. 8(a) and 5(a)], gets attracted away from the target state during turns because the linearization gain $k_{\dot{x}}$ cannot simultaneously fit turning and straight-line running, and only slowly returns to the target state after turns due to a slanted return map [see Fig. 8(b)]. This difference in performance also shows on rough ground where the deviations from the target

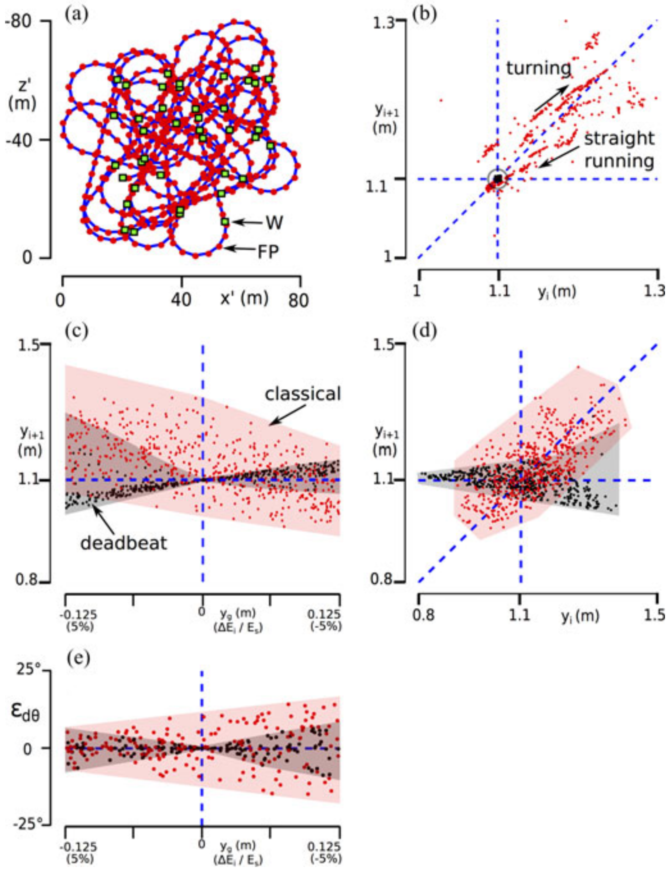

Fig. 8. Comparison with the classical leg-placement strategy. (a) Random navigation trail example generated by the leg-placement strategy of the Raibert controller with limit on turning rate of 20° per step to avoid falling. (b) Apex return maps generated from running and turning on flat ground for the classical strategy (red dots) and the time-embedded deadbeat strategy (black dots in a very tight cluster). Steady-state locomotion corresponds to dots on the diagonal line ($y_{i+1} = y_i$). (c) Errors in apex height for running and turning in rough terrain as a function of ground height or system energy. (d) Corresponding rough-ground return maps. (e) Errors in deflection over rough terrain as a function of ground height or energy.

state are large and the return to the target state remains slow [see Fig. 8(c) and (d)] for the classical control. Likewise, the directional tracking is also looser with the classical control [see Fig. 8(e)].

For systematic errors in the model parameters, the performance of the classical control does not change significantly. In general, it is still worse than that of the time-embedded strategy with the same constraint of 20° per step on the maximum turning rate (RMS and max error values in Table I) due to the slower speed of recovery in the return map. However, unlike the time-embedded strategy, the classical control shows no large drift from the target state in steady locomotion (steady-state errors in Table I) due to the feedback properties of (11), which tend to correct for unmodeled behavior.

V. Conclusion

Using a compact representation of the 3-D spring–mass model's dynamics, we have derived and demonstrated in simulation a time-based leg-placement strategy for highly robust running and steering in uncertain 3-D environments. The identified control strategy generalizes the time-based control derived for the planar spring–mass model [21], [22] and allows a legged system to navigate very rough terrain (up to 30% of the leg length in the presented example) without feedback about ground disturbances (see Fig. 4). It is nearly deadbeat with respect to stabilizing the apex height and heading direction, as long as the relative disturbances in system energy and the other model parameters stay small (see Figs. 5 and 7, Table I). Moreover, it outperforms a classical leg-placement strategy in terms of turning rate and disturbance rejection (see Fig. 8).

The compactness and performance of the proposed control strategy stems in large part from separating the control into feedback detection of the direction of motion and feedforward compensation for unknown ground disturbances. Like previous work on the control of 3-D spring–mass running, the leg-placement strategy developed here requires state feedback about the direction of motion. In [28] and [30], this feedback information is used explicitly. Although Peuker *et al.* [29] find that control relative to the velocity vector in flight yields inherent stability without the need for corrective strategies, it depends on measuring this state at least intermittently. The control developed here requires a similar intermittent feedback. At every apex during flight, the direction of motion needs to be known to align the apex frame. However, the robustness to ground level disturbances is realized with feedforward control. Instead of correcting for past disturbances, the time-indexed control adapts the leg placements to accommodate ground variations in the upcoming step by taking advantage of the relationship between falling time and falling height. The resulting control strategy reveals how highly robust running and steering could be achieved in uncertain environments.

While our results suggest a powerful alternative for leg-placement strategies in running robots, an actual implementation will require further work. On one hand, separating the control between global steering and relative motions within steps (see Section II-B) facilitates hardware implementation. The separation decouples planning from stabilizing motions. It simplifies the representation of deadbeat policies to the essential local behaviors of stability and steering, which can be embedded in real time with low-dimensional lookup tables, and it reduces the state estimation requirements to detect the direction of gravity, the direction of motion, and the apex event. (All three estimates can be obtained from inertial measurement units, the last two can also be obtained from body-centric sensors at takeoff. These sensing requirements are consistent with other placement strategies in the literature.) Finally, the control does not need ground level information.

On the other hand, the reliance on precomputed leg-placement strategies derived from the spring–mass model introduces implementation challenges. Automatic parameter calibration of the spring–mass model [24] may be required to match the behavior function to a running robot, and model extensions that regulate energy [2], [5], [23], [38]–[40] will be necessary to handle sloped terrain and dissipative losses. Furthermore, the model trades realism for mathematical and computational tractability. It neither considers actual swing leg dynamics nor does it account for the pitch, roll, and yaw moments generated by the legs about the trunk. Several studies have attacked these problems by taking advantage of passive mechanical properties, using local feedback controls, and developing reduction-based control algorithms [2], [5], [14], [25], [41], [42]. Specifically, for planar locomotion, we have recently shown that the leg-placement policy derived from the spring–mass model yields similar robustness in more realistic robot models with rigid body dynamics and continuous ground contacts [43]. Future work will need to clarify how well these methods extend to the high speeds and aggressive maneuvers that the 3-D spring–mass model allows. Alternatively, it may be possible to extend the theory developed here to include some of the additional dynamics in similar feedforward policies.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful comments and suggestions, which led to substantial improvements to the paper.

References

[1] M. Raibert, H. Brown, M. Chepponis, E. Hastings, S. Murthy, and F. Wimberly, "Dynamically stable legged locomotion," The Robotics Inst., Carnegie–Mellon Univ., Pittsburgh, MA, USA, Tech. Rep. CMU–RI–TR–83–1, 1983.

[2] M. Raibert, *Legged Robots That Balance*. Cambridge, MA, USA: MIT Press, 1986.

[3] R. Blickhan, "The spring-mass model for running and hopping," *J. Biomech.*, vol. 22, no. 11-12, pp. 1217–1227, 1989.

[4] T. McMahon and G. Cheng, "The mechanism of running: How does stiffness couple with speed?" *J. Biomech.*, vol. 23, pp. 65–78, 1990.

[5] G. Zeglin and B. Brown, "Control of a bow leg hopping robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1998, vol. 1, pp. 793–798.

[6] R. Alexander, "Three uses for springs in legged locomotion," *Int. J. Robot. Res.*, vol. 9, no. 2, pp. 53–61, 1990.

[7] R. Blickhan and R. Full, "Similarity in multilegged locomotion: Bouncing like a monopode," *J. Comp. Physiol.*, vol. 173, pp. 509–517, 1993.

[8] C. Farley, J. Glasheen, and T. McMahon, "Running springs: Speed and animal size," *J. Exp. Biol.*, vol. 185, pp. 71–86, 1993.

[9] M. Dickinson, C. Farley, R. Full, M. Koehl, R. Kram, and S. Lehman, "How animals move: An integrative view," *Science*, vol. 288, no. 5463, pp. 100–106, 2000.

[10] W. J. Schwind and D. E. Koditschek, "Approximating the stance map of a 2-DOF monoped runner," *J. Nonlinear Sci.*, vol. 10, pp. 533–568, 2000.

[11] H. Geyer, A. Seyfarth, and R. Blickhan, "Spring-mass running: Simple approximate solution and application to gait stability," *J. Theoret. Biol.*, vol. 232, no. 3, pp. 315–328, 2005.

[12] U. Saranli, Ö. Arslan, M. M. Ankaral, and Ö. Morgül, "Approximate analytic solutions to non-symmetric stance trajectories of the passive spring-loaded inverted pendulum with damping," *Nonlinear Dyn.*, vol. 62, no. 4, pp. 729–742, 2010.

[13] P. Holmes, R. J. Full, and D. E. Koditschek, "The dynamics of legged locomotion: Models, analyses, and challenges," *SIAM Rev.*, vol. 48, no. 2, pp. 207–304, 2006.

[14] J. Schmitt and P. Holmes, "Mechanical models for insect locomotion: Dynamics and stability in the horizontal plane I. theory," *Biol. Cybern.*, vol. 83, pp. 501–515, 2000.

[15] J. Schmitt and P. Holmes, "Mechanical models for insect locomotion: Dynamics and stability in the horizontal plane II. application," *Biol. Cybern.*, vol. 83, pp. 517–527, 2000.

[16] J. Schmitt, M. Garcia, R. C. Razo, P. Holmes, and R. J. Full, "Dynamics and stability of legged locomotion in the horizontal plane: A test case using insects," *Biol. Cybern.*, vol. 86, no. 5, pp. 343–53, 2002.

[17] R. Altendorfer, R. Ghigliazza, and P. Holmes, "Exploiting passive stability for hierarchical control," in *Proc. 5th Int. Conf. Climb. Walk. Robot.*, Paris, France, 2002, pp. 177–184.

[18] A. Seyfarth, H. Geyer, M. Günther, and R. Blickhan, "A movement criterion for running," *J. Biomech.*, vol. 35, pp. 649–655, 2002.

[19] R. Altendorfer, D. E. Koditschek, and P. Holmes, "Towards a factored analysis of legged locomotion models," in *Proc. Int. Conf. Robot. Autom.*, Taipei, Taiwan, 2003, pp. 37–44.

[20] R. Ghigliazza, R. Altendorfer, P. Holmes, and D. Koditschek, "A simply stabilized running model," *SIAM J. Appl. Dynam. Syst.*, vol. 2, no. 2, pp. 187–218, 2003.

[21] A. Seyfarth and H. Geyer, "Natural control of spring-like running–optimized self-stabilization," in *Proc. 5th Int. Conf. Climb. Walk. Robot.*, 2002, pp. 81–85.

[22] A. Seyfarth, H. Geyer, and H. M. Herr, "Swing-leg retraction: A simple control model for stable running," *J. Exp. Biol.*, vol. 206, pp. 2547–2555, 2003.

[23] J. Schmitt and J. Clark, "Modeling posture-dependent leg actuation in sagittal plane locomotion," *Bioinspir. Biomim.*, vol. 4, no. 4, p. 046005, 2009, doi:10.1088/1748-3182/4/4/046005.

[24] O. Arslan, U. Saranli, and O. Morgul, "Reactive footstep planning for a planar spring mass hopper," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Oct. 2009, pp. 160–166.

[25] M. M. Ankarali and U. Saranli, "Control of underactuated planar pronking through an embedded spring-mass hopper template," *Autonom. Robot.*, vol. 30, no. 2, pp. 217–231, 2011.

[26] B. Andrews, B. Miller, J. Schmitt, and J. E. Clark, "Running over unknown rough terrain with a one-legged planar robot," *Bioinspir. Biomim.*, vol. 6, no. 2, 2011, doi:10.1088/1748-3182/6/2/026009.

[27] M. Ernst, H. Geyer, and R. Blickhan, "Extension and customization of self-stability control in compliant legged systems," *Bioinspir. Biomim.*, vol. 7, no. 4, 2012, doi:10.1088/1748-3182/7/4/046002.

[28] J. Seipel and P. Holmes, "Running in three dimensions: Analysis of a point-mass sprung-leg model," *Int. J. Robot. Res.*, vol. 24, no. 8, pp. 657–674, 2005.

[29] F. Peuker, C. Maufroy, and A. Seyfarth, "Leg-adjustment strategies for stable running in three dimensions," *Bioinspir. Biomim.*, vol. 7, no. 3, 2012, doi:10.1088/1748-3182/7/3/036002.

[30] G. S. 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, pp. 026 106-1–026 106-14, 2009.

[31] F. Peuker and A. Seyfarth, "Adjusting legs for stable running in three dimensions," in *6th World Congress Biomechanics* (ser. IFMBE Proceedings), C. T. Lim, J. C. H. Goh, and R. Magjarevic, Eds. Berlin, Germany: Springer-Verlag, 2010, vol. 31, pp. 3–6.

[32] O. Khatib, L. Sentis, and J.-H. Park, *A Unified Framework for Whole-Body Humanoid Robot Control With Multiple Constraints and Contacts* (ser. Springer Tracts in Advanced Robotics). Berlin, Germany: Springer-Verlag, 2008, vol. 44, pp. 303–312.

[33] M. A. Townsend. (1985). Biped gait stabilization via foot placement. *J. Biomech.* [Online]. *18(1)*, pp. 21–38. Available: http://www.sciencedirect.com/science/article/pii/0021929085900429

[34] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2001, vol. 1, pp. 239–246.

[35] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Proc. 6th IEEE-RAS Int. Conf. Humanoid Robot.*, Dec. 2006, pp. 200–207.

[36] M. A. Daley and A. A. Biewener, "Running over rough terrain reveals limb control for intrinsic stability," *Proc. Natl. Acad. Sci. USA*, vol. 103, no. 42, pp. 15 681–15 686, 2006.

[37] D. G. E. Hobbelen and M. Wisse, "Swing-leg retraction for limit cycle walkers improves disturbance rejection," *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 377–389, Apr. 2008.

[38] H. Geyer, A. Seyfarth, and R. Blickhan, "Positive force feedback in bouncing gaits?" *Proc. R. Soc. Lond. B, Biol. Sci.*, vol. 270, pp. 2173–2183, 2003.

[39] M. M. Ankarali and U. Saranli, "Stride-to-stride energy regulation for robust self-stability of a torque-actuated dissipative spring-mass hopper," *Chaos*, vol. 20, no. 3, pp. 033 121-1–033 121-13, 2010.

[40] D. Koepl and J. W. Hurst, "Force control for planar spring-mass running," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Sep. 2011, pp. 3758–3763.

[41] U. Saranli and D. Koditschek, "Template based control of hexapedal running," in *Proc. IEEE Int. Conf. Robot. Autom.*, Taipei, Taiwan, Sep. 2003, vol. 1, pp. 1374–1379.

[42] R. D. Gregg and M. W. Spong, "Reduction-based control of three-dimensional bipedal walking robots," *Int. J. Rob. Res.*, vol. 29, no. 6, pp. 680–702, 2010.

[43] A. Wu and H. Geyer, "Deadbeat running and steering in 3D environments," in *Proc. Dyn. Walk.*, 2012. Available: http://dynamicwalking.org/dw2012/?q=proceedings

Albert Wu received the B.S. degree in mechanical engineering from the California Institute of Technology Pasadena, CA, USA, in 2009 and the S.M. degree in aeronautics and astronautics from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2011. He is currently working toward the Ph.D. degree in robotics with Carnegie Mellon University, Pittsburgh, PA, USA.

His research interests include the area of legged locomotion.

Hartmut Geyer (M'13) received the Dipl. degree in physics and the Ph.D. degree in biomechanics from the Friedrich-Schiller-University of Jena, Jena, Germany, in 2001 and 2005, respectively.

He is currently an Assistant Professor with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. His research interests include the principles of legged dynamics and control, their relation to human motor control, and resulting applications in humanoid and rehabilitation robotics.