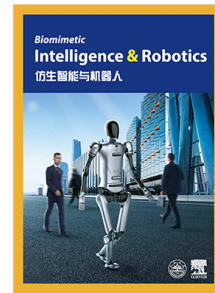


Journal Pre-proof

Humanoid robot running through random stepping stones and jumping over obstacles: Step adaptation using spring-mass trajectories

Sait Sovukluk, Johannes Engelsberger, Christian Ott



PII: S2667-3797(26)00025-2
DOI: <https://doi.org/10.1016/j.birob.2026.100297>
Reference: BIROB 100297

To appear in: *Biomimetic Intelligence and Robotics*

Received date: 11 July 2025
Revised date: 8 October 2025
Accepted date: 11 December 2025

Please cite this article as: S. Sovukluk, J. Engelsberger and C. Ott, Humanoid robot running through random stepping stones and jumping over obstacles: Step adaptation using spring-mass trajectories, *Biomimetic Intelligence and Robotics* (2026), doi: <https://doi.org/10.1016/j.birob.2026.100297>.

This is a PDF of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability. This version will undergo additional copyediting, typesetting and review before it is published in its final form. As such, this version is no longer the Accepted Manuscript, but it is not yet the definitive Version of Record; we are providing this early version to give early visibility of the article. Please note that Elsevier's sharing policy for the Published Journal Article applies to this version, see: <https://www.elsevier.com/about/policies-and-standards/sharing#4-published-journal-article>. Please also note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2026 Publishing services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Humanoid Robot Running Through Random Stepping Stones and Jumping Over Obstacles: Step Adaptation Using Spring-Mass Trajectories

Sait Sovukluk^{a,*}, Johannes Engelsberger^b, Christian Ott^{a,b}

^aAutomation and Control Institute (ACIN), TU Wien, Vienna 1040, Austria

^bInstitute of Robotics and Mechatronics, German Aerospace Center (DLR), Weßling 82234, Germany

Abstract

This study proposes a step adaptation framework for running through spring-mass trajectories and deadbeat control gain libraries. It includes four main parts: (1) Automatic spring-mass trajectory library generation; (2) Deadbeat control gain library generation through an actively controlled template model that resembles the whole-body dynamics well; (3) Trajectory selection policy development for step adaptation; (4) Mapping spring-mass trajectories to a humanoid model through a whole-body control (WBC) framework also accounting for closed-kinematic chain systems, self collisions, and reactive limb swinging. We show the inclusiveness and the robustness of the proposed framework through various challenging and agile behaviors such as running through randomly generated stepping stones, jumping over random obstacles, performing slalom motions, changing the running direction suddenly with a random leg, and rejecting significant disturbances and uncertainties through the MuJoCo physics simulator. We also perform additional simulations under a comprehensive set of uncertainties and noise to better justify the proposed method's robustness against real-world challenges, such as signal noises, imprecision, modeling errors, and delays. All the aforementioned behaviors are performed with a single library and the same set of WBC control parameters without additional tuning. The spring-mass and the deadbeat control gain library are automatically computed in 4.5 seconds in total for 315 different trajectories.

Keywords: Humanoid and Bipedal Locomotion, Whole-Body Motion Planning and Control, Legged Locomotion

1. Introduction

The spring-mass model, also known as the spring-loaded inverted pendulum (SLIP), was first introduced to model the steady-state running of animals and humans [1], in which running is defined as the center-of-mass (CoM) bouncing on a springy leg through single-leg support and flight phases. Comprehensive experimental and dimensionless mathematical analyses revealed that the ground reaction force pattern of steady-state running of humans and animals, ranging from insects to horses, can be modeled through the spring-mass model, surprisingly, with almost the same relative (dimensionless) leg stiffness (~ 10) [2, 3]. It is also revealed that ground reaction force patterns for human walking are also closely related to the spring-mass model [4]. Even though the spring-mass model is a simplification, it can generate nearly every walking and running gait observed in people and animals [5].

While a simple spring-mass model from a biomechanics perspective, SLIP exhibits many control challenges as it is a hybrid system with nonlinear stance dynamics [6]. Seyfarth et al. [7] studied the stability characteristics of planar spring-mass dynamics, and they discovered a J-shaped dependency in the adjustment of the angle of attack to leg stiffness. Similar findings triggered the development of apex-to-apex deadbeat controllers to find out how the leg states (e.g., angle, stiffness, damping)

should be modified to maintain stability and convergence during the next stance phase [6, 8, 9]. The first successful realization of the spring-mass model appeared in Raibert's hoppers, runners, and walkers [10]. ATRIAS is another bipedal robot that is carefully designed to match the spring-mass dynamics and was the first machine to demonstrate humanlike walking-gait dynamics [11]. Mapping a 3D spring-mass (SLIP) model to a full-scale robotic human model was first studied in [12] and later on [13]. It's achieved via obtaining the desired center of mass (CoM) trajectories through the spring-mass model and then commanding them to a robotic human model through a whole-body controller (WBC) while keeping the torso upright and employing proper leg and arm swinging. However, both [12] and [13] lack step adaptation and focus only on velocity control. Furthermore, as the spring-mass model does not account for articulated body dynamics, mapping such center of mass trajectories to a full-scale humanoid robot requires careful tuning of limb-swing trajectories to ensure that the articulated body and angular momentum effects are properly handled at the whole-body control level. A different center of mass trajectory may require a different limb-swing behavior due to changes in the force profiles, velocities, stance time, and flight time, preventing the utilization of various CoM trajectories for step adaptation purposes through the same limb-swing control. In this study, we address both the step adaptation and reactive limb-swing trajectory generation problems, enabling navigation through random environments through switching between various CoM trajectories without any trajectory-specific tuning.

*Corresponding author.

Email address: sovukluk@acin.tuwien.ac.at (Sait Sovukluk)

1 The framework’s agility and disturbance rejection make it suit-
 2 able for outdoor challenges, such as search-and-rescue or in-
 3 dustrial inspection in unstructured environments.

4 The literature on dynamic humanoid running is largely lack-
 5 ing in step adaptation and appears more limited compared
 6 to studies on walking. Furthermore, different from walking,
 7 where there is always contact with the ground, running is
 8 more demanding in terms of power outputs, peak forces, im-
 9 pacts, state estimation, and contact conditions as the robotic
 10 system alternates between the stance and flight phases on a
 11 single leg, requiring exceptional hardware that is also precise
 12 and impact-resistant. Consequently, most of the full-scale hu-
 13 manoid running literature appears in simulation environments
 14 [14, 15, 16, 17, 18, 19, 20], and the experimental verifications
 15 are usually limited to velocity-based running with insignificant
 16 flight phases and without step adaptation [21, 22, 23, 24]. The
 17 recent machine learning literature reveals that reinforcement
 18 learning-based methods for running are mostly based on imi-
 19 tation learning, which requires a strong reference from human
 20 running [25, 26, 23]. As imitation is a mapping from human
 21 data, the generalization of running behaviors for random step
 22 adaptation has not yet been demonstrated.

24 Dynamic humanoid running with step adaptation and distur-
 25 bance rejection capabilities without prior environmental knowl-
 26 edge is an understudied topic that has not been thoroughly ad-
 27 dressed in the literature. The combination of the nonlinear and
 28 hybrid nature of running dynamics with foothold constraints
 29 presents a challenging problem that requires special care. This
 30 problem is easier and has already been addressed in the case of
 31 walking [27]. Walking dynamics is more flexible, stable, and
 32 easier to solve, and also allows static walking to preserve stabil-
 33 ity through balancing, as there is always at least one leg on the
 34 ground. There are two running frameworks appearing with step
 35 adaptation capability: BID [28] and DCM-based running [29].
 36 Different from the spring-mass model, where the dynamics are
 37 nonlinear and not closed-form solvable, both methods provide
 38 closed-form solvable linear running dynamics with a trade-off
 39 of preselection requirement of some stability- and feasibility-
 40 critical parameters, including but not limited to: stance time,
 41 flight time, and touchdown height, along with the desired foot
 42 placement point. Furthermore, improper stance time selection
 43 results in leg overextension as the CoM moves far away from
 44 the foot placement point during the stance phase. Similarly, the
 45 touchdown height selection also affects stability and may result
 46 in violating the friction constraint. Hence, the feasibility and
 47 stability in both studies depend on manual parameter selections,
 48 requiring a trial-and-error process for different environments,
 49 running velocities, and types. BID-based planning includes
 50 a disturbance rejection controller, whereas DCM-based plan-
 51 ning does not. The proposed step adaptation framework, on the
 52 other hand, is tuning-free and only requires selection for the de-
 53 sired range for running velocity, apex height, and leg stiffness.
 54 Compared to [28]’s 2016 BID and [29]’s 2023 DCM running,
 55 our work eliminates manual parameter tuning for stance/flight
 56 times and determines the stability critical parameters inherently,
 57 also considering feasibility, stability, disturbances, errors, and
 58 desired foot placement points. As a result, the proposed step
 59
 60
 61
 62
 63
 64
 65

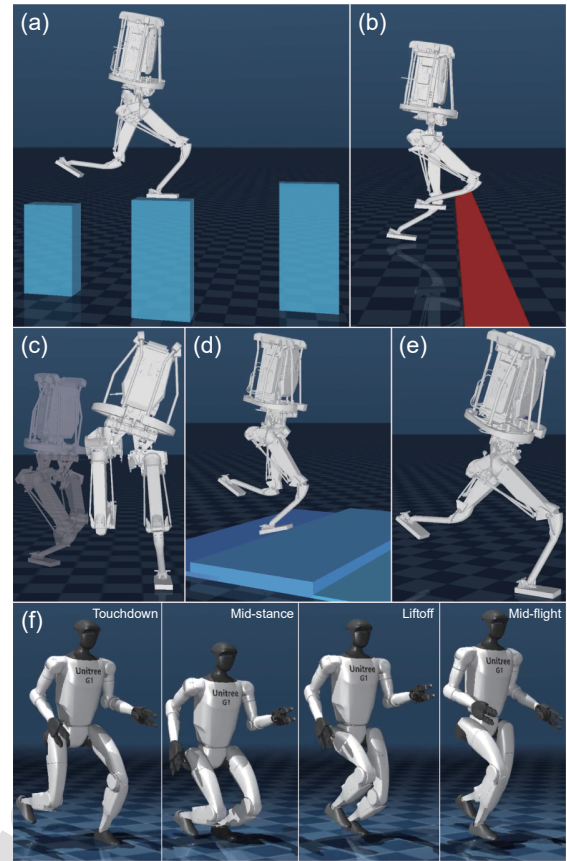


Figure 1: (A) Running through randomly generated stepping stones; (B) Jumping over randomly generated obstacles; (C) Sudden direction change with the inner leg; (D) Running through unobserved height differences; (E) Rejecting a forward push disturbance with a wide step; (F) Snapshots of Unitree G1 running. Multiple robots are presented to show the generalizability of the proposed framework. The supplemental video collects all and more.

adaptation framework with reactive limb-swing capabilities en-
 ables real-time adaptation to random stepping stones without
 prior environmental information.

1.1. Contribution

- The spring-mass model is a velocity-based running model, i.e., the error is defined in terms of velocity and jumping height, and is constructed around periodic trajectories. Hence, achieving step adaptation to run through random stepping stones is the main contribution of this paper. The step adaptation is achieved through switching between different trajectories contained in an automatically generated trajectory library, which takes ~ 4.5 seconds to generate for 315 different trajectories. We first develop the necessary conditions for convergence between periodic trajectories and analyze the minimum number of steps required for planning. Then, develop a trajectory selection policy that switches between adequate trajectories depending on the foot placement point requirements, considering stability, kinematic and dynamic feasibility, and the convergence conditions. The proposed method is carefree, and the sta-

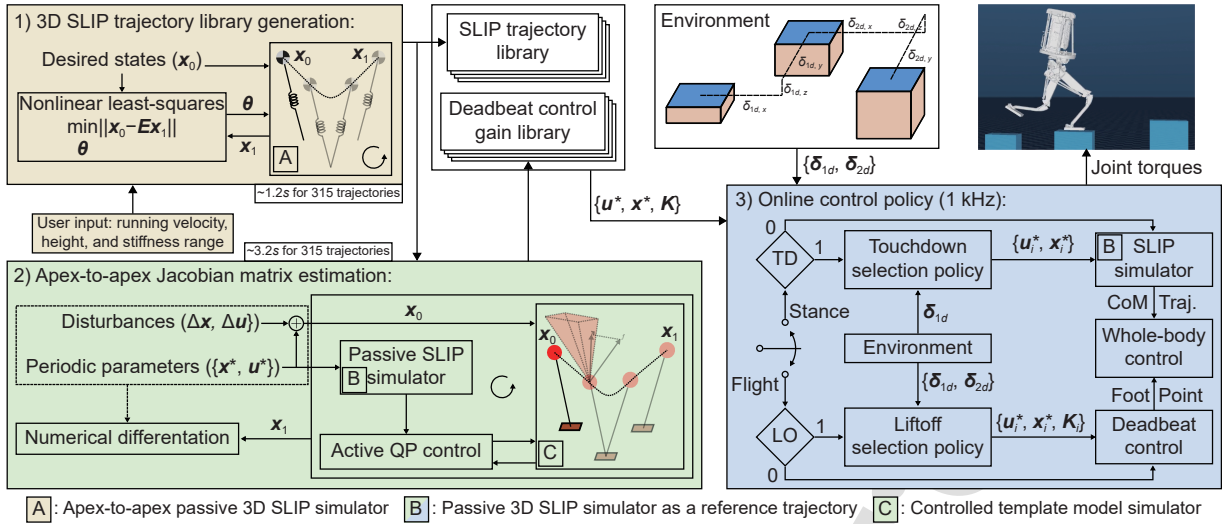


Figure 2: Overview of the proposed framework. Three main steps include (1) 3D SLIP periodic trajectory search, (2) apex-to-apex Jacobian estimations through an actively controlled system that resembles the whole-body dynamics well, and (3) online control that combines trajectory selection policies and the whole-body controller with a reactive limb swing task.

bility critical parameters, such as stance time, flight time, and touchdown angle, are determined inherently.

- The CoM trajectories, that are obtained from reduced or template models, are realized on full-scale humanoid robots through whole-body controllers as a CoM trajectory tracking task combined with other tasks such as torso orientation, leg swing, and arm swing control through hand-crafted limb swing trajectories [12, 13, 28, 29]. However, in the case of frequent trajectory switching and disturbances, where the force profiles, stance time, flight time, and foot placement conditions vary, a different limb-swing behavior is required to maintain postural stability. We address this problem through a reactive limb swing control task that utilizes dynamic coupling between the links and the base frame orientation, thereby enhancing disturbance rejection capabilities. We utilize the redundant leg and arm joints and control their states depending on the base frame orientation error, such that the postural stability is preserved while the robot follows the given CoM trajectories and foot placement point commands. The extensive set of behaviors proposed in this study (also shown in Fig. 1 and the supplemental video) is realized using a single library and the same whole-body controller, without any additional parameter or trajectory tuning.
- This work combines a diverse set of dynamic behaviors in a single framework. These behaviors include running through previously unknown, randomly generated stepping stones, jumping over suddenly appearing obstacles, sudden turns and slalom motions, and disturbance rejection, including force disturbances, unobserved height differences, and edge stepping. To further justify the proposed method's robustness against real-world challenges, we also simulate its operation under a comprehensive and significant set of uncertainties and noise, including signal

noise, modeling errors, imprecision, and delays.

1.2. Content

The flow of the paper is summarized in Fig. 2. The paper starts with a summary of the spring-mass dynamics and periodic trajectory optimization formulation. Then, it proceeds with trajectory library generation, observations on trajectory characteristics, and calculation of stepping distances. The development continues with the computation of deadbeat control gain through an actively controlled spring-mass model for stabilization, disturbance rejection, and trajectory switching purposes. The third part is related to the development of the step adaptation policy. We first analyze how many steps to look ahead to guarantee convergence in the spring-mass dynamics level. We then propose two sets of algorithms to jump over obstacles and run through stepping stones considering feasibility, stability, and convergence conditions. The fourth part includes conventional inverse-dynamics whole-body control formulations with two add-ons: a task formulation for reactive limb swing and self-collision avoidance. Then, we discuss computation times and simulation results for an extensive range of behaviors, including noise and uncertainty analysis. Lastly, a conclusion section relates the paper's objective to its methodology and results, and summarizes the outcomes.

2. 3D Spring-Mass Trajectory Library

This section involves constructing a nonlinear least-squares problem to solve for periodic spring-mass trajectories. Then, this structure is used to generate a trajectory library that combines different types of running in terms of apex height, velocity, stiffness, and lateral leg angle. This section corresponds to the first block of Fig. 2 and discusses mapping spring-mass trajectories to foot placement point information.

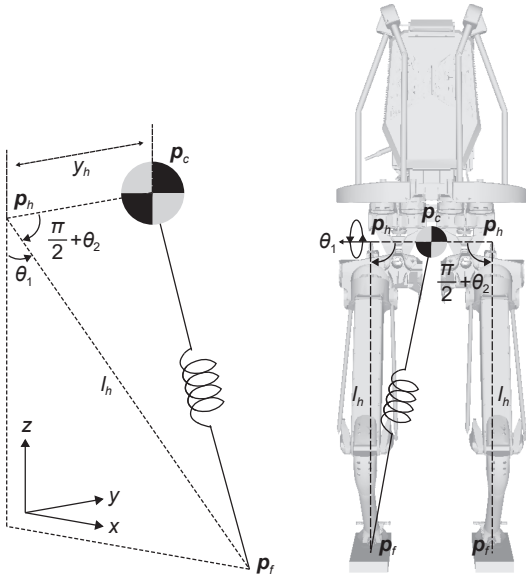


Figure 3: 3D SLIP model representation. Parameter y_h separates the robot's center of mass and hip positions. The virtual leg has a length of l_h and connects the hip to the foot. The touchdown angle is determined by θ_1 and θ_2 . As the virtual leg is offset by y_h , if θ_2 is zero, the foot is in the hip sagittal (parasagittal) plane. Otherwise, the lateral foot placement point is wider or narrower than the virtual hip point y_h .

2.1. Background: 3D Spring-Mass Dynamics

Defining $\mathbf{p}_c \in \mathbb{R}^3$ to be the position of CoM, the stance dynamics of the point foot 3D spring-mass model (see Fig. 3) is

$$m\ddot{\mathbf{p}}_c = k(r_0 - \|\mathbf{r}\|)\hat{\mathbf{r}} + m\mathbf{g}, \quad (1)$$

where m , k , r_0 , and \mathbf{g} are the total mass, spring constant, leg rest length, and gravitational vector. Additionally, $\mathbf{r} = \mathbf{p}_c - \mathbf{p}_f$ represents the leg vector, i.e., a vector from the foot point to the point mass (see Fig. 3) and $\hat{\mathbf{r}}$ is its unit vector. Through the same notation with [12] and [13], the virtual hip point is located at \mathbf{p}_h and it helps to decompose leg angles θ_1 and θ_2 into more intuitive quantities. During the flight phase, the foot point \mathbf{p}_f is a function of \mathbf{p}_c , leg angles, hip distance, and leg length,

$$\mathbf{p}_f = \mathbf{p}_c + \begin{bmatrix} 0 \\ \sigma y_h \\ 0 \end{bmatrix} + l_h \begin{bmatrix} \sin \theta_1 \cos \theta_2 \\ \sigma \sin \theta_2 \\ -\cos \theta_1 \cos \theta_2 \end{bmatrix}, \quad (2)$$

where σ is a sign multiplier and changes between $+1$ and -1 depending on which leg is the stance leg. At the moment of touchdown, \mathbf{p}_f is fixed on the ground and sets an initial condition for the stance phase dynamics (1). A touchdown and liftoff condition separate the stance and flight phase dynamics such that the overall dynamics Σ is given by

$$\Sigma : \begin{cases} \ddot{\mathbf{p}}_c = \frac{k}{m}(r_0 - \|\mathbf{r}\|)\hat{\mathbf{r}} + \mathbf{g} & \|\mathbf{r}\| < r_0 \\ \ddot{\mathbf{p}}_c = \mathbf{g} & \|\mathbf{r}\| \geq r_0 \end{cases}. \quad (3)$$

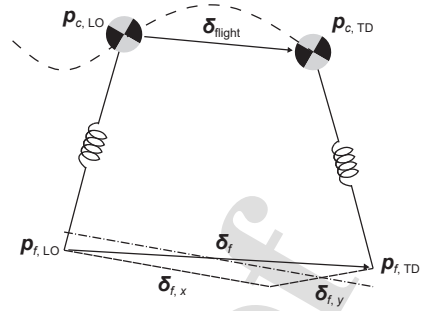


Figure 4: An illustration for foot placement distance between two consecutive steps on a constant height flat surface.

2.2. Background: Apex-to-apex Periodic Trajectory Search

Studies in [30] and [31] showed that for a constant energy level of a planar point-mass model, there exists a unique relationship between system parameters. For example, for a specific spring constant, running height, and running velocity, there exists a unique touchdown angle that results in a periodic limit cycle. Consequently, if θ_1 is the touchdown angle of a planar point-mass model, this parameter is a function of running height h , spring constant k , and running velocity v_x , i.e., $\theta_1(h, k, v_x)$. Selecting $\boldsymbol{\theta} = [\theta_1, \theta_2]^T$ and $\mathbf{v} = [v_x, v_y]^T$, one can rewrite the touchdown angle function as $\boldsymbol{\theta}(h, k, \mathbf{v})$.

For an apex state definition of $\mathbf{x} = [v_x, v_y, h]^T$ a mapping function from one apex state to the next one can be defined as $P : (\mathbf{x}_j, \boldsymbol{\theta}, k) \rightarrow \mathbf{x}_{j+1}$. Accordingly, for a selection of h , k , and \mathbf{v} , the periodic trajectory search problem can be formulated as a nonlinear least-squares problem,

$$\min_{\boldsymbol{\theta}} \|\mathbf{x}_j - E\mathbf{x}_{j+1}\|, \quad (4)$$

where $E = \text{diag}(1, -1, 1)$ is placed for sign changes due to the leg switching between two subsequent apex states. Similarly, since it is not intuitive to pick a proper value for v_y , for a selection of h , k , v_x , and θ_2 , the nonlinear least-squares problem can be reformulated:

$$\min_{\theta_1, v_y} \|\mathbf{x}_j - E\mathbf{x}_{j+1}\|. \quad (5)$$

2.3. Construction and Interpretation of Trajectory Library

A combination of different types of running can be stored in a trajectory library, allowing the robot to switch between different trajectories and adapt to changing environmental conditions. Then, to achieve step adaptation, the trajectory information must be transformed into a more useful form, namely, stepping distances. The distance between two consecutive foot placement points can be formulated as

$$\boldsymbol{\delta}_f = \underbrace{(\mathbf{p}_{c,LO} - \mathbf{p}_{f,LO})}_{r_{LO}} + \boldsymbol{\delta}_{\text{flight}} + \underbrace{(\mathbf{p}_{f,TD} - \mathbf{p}_{c,TD})}_{-r_{TD}}, \quad (6)$$

where $(\cdot)_{TD}$ and $(\cdot)_{LO}$ represent parameters at the moment of touchdown and lift-off, respectively. Additionally, $\boldsymbol{\delta}_{\text{flight}}$ represents the CoM displacement during the flight phase (see Fig. 4).

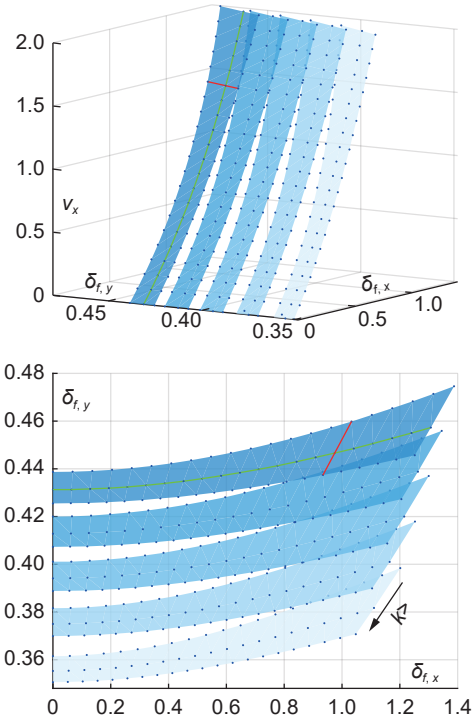


Figure 5: An example trajectory library for undisturbed periodic flat-surface running with constant θ_2 (top) and its projection onto the stepping distance ($\delta_{f,x}, \delta_{f,y}$) plane as shown in Fig. 4 (bottom). In the library, the running velocity v_x , leg stiffness k , and apex height h_{apex} vary. Each tone of blue is a collection of a constant apex height, and darker colors represent higher values. The green line is an example set of trajectories with constant stiffness and varying forward velocity. Similarly, the red line is an example set of trajectories with constant forward velocity but varying leg stiffness. Lastly, the black arrow represents the direction of increasing leg stiffness for each constant apex height surface.

As a result, a library $\mathcal{T} \in \{\mathbb{R}^{1 \times 6}, \mathbb{S}, \mathbb{S}\}^n$ with n trajectories forms the following table:

θ_2	h_{apex}	k	$v_{x,\text{apex}}$	θ_1	$v_{y,\text{apex}}$	$\delta_{f,x}$	$\delta_{f,y}$
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

where θ_2 , h_{apex} , k and $v_{x,\text{apex}}$ are selected/desired parameters; θ_1 and $v_{y,\text{apex}}$ are result of periodic trajectory search problem (5); $\delta_{f,x}$ and $\delta_{f,y}$ are result of (6). Note that $\delta_{f,x}$ and $\delta_{f,y}$ change as the flight time, surface height, leg length, and touchdown angle change. The deadbeat controller, which is proposed in the following sections, may also modify the touchdown angle and leg length for disturbance rejection purposes. Hence, $\delta_{f,x}$ and $\delta_{f,y}$ should be recalculated at touchdown and liftoff moments.

The foot placement estimation strategy differs in the touchdown and liftoff moments. On the other hand, in the case of an undisturbed periodic running, both estimations output the same result. Assume $\theta_{i,1}^*$ and $\theta_{i,2}^*$ are the periodic longitudinal and lateral leg angle of the i^{th} trajectory. As a result of the symmetricity of periodic SLIP trajectories, $\theta_{1,\text{LO}} = -\theta_{i,1}^*$ and $\theta_{2,\text{LO}} = \theta_{i,2}^*$.

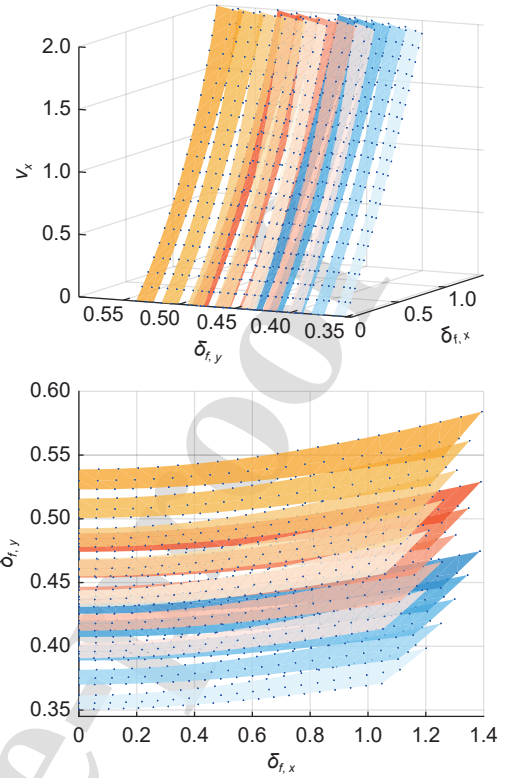


Figure 6: Extended trajectory library over Fig. 5 with varying lateral leg angle θ_2 . Blue, orange, and yellow represent increasing θ_2 in the same order.

Hence, substitution of (2) into (6) leads to

$$\delta_{f,i} = \delta_{\text{flight}} + 2 \begin{bmatrix} 0 \\ \sigma y_h \\ 0 \end{bmatrix} + 2l_h \underbrace{\begin{bmatrix} \sin(-\theta_{i,1}^*) \cos \theta_{i,2}^* \\ \sigma \sin \theta_{i,2}^* \\ -\cos(-\theta_{i,1}^*) \cos \theta_{i,2}^* \end{bmatrix}}_{r_{\text{LO}} - r_{\text{TD}}}. \quad (7)$$

Accounting for possible height changes, define total flight time $t_{\text{flight}} = t_{\text{rise}} + t_{\text{fall}}$ and the foot height w.r.t. ground at the apex state $p_{f,z,\text{apex}} = h^* - l_h \cos \theta_1^* \cos \theta_2^*$. As a result, $t_{\text{rise}} = \sqrt{2p_{f,z,\text{apex}}/g}$. Similarly, for a height difference δ_z between the consecutive steps, $t_{\text{fall}} = \sqrt{2(p_{f,z,\text{apex}} - \delta_z)/g}$ such that the fall time is longer or shorter depending on δ_z . Consequently,

$$\delta_{\text{flight}} = t_{\text{flight}} \begin{bmatrix} v_x^* \\ v_y^* \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \delta_z \end{bmatrix}. \quad (8)$$

Note that in case of remaining error at the liftoff moment, the system is not periodic anymore, and the symmetricity does not hold. At the liftoff moment, $(p_{c,\text{LO}} - p_{f,\text{LO}})$ can be calculated through the system states. Leg vector for the next touchdown $(p_{f,\text{TD}} - p_{c,\text{TD}})$, on the other hand, should be calculated accounting for the deadbeat controlled leg states.

2.3.1. Visualization of an Example Trajectory Library

Visualization of an example periodic trajectory library for a constant lateral leg angle θ_2 is shown in Fig. 5. In the library, the

parameter variations are as follows: $v_x = \{0, 0.1, \dots, 2\}m/s$, $h = \{0.90, 0.925, 0.95, 0.975, 1\}m$, $k = \{6000, 8000, 10000\}N/m$. The parameter ranges are determined considering the hardware limits and kinematics of Kangaroo. The figure shows that running with a constant apex height forms a surface relative to the remaining parameters: running velocity and stiffness. As the flight time is closely related to the apex height, for the same running velocity, the stepping distance increases proportionally with the apex height. Furthermore, as decreased leg stiffness results in more inclined touchdown and lift-off angles, foot placement distance increases as the stiffness decreases for the same running velocity. Each set of points $(\delta_{f,x}, \delta_{f,y}, v_x)$ is unique in 3D space, and their projection onto the stepping distance plane may overlap. The variation of θ_2 introduces additional depth in 3D space and new points on the $\delta_{f,x}\delta_{f,y}$ plane (see Fig. 6). The most observable characteristic of points with different lateral leg angles is the overall shift in the $\delta_{f,y}$ axis, i.e., the lateral displacement $\delta_{f,y}$ increases or decreases proportionally with θ_2 . All behaviors shown in this work are generated using the same library shown in Fig. 5.

3. Deadbeat Control of Active Spring-Mass Model

Running is a combination of two consecutive phases: the single support stance and the flight phase. As the floating base dynamics is driven by the ground reaction forces [32, 33, 34, 35], they play a vital role in locomotion. During the stance phase, the robotic system drives its joints and the ground reaction forces such that the CoM and limbs track the desired trajectories. The flight phase is fully ballistic and can be considered as a posture preparation for the following stance phase. As legged robots are not fixed but have floating base systems, the set of feasible ground reaction forces is strictly constrained by foothold, friction, and positive vertical force constraints to prevent leg tilting, sliding, and leg separation, respectively. Since the foot region limits the set of feasible ground reaction forces (see Fig. 7) and the foot location is kept constant during the stance phase, the relative foot placement point is crucial and determines an initial condition for the stance phase dynamics. The foot should be placed at a point such that the resultant feasible polyhedral cone collects the set of ground reaction forces that are needed to converge to the desired trajectory or to reject disturbances.

This section aims to obtain deadbeat control gains through a model that resembles the full-order whole-body controlled system's CoM dynamics well in terms of control force feasibility. For this purpose, instead of conventional point-foot passive 3D spring-mass dynamics where the leg stiffness is varied for compression and decompression phases for control purposes, this study employs an actively controlled CoM model with a plane foot to estimate each trajectory's apex-to-apex Jacobians and deadbeat control gains. The proposed template model inherits the active control mechanism, friction constraints, and the feasible polyhedral force cone conditions. Consequently, the estimated gains are more relatable to whole-body controlled humanoid models in terms of force and behavioral feasibility. The deadbeat control gain matrix $K \in \mathbb{R}^{3 \times 3}$ relates the flight

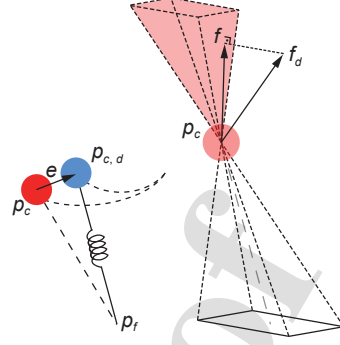


Figure 7: Left: The error definition of the control system where the blue SLIP represents the desired undisturbed periodic state evolution and the red is the actively controlled system. Right: A detailed sketch of the actively controlled template model with a foot. If the desired total control force f_d is outside the feasible polyhedral cone, it is projected to the closest feasible solution.

phase state errors to the leg angle and length offsets such that the feasible set of ground reaction forces results in eliminating errors or converging into a new trajectory during the next stance phase. The proposed model assumes that the centroidal angular momentum and additional limb controls are handled at the whole-body control level, which will be discussed in detail in the following sections.

3.1. Point Mass Model and Control Force

Let

$$m\ddot{p}_c = mg + f, \quad (9)$$

be the stance dynamics for point mass dynamics where $\{f \in \mathcal{F}^3 \mid \mathcal{F}^3 \subset \mathbb{R}^3\}$ is the set of feasible leg forces. Starting from the initial condition where both systems have the same touchdown point and integrating the undisturbed periodic SLIP dynamics, one can get the desired position, velocity, and acceleration evolution of the CoM dynamics (see Fig. 7). The difference between the actual and desired trajectories forms the error definition $e = p_{c,d} - p_c$. In the case of $f = f_d$, where,

$$f_d = m\ddot{p}_{c,d} - mg + K_D\dot{e} + K_P e, \quad (10)$$

the closed-loop dynamics is exponentially stable:

$$m\ddot{e} + K_D\dot{e} + K_P e = 0. \quad (11)$$

In the case the desired control force is not within the feasible set, i.e., $f_d \notin \mathcal{F}^3$, the control force must be projected onto the closest feasible solution. Let $V = [\hat{v}_{p_{f1}}^{p_c} \hat{v}_{p_{f2}}^{p_c} \hat{v}_{p_{f3}}^{p_c} \hat{v}_{p_{f4}}^{p_c}] \in \mathbb{R}^{3 \times 4}$ represents the combination of all corner unit vectors from foot corners to the CoM. A linear combination of the corner vectors represents all feasible forces within the foot region. Consequently, for a set of scalar multipliers $\alpha \in \mathbb{R}^4$, the overall control force formulation with the feasibility constraints takes the following form:

$$\min_{\alpha} \|f_d - V\alpha\| \quad (12)$$

Such that:

$$\alpha \geq 0, \quad (12a)$$

$$\frac{|(V\alpha)_x|}{(V\alpha)_z} < \mu, \quad \frac{|(V\alpha)_y|}{(V\alpha)_z} < \mu. \quad (12b)$$

The positive ground reaction force and friction cone constraints are handled by (12a) and (12b), respectively. If the minimization problem (12) converges to zero, then f_d is already within the polyhedral limits, e.g., $f = V\alpha = f_d$. Otherwise, $f = V\alpha$ is the closest projection. In the case of projection, the exponential stability is disturbed and convergence is not guaranteed anymore. As the polyhedral cone moves and rotates as p_f changes (see Fig. 7), a step adaptation must be performed to keep f_d inside the feasible set \mathcal{F}^3 to achieve stability and convergence.

3.2. Apex to Apex Deadbeat Control Gain Estimation

Selecting the apex state as the Poincaré section, a discrete mapping from one apex to the next one for the actively controlled point mass dynamics (9) can be defined as $P_{\text{active}} : (x_j, u_j) \rightarrow x_{j+1}$, where $x_j = [v_x, v_y, h]^T$ is apex velocities and height; $u_j = [\theta_1, \theta_2, l_h]^T$ is leg angles and virtual leg length (see Fig. 3). Assuming (x^*, u^*) results in a periodic trajectory and defining $\Delta x = x - x^*$ and $\Delta u = u - u^*$, the first-order Taylor Series approximation leads to

$$\begin{aligned} x_{k+1} &= P_{\text{active}}(x_k^* + \Delta x, u^* + \Delta u) \\ &\approx E x_k^* + J_x \Delta x + J_u \Delta u, \end{aligned} \quad (13)$$

where $J_x = \partial P_{\text{active}} / \partial x \in \mathbb{R}^{3 \times 3}$ and $J_u = \partial P_{\text{active}} / \partial u \in \mathbb{R}^{3 \times 3}$ are Jacobians of the return map evaluated at (x^*, u^*) and $E = \text{diag}(1, -1, 1)$ is placed for sign changes due to the leg switching. Deviations from periodic apex states can be rejected by changing the leg states for the next touchdown:

$$J_u \Delta u = -J_x \Delta x. \quad (14)$$

One can solve (14) and obtain a gain matrix $K \in \mathbb{R}^{3 \times 3}$ that offsets the periodic leg states depending on error:

$$u = u^* + K \underbrace{(x - x^*)}_{\Delta x}, \quad (15)$$

where $K = -J_u^{-1} J_x$. A deadbeat control gain library \mathcal{K} collects gains for each trajectory such that $\mathcal{K} = \{K_0, \dots, K_n\}$.

4. Step Adaptation

The spring-mass model is a velocity-based running model, constructed around periodic trajectories, and is stabilized through control of leg angle and length at touchdown. Consequently, if no step adaptation is necessary, the method does not require any environmental information beyond touchdown detection, which is necessary for switching from the flight phase to the stance phase. On the other hand, running through discrete surfaces, such as stepping stones or obstacles, the foot placement regions are constrained and step adaptation is necessary. As step adaptation requires frequent switching between multiple trajectories, an adequate analysis and trajectory switching

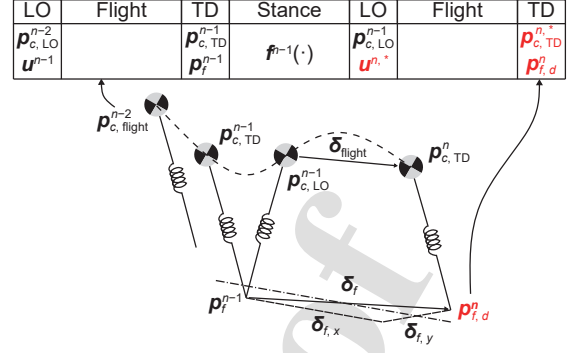


Figure 8: An illustration for multiple-step running with the spring-mass model. The desired stepping configuration is shown in red.

procedure must be developed. This section develops a trajectory selection policy to run over random stepping stones and jump over obstacles. We employ the trajectory and deadbeat control gain libraries and select a trajectory that satisfies convergence, feasibility, and step location conditions.

4.1. Required Minimum Number of Steps to Plan

The exponential stability (11) at point-mass dynamics level is guaranteed only if the desired control action f_d (10) remains within the feasible set \mathcal{F}^3 . When the feasibility projection in (12) is observed, it can be seen that the polyhedral limits, hence the feasibility conditions, are determined by the relative foot placement point with respect to the CoM location. The deadbeat control action in (15) modifies the periodic foot placement point considering the state errors such that the desired control action f_d (10) remains close enough to the feasible set \mathcal{F}^3 allowing convergence during the stance phase.

Recall the hybrid dynamics for the point-mass model,

$$\Sigma : \begin{cases} \ddot{p}_c = g + f/m, & \|r\| < r_0 \\ \ddot{p}_c = g, & \|r\| \geq r_0 \end{cases}, \quad (16)$$

and let $(\cdot)_{(TD,LO)}$ represent the states at the moment of touchdown and liftoff and $p_{f,d}^n$ be the desired foot placement point for the n^{th} step (see Fig. 8). As stability is closely related to the CoM position and velocity with respect to the foot placement point, foot placement on $p_{f,d}^n$ must be done with proper center of mass states to preserve the stability and continue to run. Consequently, we target stepping on $p_{f,d}^n$ with any periodic trajectory (x^*, u^*) . To determine the minimum number of steps required for guaranteed convergence, we perform a controllability analysis of the point-mass dynamics and deadbeat control actions through back-integration (see Fig. 8).

Assume foot placement on $p_{f,d}^n$ at the n^{th} step with any periodic (x^*, u^*) configuration. The center of mass position at the touchdown is an inverse kinematics function of leg states (2),

$$p_{c,TD}^{n,*} = \text{invkin}(p_{f,d}^n, u^{n,*}), \quad (17)$$

where p^* represent configuration states being on the periodic trajectory. The liftoff states of $(n-1)^{\text{th}}$ are related to the touchdown states of n^{th} through the back-integration of ballistic dynamics. Due to the passive dynamics, the states remain on the

periodic manifold:

$$(\mathbf{p}_{c,LO}^{n-1,*}, \dot{\mathbf{p}}_{c,LO}^{n-1,*}) = \text{integration}_{[t=TD^n \rightarrow LO^{n-1}]} (\Sigma_{\text{flight}}, \mathbf{p}_{c,TD}^{n,*}, \dot{\mathbf{p}}_{c,TD}^{n,*}). \quad (18)$$

The touchdown configuration of step $(n-1)$ can be obtained by back-integration through the stance dynamics:

$$(\mathbf{p}_{c,TD}^{n-1}, \dot{\mathbf{p}}_{c,TD}^{n-1}) = \text{integration}_{[t=LO^{n-1} \rightarrow TD^{n-1}]} (\Sigma_{\text{stance}}, \mathbf{p}_{c,LO}^{n-1,*}, \dot{\mathbf{p}}_{c,LO}^{n-1,*}, f(\mathbf{f}_d^{n-1}(\cdot), \mathbf{p}_f^{n-1}, \mu)). \quad (19)$$

The stance phase of step $(n-1)$ in (19) is the first phase with the active control force. Consequently, the touchdown states $(\mathbf{p}_{c,TD}^{n-1}, \dot{\mathbf{p}}_{c,TD}^{n-1})$ are not necessarily periodic and convergence to the desired periodic trajectory must be completed during this phase. However, \mathbf{p}_f^{n-1} is already predetermined by the landing configuration of the previous flight phase. As \mathbf{p}_f^{n-1} is related to the feasibility constraints imposed on the control force (12), convergence without deadbeat control (15) is possible if $\mathbf{f}_d^{n-1}(\cdot)$ remain close enough to the feasible subset \mathcal{F}^3 , but not guaranteed. Consequently, starting to plan from $LO^{(n-2)}$ and initiating step $TD^{(n-1)}$ with the deadbeat controlled leg states,

$$\mathbf{p}_{f,\text{controlled}}^{n-1} = \text{fwdkin}(\text{integration}_{[t=LO^{n-2} \rightarrow TD^{n-1}]} (\Sigma_{\text{flight}}, \mathbf{p}_{c,LO}^{n-2}, \dot{\mathbf{p}}_{c,LO}^{n-2}), \mathbf{u}^* + \mathbf{K}\Delta\mathbf{x}) \quad (20)$$

results in guaranteed convergence to any periodic trajectory that would land on $\mathbf{p}_{f,d}^n$ on the second step.

4.2. Jumping Over Randomly Appearing Obstacles

Let some random obstacles appear in front of the robot immediately after touchdown, so that the robot must jump over them and continue running, maintaining its stability. Let $\delta_{1d,z}$ be the height of the obstacle and the end of the obstacle is $\delta_{1d,x}$ away from the back of the foot (see Fig. 9). As the obstacle appears immediately after touchdown, planning begins from (19), targeting stepping on any \mathbf{p}_f^n that lies behind the object. As \mathbf{p}_f^{n-1} is predetermined by the previous flight phase, convergence to any trajectory is not guaranteed but still possible for some, depending on the overlap between the desired forces $\mathbf{f}_{d,i}(\cdot)$ and the feasibility set \mathcal{F}_i^3 for i^{th} trajectory. Let \mathbf{x}_{TD} and \mathbf{u}_{TD} be the actual spring-mass states of the system at the touchdown moment. Recalling from (15), if the touchdown leg states are close enough to the deadbeat controlled leg states for i^{th} trajectory,

$$\mathbf{u}_{TD} \approx \mathbf{u}_i^* + \mathbf{K}_i(\mathbf{x}_{TD} - \mathbf{x}_i^*), \quad (21)$$

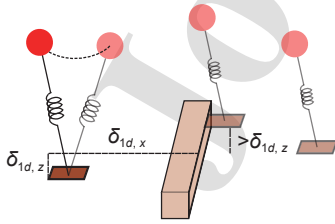


Figure 9: The CoM model jumping over randomly generated obstacles.

Algorithm 1 Jumping over obstacles: the touchdown policy.

1. Eliminate all trajectories whose leg clearance at the apex height is less than $\delta_{1d,z}$. Recalling (2), the elimination condition follows:

$$\delta_{1d,z} < p_{c,z,\text{apex},i} - l_h \cos \theta_{1,i} \cos \theta_{2,i}.$$

2. Calculate foot displacements using (6) for the case of full convergence and store them in a library $\Delta = [\delta_1^T \quad \delta_2^T \quad \dots]^T$.

3. Reduce the displacement library Δ with

$$\delta_{1d,x} < \Delta_x.$$

4. Pick a trajectory with the highest possibility of convergence:

$$\min_i \|\mathbf{u}_{TD} - (\mathbf{u}_i^* + \mathbf{K}_i(\mathbf{x}_{TD} - \mathbf{x}_i^*))\|.$$

convergence is possible in the given stance phase. The trajectory selection policy employs (21) and comprises two complementary parts: touchdown (see Alg. 1) and liftoff (see Alg. 2). Both algorithms output the same trajectory when no disturbance is present. On the other hand, repetitive re-selection is done for re-planning purposes to address disturbances and uncertainties. The touchdown algorithm, Alg. 1, first gathers all trajectories that satisfy the jumping conditions and picks the one with the highest convergence possibility with respect to (21). The liftoff algorithm, Alg. 2, updates the selection in the case of disturbance, uncertainty, or incomplete convergence conditions. The algorithm seeks a trajectory that satisfies the landing conditions and maintains stability after landing. As multiple trajectories may satisfy conditions in Alg. 2, one with the closest apex states is chosen for minimal convergence effort.

4.3. Running Through Randomly Generated Stepping Stones

Let the following two stepping stones are separated by $\delta_{1d} \in \mathbb{R}^3$ and $\delta_{2d} \in \mathbb{R}^3$ (see Fig. 10). Furthermore, let $\delta_r = [l/2 \quad w/2]^T$ be a relaxation such that the foot does not have to be on the center but is still inside the stepping region. Due to strict foothold constraints, planning starts from (20), corresponding to step $(n-2)$, so that landing step $(n-1)$ with the deadbeat controlled leg states results in guaranteed convergence to the desired trajectory to land on $\mathbf{p}_{f,d}^n$. Steps $(n-2)$, $(n-1)$, and (n) represent the current foot location, one-ahead stepping stone, and two-ahead stepping stone, respectively.

The selection policy is triggered at each touchdown and liftoff, resulting in repetitive re-selections to address disturbances and uncertainties. The touchdown and liftoff selection policies are shown in Alg. 3 and Alg. 4 and are extensions of Alg. 1 and Alg. 2, respectively. The touchdown selection policy targets stepping on the next stepping stone. The liftoff selection policy targets stepping anywhere on the one-ahead stepping stone with the deadbeat controlled leg states, such that converging onto the suitable trajectory results in stepping on the two-ahead stepping stone with a periodic trajectory.

Algorithm 2 Jumping over obstacles: the liftoff policy.

1. Estimate the apex state \mathbf{x}_{apex} of the free-flying ballistic dynamics. For a rise time $t_{\text{rise}} = v_z/g$ definition, the apex height is simply:

$$h_{\text{apex}} = p_{\text{com},z} + v_z t_{\text{rise}} - 0.5g t_{\text{rise}}^2.$$

2. Recall (15) and calculate the deadbeat-controlled new set of leg states for guaranteed convergence to any trajectory:

$$\mathbf{u}_{i,\text{new}} = \mathbf{u}_i^* + \mathbf{K}_i(\mathbf{x}_{\text{apex}} - \mathbf{x}_i^*).$$

3. Eliminate all kinematically infeasible trajectories:

$$\begin{aligned} l_{h,\text{min}} &< l_{h,\text{new},i} < l_{h,\text{max}}, \\ \theta_{1,\text{min}} &< \theta_{1,\text{new},i} < \theta_{1,\text{max}}, \\ \theta_{2,\text{min}} &< \theta_{2,\text{new},i} < \theta_{2,\text{max}}. \end{aligned}$$

4. Eliminate all trajectories whose leg clearance at the apex height is less than $\delta_{1d,z}$. Recalling (2), the elimination condition follows:

$$\delta_{1d,z} < p_{c,z,\text{apex},i} - l_{h,\text{new},i} \cos \theta_{1,\text{new},i} \cos \theta_{2,\text{new},i}.$$

5. Calculate the possible foot placement points (6) with the new leg states $\mathbf{u}_{i,\text{new}}$ and store them: $\Delta = [\delta_1^T \quad \delta_2^T \quad \dots]^T$.

6. Reduce the displacement library Δ with

$$\delta_{1d,x} < \Delta_x.$$

7. Pick a trajectory with minimal convergence effort:

$$\min_i \|\mathbf{x}_{\text{apex}} - \mathbf{x}_i^*\|.$$

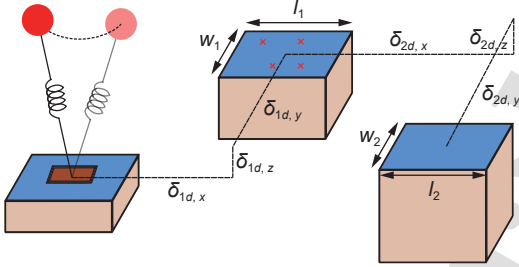


Figure 10: The CoM model navigating over stepping stones. The red crosses represent possible random foot placement points due to the relaxation conditions, which require an update on δ_{2d} for the next step.

Algorithm 3 Random stepping stones: the touchdown policy.

1. Apply Alg. 1 for the next step with the following two modifications:

- In step 2, take account of surface-level differences and calculate the flight time ($t_{\text{flight}} = t_{\text{rise}} + t_{\text{fall}}$) accordingly.
- In step 3, the reduction condition changes:

$$\begin{aligned} \delta_{1d,x} - w_1/2 &< \Delta_x < \delta_{1d,x} + w_1/2, \\ \delta_{1d,y} - l_1/2 &< \Delta_y < \delta_{1d,y} + l_1/2. \end{aligned}$$

4.4. Changing the Running Direction

Changing the heading angle combines two tasks: rotating the inertia/body and changing the CoM velocity. This section

Algorithm 4 Random stepping stones: the liftoff policy.

1. Apply Algorithm 2 to find all trajectories that end up stepping on the one-ahead stepping stone with the following modifications:

- In step 5, take account of surface-level differences and calculate the flight time ($t_{\text{flight}} = t_{\text{rise}} + t_{\text{fall}}$) accordingly.
- In step 6, the reduction condition changes:

$$\begin{aligned} \delta_{1d,x} - w_1/2 &< \Delta_x < \delta_{1d,x} + w_1/2, \\ \delta_{1d,y} - l_1/2 &< \Delta_y < \delta_{1d,y} + l_1/2. \end{aligned}$$

- Skip step 7.

2. Eliminate all trajectories whose leg clearance at the apex height is less than $\delta_{2d,z}$.

3. As a result of the relaxation conditions, each trajectory has a different foot placement point on the first stepping stone (see Fig. 10). Update $\delta_{2d,x}$ and $\delta_{2d,y}$ for each remaining trajectory:

$$\begin{aligned} \delta_{2d,x,i} &= \delta_{2d,x} - (\delta_{i,x} - \delta_{2d,x}), \\ \delta_{2d,y,i} &= \delta_{2d,y} + (\delta_{i,y} - \delta_{2d,y}). \end{aligned}$$

4. Calculate foot displacements using (6) for the case of full convergence, also considering the height differences and flight times accordingly, and reconstruct the displacement library Δ .

5. Further reduce the displacement library Δ for the following two relaxed conditions:

$$\begin{aligned} \delta_{2d,x,i} - w_2/2 &< \Delta_{x,i} < \delta_{2d,x,i} + w_2/2, \\ \delta_{2d,y,i} - l_2/2 &< \Delta_{y,i} < \delta_{2d,y,i} + l_2/2. \end{aligned}$$

6. Pick a trajectory with minimal convergence effort:

$$\min_i \|\mathbf{x}_{\text{apex}} - \mathbf{x}_i^*\|.$$

assumes body rotations are handled at the whole-body control level and focuses on CoM control. During running, rotating the body while failing to change the CoM velocity direction causes the robot to roll and fall to the sides (Fig. 1C shows the body lean and the tendency to roll side when a sudden turn is applied while running). Therefore, step adaptation is required for the next step through the deadbeat controller.

Define $\mathbf{R}_h \in \mathbb{R}^{3 \times 3}$ to be a rotation matrix from the world frame ${}^0\mathcal{F}$ to the new frame ${}^1\mathcal{F}$ aligned with the new desired

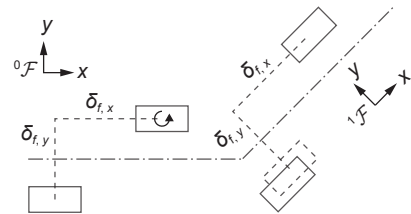


Figure 11: Foot placement illustration when the robot changes the running direction suddenly at the second step. During the stance phase, the system aims to converge to the desired trajectory. During the flight phase, the deadbeat controller reacts to the remaining errors by shifting the foot placement point towards the bottom right of the nominal.

running direction (see Fig. 11). The reference CoM trajectories, $\mathbf{p}_{c,d}$, $\dot{\mathbf{p}}_{c,d}$, and $\ddot{\mathbf{p}}_{c,d}$ are then transformed into the new frame, i.e., ${}^1\mathbf{p}_{c,d} = \mathbf{R}_h^0 \mathbf{p}_{c,d}$. Similarly, the error definition for the deadbeat controller (15) is transformed back to the world frame such that $\Delta \mathbf{x} = \mathbf{R}_h^{\top 1} \mathbf{x} - \mathbf{x}^*$.

4.5. Scalability of the Policy

The trajectory selection algorithms involve combinations of scalar and 3D vector-matrix algebra operations for each trajectory. Hence, the computation time is linearly proportional to the number of trajectories in the library. We measure $20\mu\text{s}$ to compute the most expensive algorithm, Alg. 4, for the example library used in this study, which contains 315 trajectories (see Fig. 5). Consequently, the number of trajectories can reach more than ten thousand satisfying the real-time constraint.

5. Implementation on Humanoid Robots

Humanoid robots are high-degree-of-freedom complex floating base systems. They include multiple subsystems, such as legs, arms, head, and other joints on the torso and pelvis. Control of humanoid robots includes a set of task definitions covering the whole-body dynamics, such as swinging the legs and arms, keeping the torso upright, and following a CoM trajectory. The task formulations can be based on different controllers, such as inverse dynamics control as in this paper, or passivity-based control [36, 37, 38]. Then, the tasks are combined in a QP problem and solved all together to find a solution that satisfies all the tasks the best [39, 40, 41]. Task prioritization can be handled either strictly by projecting lower-priority tasks into the null space of high-priority tasks [39, 40, 42] or softly with simple weight adjustments, as in this paper. Detailed WBC formulations with strict and soft prioritization are provided in [43] and [44], respectively.

This section combines the conventional inverse dynamics whole-body control formulation with two add-ons for collision avoidance and reactive limb swing control. The new task definitions introduce adaptiveness to the WBC formulation, enabling it to handle multiple CoM trajectories without requiring additional tuning of gains and limb-swing trajectories. The reactive limb swing control is relevant and important in the case of frequent trajectory switching at the CoM level, resulting in a wide variation of stance and flight phase characteristics. The collision detection and avoidance prevent reactive limb swinging from resulting in self-collision and can be implemented to prevent possible arm-to-body and leg-to-leg collisions.

5.1. Floating Base System Dynamics

Let \mathbf{q} be a set of configuration variables and $\mathbf{v} = (\mathbf{v}_b, \mathbf{v}_j)$ be the generalized velocity where $\mathbf{v}_b = (\mathbf{v}_b, \boldsymbol{\omega}_b) \in \mathbb{R}^6$ is the linear and angular velocity of the floating base and $\mathbf{v}_j \in \mathbb{R}^n$ is the generalized velocity of the joints. The well-known robotic system dynamics results in

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} + \boldsymbol{\tau}_g(\mathbf{q}) = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{bmatrix} + \mathbf{J}_c(\mathbf{q})^\top \mathbf{f}_c + \mathbf{J}_p(\mathbf{q})^\top \mathbf{f}_p, \quad (22)$$

where \mathbf{M} is inertia matrix; \mathbf{C} is Coriolis matrix; $\boldsymbol{\tau}_g$ is gravity vector; $\boldsymbol{\tau}$ is joint torques; \mathbf{J}_c is contact Jacobians; \mathbf{f}_c is contact forces; \mathbf{J}_p is closed chain constraint Jacobians; \mathbf{f}_p is closed chain constraint forces. In the case of closed kinematic linkages, the passive joints are constrained to move together with their corresponding actively driven counterparts through holonomic constraints [45, 46].

5.2. Task Formulation via Inverse Dynamics Control (IDC)

For a given task Jacobian \mathbf{J}_i , the task space velocity yields

$$\dot{\mathbf{x}}_i = \mathbf{J}_i \mathbf{v}. \quad (23)$$

The time derivative of (23) yields the task acceleration,

$$\ddot{\mathbf{x}}_i = \mathbf{J}_i \dot{\mathbf{v}} + \dot{\mathbf{J}}_i \mathbf{v}. \quad (24)$$

Setting $\ddot{\mathbf{x}}_i = \ddot{\mathbf{x}}_{i,d}$, where

$$\ddot{\mathbf{x}}_{i,d} = \ddot{\mathbf{x}}_{i,ref} + \underbrace{\mathbf{K}_{D,i}(\dot{\mathbf{x}}_{i,ref} - \dot{\mathbf{x}}_i)}_e + \underbrace{\mathbf{K}_{P,i}(\mathbf{x}_{i,ref} - \mathbf{x}_i)}_{e_i} \quad (25)$$

and $\{\mathbf{K}_{D,i}, \mathbf{K}_{P,i}\}$ are positive definite gain matrices, the closed-loop error dynamics results in being exponentially stable,

$$\ddot{e}_i + \mathbf{K}_{D,i}\dot{e}_i + \mathbf{K}_{P,i}e_i = \mathbf{0}. \quad (26)$$

5.3. Collision Detection and Avoidance

Let $E : \mathbb{R}^3 \rightarrow \mathbb{R}$ be a general ellipsoid function,

$$E(\mathbf{p}) = \frac{p_x}{a^2} + \frac{p_y}{b^2} + \frac{p_z}{c^2} - 1, \quad (27)$$

where $\{a, b, c\} \in \mathbb{R}$ are radius of each axes. Consequently,

$$\mathcal{E} = \{\mathbf{p} \in \mathbb{R}^3 \mid E(\mathbf{p}) = 0\}$$

collects all the points on the ellipsoidal surface. Assume a task T is placed at point \mathbf{p}_T and a vector $\mathbf{v}_{B,T} = \mathbf{p}_T - \mathbf{p}_B$ defines a vector from the ellipsoidal's center to the task. Let \mathbf{n} be the surface normal. In the case $E(\mathbf{v}_{B,T}) \leq 0$ and $\dot{\mathbf{x}}_T^\top \mathbf{n} < 0$, implying, the task is inside the ellipsoid and its velocity towards

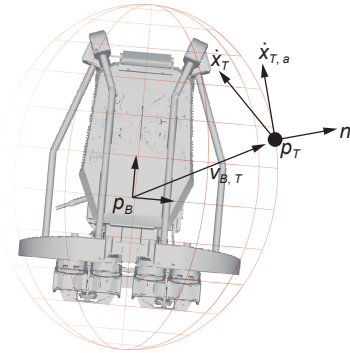


Figure 12: An illustration of a task control reaching the ellipsoidal boundary. The task velocity $\dot{\mathbf{x}}_T$ is projected on the tangent space to prevent collision.

inside, respectively, the task velocity must be projected to prevent collision (see Fig. 12). Projection onto the nullspace of the ellipsoid's surface normal results in movement in the tangential direction. The allowed velocity $\dot{\mathbf{x}}_{T,a}$ is simply

$$\dot{\mathbf{x}}_{T,a} = \dot{\mathbf{x}}_T - \text{proj}_n(\dot{\mathbf{x}}_T), \quad (28)$$

where

$$\mathbf{n} = \left. \frac{dE}{d\mathbf{q}} \right|_{p=p_T}.$$

The resultant task velocity is introduced as a high-priority task in WBC formulation. In case $E(\mathbf{v}_{B,T}) > 0$ or $E(\mathbf{v}_{B,T}) \leq 0$ but $\dot{\mathbf{x}}_T^\top \mathbf{n} > 0$ no action is necessary as the task is either outside of the region, or inside of the region but moving towards outside.

5.4. Reactive Limb Swing Control

Another important aspect of humanoid robot locomotion is torso and limb swing control during running. As the 3D spring-mass dynamics covers only the CoM control and the foot placement point, the remaining limb swing and posture control must be handled at the WBC level. Independent of the CoM stability, preserving the postural stability, e.g., keeping the body upright, is crucial for the continuity of the motion. As this study combines varying running behaviors, in terms of trajectories, stance time, and flight time, the limb-swing and torso control must be generic and suitable for all.

5.4.1. Orientational Dynamics and Dynamic Coupling

The centroidal momentum is related to the generalized velocities through

$$\mathbf{h}_G = \mathbf{A}_G(\mathbf{q})\mathbf{v}, \quad (29)$$

where $\mathbf{A}_G \in \mathbb{R}^{6 \times (n_j+6)}$ is the centroidal momentum matrix (CMM) [33, 34] and n_j is the number of joints. The CMM is composed of linear and angular parts,

$$\mathbf{A}_G = \begin{bmatrix} \mathbf{A}_l \in \mathbb{R}^{3 \times (n_j+6)} \\ \mathbf{A}_k \in \mathbb{R}^{3 \times (n_j+6)} \end{bmatrix}. \quad (30)$$

Furthermore, the centroidal angular momentum matrix (CAMM) \mathbf{A}_k can also be decomposed into

$$\mathbf{A}_k = \begin{bmatrix} \mathbf{A}_v \in \mathbb{R}^{3 \times 3} & \mathbf{A}_\omega \in \mathbb{R}^{3 \times 3} & \mathbf{A}_j \in \mathbb{R}^{3 \times n_j} \end{bmatrix}, \quad (31)$$

where \mathbf{A}_v , \mathbf{A}_ω , and \mathbf{A}_j represent body translational, body rotational, and joint velocity portions, respectively. Among the submatrices of the CAMM, \mathbf{A}_ω is invertible and \mathbf{A}_v is always a zero matrix for all configurations $\forall \mathbf{q}$ [47]. The centroidal angular momentum is then equal to

$$\mathbf{k}_G = \begin{bmatrix} \mathbf{A}_\omega & \mathbf{A}_j \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}_b \\ \mathbf{v}_j \end{bmatrix}. \quad (32)$$

Reorganizing (32) results in the base frame orientation dynamics showing the coupling between the joint velocities and base frame rotational velocity,

$$\boldsymbol{\omega}_b = \mathbf{A}_\omega^{-1}(\mathbf{k}_G - \mathbf{A}_j\mathbf{v}_j). \quad (33)$$

Furthermore, the rate of centroidal angular momentum,

$$\dot{\mathbf{k}}_G = \sum_{i=1}^{n_c} ((\mathbf{p}_{\text{contact},i} - \mathbf{p}_{\text{CoM}}) \times \mathbf{f}_{\text{contact},i}), \quad (34)$$

is related to the interaction forces at the contact points [32, 34]. During the flight phase, where the contact forces are zero, the centroidal angular momentum is conserved, $\dot{\mathbf{k}}_G = \mathbf{0}$. Consequently, the base orientation cannot be controlled explicitly, but is a function of the constant angular momentum and joint velocities (33). During the stance phase, however, the problem of orientation control is easier as it can be controlled explicitly by the ground reaction forces (34).

5.4.2. Control Action

The stance phase includes multiple tasks: tracking the given CoM trajectory originating from the spring-mass model, keeping the stance foot fixed on the ground, keeping the torso upright, swinging the other foot forward to prepare for the next step, and swinging the arms. Among the stance phase tasks, arm swing is a redundant task that can be utilized in a reactive manner to support postural stability. The flight phase, on the other hand, is more redundant and harder to control than the stance phase, as no foot is fixed on the ground. The only control target originating from the spring-mass model is placing the upcoming stance foot at a certain location with respect to the CoM and waiting for touchdown. The remaining joints, which are the swing leg and arms, must be controlled for the postural stability to prevent any excessive roll due to the conservation of angular momentum (34) and coupled body velocity dynamics (33).

Let n_e and n_r be the number of essential and redundant joints such that $n_j = n_e + n_r$. Furthermore, let

$$\begin{bmatrix} \mathbf{J}_v \\ \mathbf{J}_\omega \\ \mathbf{J}_e \\ \mathbf{J}_r \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times n_e} & \mathbf{0}_{3 \times n_r} \\ \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times n_e} & \mathbf{0}_{3 \times n_r} \\ \mathbf{0}_{n_e \times 3} & \mathbf{0}_{n_e \times 3} & \mathbf{I}_{n_e \times n_e} & \mathbf{0}_{n_e \times n_r} \\ \mathbf{0}_{n_r \times 3} & \mathbf{0}_{n_r \times 3} & \mathbf{0}_{n_r \times n_e} & \mathbf{I}_{n_r \times n_r} \end{bmatrix} \quad (35)$$

collect all joint-level Jacobians for body translational, body rotational, essential joints, and redundant joints, respectively. The design objective is controlling the base frame orientation task related to \mathbf{J}_ω , in the nullspace of $\mathbf{J}_t = [\mathbf{J}_v; \mathbf{J}_e]$, such that only redundant joints are utilized. Let

$$\mathbf{N}_t = \mathbf{I} - \bar{\mathbf{J}}_t \mathbf{J}_t \quad (36)$$

be the nullspace projection of \mathbf{J}_t where $\bar{\mathbf{J}}_t$ is dynamically consistent pseudo-inverse of \mathbf{J}_t [41]. The new task Jacobian for base orientation control becomes

$$\mathbf{J}_{\omega,p} = \mathbf{J}_\omega \mathbf{N}_t. \quad (37)$$

Defining a WBC task (24) through $\mathbf{J}_{\omega,p}$ and minimizing for the desired acceleration (25) written in terms of actual base orientational position and velocity errors results in a control through the ground reaction forces (if available) and redundant joints.

Note that the proposed task formulation for base orientation control is not a task prioritization but rather a joint isolation, as

they are written at the joint level. The formulation allows only a certain set of joints to be utilized for body orientation control, regardless of other tasks. In case multiple task formulations share common joints, they are still affected by each other and prioritized by weight matrices. A task written with $\mathbf{J}_{\omega,p}$ for the base orientation control results in inherently determined redundant joint control for the stance and flight phases.

5.5. Whole Body Control Formulation

The desired CoM and limb trajectories are mapped to a humanoid robot through an optimization-based whole-body controller. Selecting the parameters to optimize for as $\{\dot{\mathbf{v}}; \boldsymbol{\tau}; \mathbf{f}_c; \mathbf{f}_p\}$, the combination of all task controllers (25) into a quadratic problem constitutes the cost function of the problem:

$$\min_{\dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{f}_c, \mathbf{f}_p} \sum_i (\ddot{\mathbf{x}}_{i,d} - \ddot{\mathbf{x}}_i)^\top \mathbf{W}_i (\ddot{\mathbf{x}}_{i,d} - \ddot{\mathbf{x}}_i) \quad (38)$$

Such that:

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} + \boldsymbol{\tau}_g = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{bmatrix} + \mathbf{J}_c^\top \mathbf{f}_c + \mathbf{J}_p^\top \mathbf{f}_p, \quad (38a)$$

$$\mathbf{J}_c \dot{\mathbf{v}} + \dot{\mathbf{J}}_c \dot{\mathbf{q}} = \mathbf{0}, \quad (38b)$$

$$|f_{x,l}| \leq \frac{\mu f_{z,l}}{\sqrt{2}}, \quad |f_{y,l}| \leq \frac{\mu f_{z,l}}{\sqrt{2}}, \quad \text{and } f_z \geq 0 \quad \forall l, \quad (38c)$$

$$\mathbf{J}_p \dot{\mathbf{v}} + \dot{\mathbf{J}}_p \dot{\mathbf{q}} = \mathbf{0}, \quad (38d)$$

$$\boldsymbol{\tau}_{min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{max}. \quad (38e)$$

where (38b) and (38c) are stationary foot and friction constraints. As the cost function (38) is constituted by a combination of different tasks and their corresponding weight matrices \mathbf{W}_i , the tasks with higher weights are prioritized.

6. Simulations

The simulation results include flat surface running with velocity changes, sudden turns, and running through stepping stones. To further justify the proposed method's robustness against real-world challenges, we also simulate its operation under a comprehensive and significant set of signal noise, imprecision, and delays. The supplemental video includes all and more, including responses to force disturbances, unobserved height differences, edge stepping, and slalom running. Note that, although this paper presents diverse and highly dynamic behaviors (see Figs. 1 and 14), all behaviors were obtained using the same parameter set without requiring additional tuning, thanks to the reactive limb-swing formulation.

We use MuJoCo Physics Simulator [48] for simulation; Pinocchio [49] for rigid body dynamics and kinematics calculations; ProxQP to solve the QP problems in (38) and (12); Ceres Solver [50] to solve the nonlinear least-squares problems for periodic trajectory optimization (5). We set the friction coefficient $\mu = 0.6$, considering concrete terrains. Execution times for control and library calculations are provided in Table 1.

In addition to reactive limb swing control, we introduce a low-weight zero position and velocity task for redundant joints

to enhance numerical stability. The limb-swing task overrides the low-weight zero configuration task, causing the redundant limbs to swing inherently to maintain the torso's upright position, as shown in Fig. 14. Kangaroo uses backward leg swing for angular momentum compensation (no arms), while Unitree G1 employs both leg and arm swing — validating the framework's adaptability to kinematic variability. The backward leg swing is more significant for Kangaroo as it does not have any arms to compensate for angular momentum. Nevertheless, even without arms, the control system manages to handle long flight phases as shown in Fig. 14. The reactive control also significantly enhances disturbance rejection capabilities at the whole-body control level.

6.1. Response to Sudden Velocity Command Changes

One of the key factors in navigating a random environment is the system's ability to rapidly switch trajectories. We plot the system's response to suddenly changing velocity commands in Fig. 15A. The figure illustrates that when commanded at the liftoff moment, the robot transitions from stationary jogging to a $1m/s$ running velocity in a single step. The robot then accelerates to $2m/s$, decelerates back to $1m/s$, and comes to a sudden stop. During the flight phase, in the event of an error or a trajectory change command, the deadbeat controller adjusts the leg angle and length parameters to ensure that the required ground reaction forces necessary to converge on a new trajectory or reject disturbances are feasible. The figure also compares the system's response to a sudden deceleration of $2 \rightarrow 1m/s$ without deadbeat control. It shows that without any leg adjustment, the stance phase active control system cannot converge to the desired trajectory in a single step due to feasibility limitations. This is most important for running with precise foot placement point control and shows the significance of mapping SLIP input parameters with the deadbeat control gains in the step adaptation policy. Without deadbeat control, the response against disturbances and sudden velocity/direction commands usually ended up with failures in our simulations.

6.2. Response to Sudden Direction Command Changes

The deadbeat control implicitly inherits sudden and continuous direction change capabilities at the CoM trajectory generation level. The remaining inertial and posture control is achieved at the WBC level, also with the help of reactive limb-swing capabilities. The supplemental video shows that the

Table 1: Execution times where T and $\#$ represent the sampling time and number of repetitions. CPU: AMD Ryzen 7 5800X, Interface: C++.

Offline	T	$\#$	time
SLIP Library (5)	10^{-4}	315	1.2s
\mathbf{J}_x of active SLIP (13)	10^{-3}	315	1.6s
\mathbf{J}_u of active SLIP (13)	10^{-3}	315	1.6s
Online			time
Trajectory Selection with 315 Trajectories		20	μs
WBC Formulation		35	μs
WBC Solution Avg.		125	μs

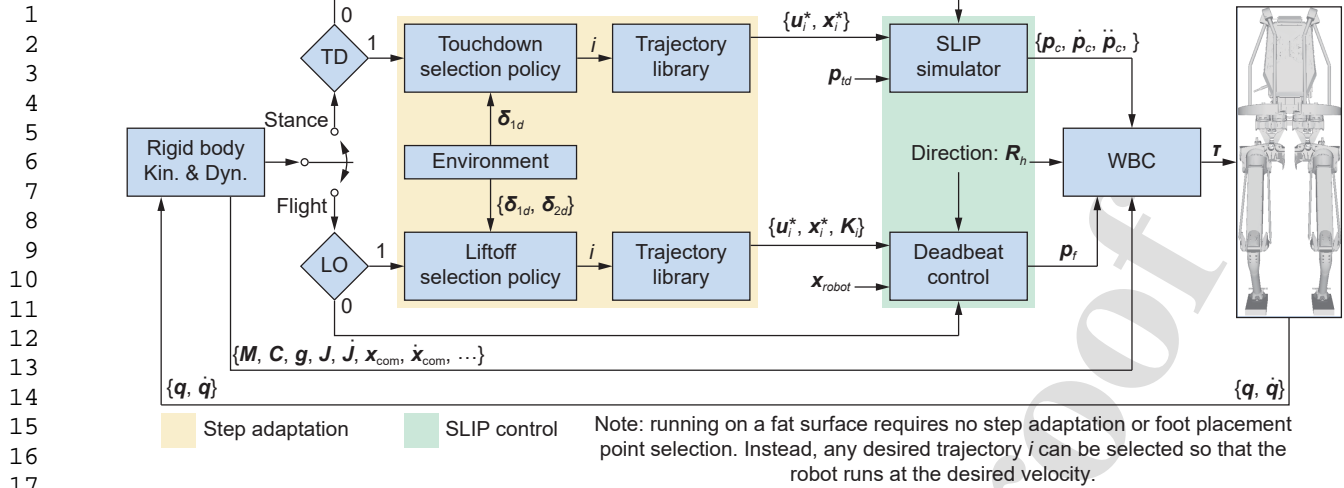
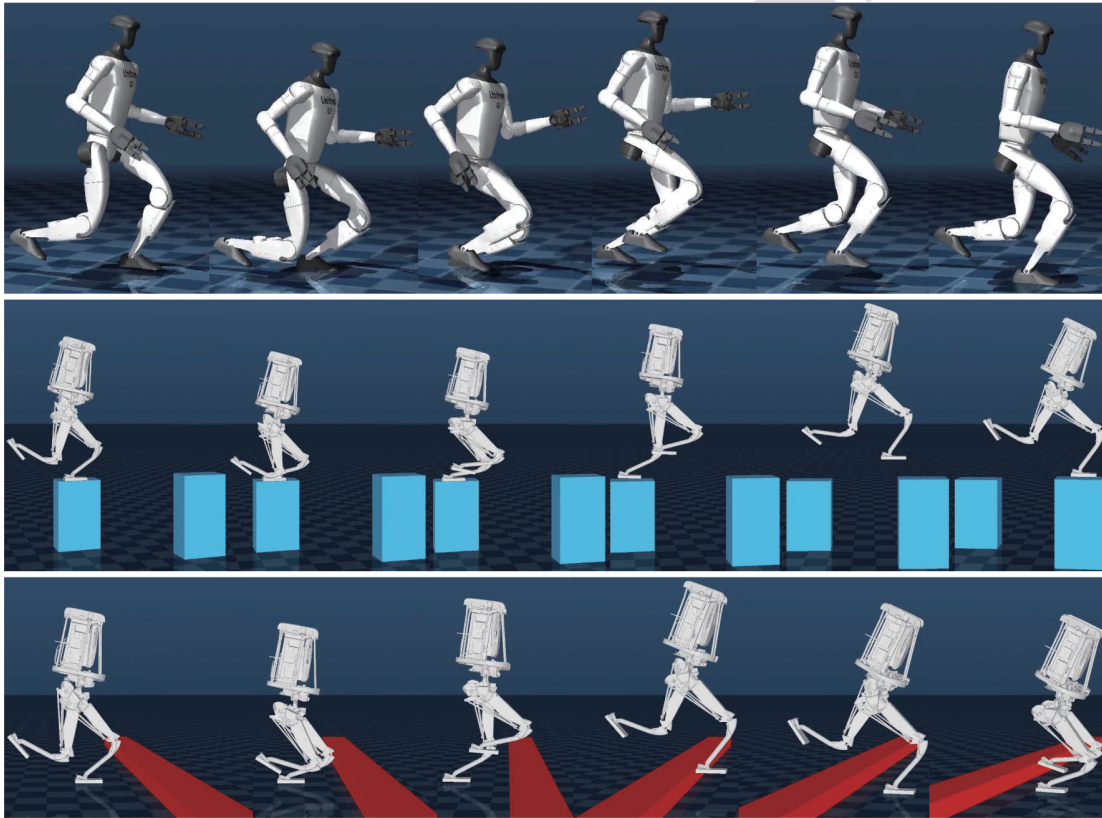


Figure 13: Overall control system diagram for the proposed method. Note that the activated parts differ for the stance and flight phases.



49 Figure 14: Top: Unitree G1 running at $1m/s$. Middle: Kangaroo running through randomly generated stepping stones which are 60 to 100, 35 to 45, and ± 10
50 centimeters apart from each other in the longitudinal, lateral, and vertical directions, respectively. Bottom: Jumping over randomly generated obstacles that appear
51 40 centimeters ahead. The objects' width and height vary between 5 to 30 and 10 to 15 centimeters, respectively. The remaining behaviors are shown in Fig. 1.

52
53
54 robot can rotate 10° per step and follow a sinusoidal (slalom)
55 trajectory with $2m/s$ running velocity. We show the system's re-
56 sponse to sudden $\pi/4$ radian direction change commands while
57 running at $1m/s$ forward velocity, both with the inner and outer
58 legs. At the touchdown, the spring-mass trajectories and the
59 body rotation commands of the torso and foot are transformed

first by $\pi/4$ radian for turning with the outer leg and then by
 $-\pi/4$ radian back for turning with the inner leg (see Fig. 15B).
As the outer leg is already on the right side during the left turn,
the feasible ground reaction forces match the required ones,
and the system approaches the desired velocity states in a single
stance. The second turn, on the other hand, is a right turn

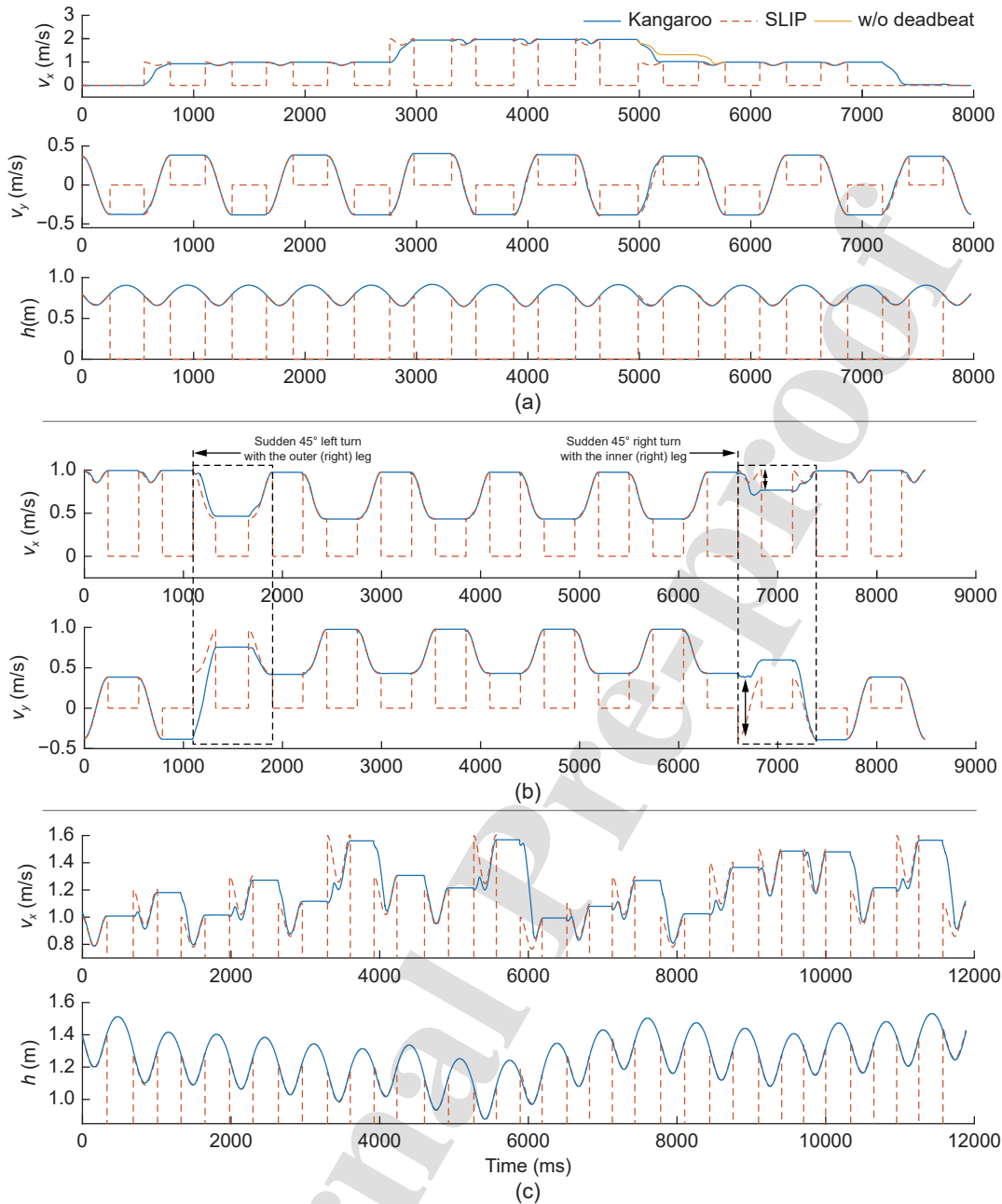


Figure 15: Response of Kangaroo to (A) sudden velocity command changes, (B) sudden direction changes by $\pi/4$ both with the inner and outer leg, (C) running through randomly generated stepping stones. Zero SLIP values indicate the flight phases.

with the right leg. The robot's current posture and foot placement point do not allow it to converge to the desired states as much, and significant magnitude errors remain, especially in lateral velocity. The robot rolls over to the side and falls without any step adjustment in the next stance. The deadbeat controller places the foot wider than its periodic placement point such that the remaining lateral velocity is rejected. As a high-magnitude sudden turn also requires significant whole-body coordination to rotate the body, our simulations without reactive-limb swing control resulted in postural instability.

6.3. Response to Randomly Generated Stepping Stones

The combination of highly maneuverable velocity control, reactive limb swing control, and the step adaptation policy with convergence considerations enables the robot to run over randomly generated stepping stones, knowing only two steps ahead. The range of the randomness are $\{0.6, 1.0\}m$ in longitudinal, $\{0.35, 0.45\}m$ in lateral, and $\{-0.1, 0.1\}m$ in vertical directions. The absolute distance between the two consecutive stepping stones is usually longer than the leg length, and due to randomness, each simulation run generates a different environment. The response to randomly generated stepping stones is

shown in Fig. 15C. The figure illustrates the capability of making frequent trajectory changes at each step.

6.4. Ground Reaction Forces

Observing the system's ground reaction force evolution is another way to examine the match between the whole body and spring-mass dynamics. As shown in Fig. 16, the ground reaction profiles of the humanoid running resemble those of a spring-mass dynamics system. As the desired running velocity changes, the active controller breaks the periodicity to converge into the desired trajectory. Once converged, the ground reaction forces follow almost the exact pattern of the conventional 3D spring-mass model.

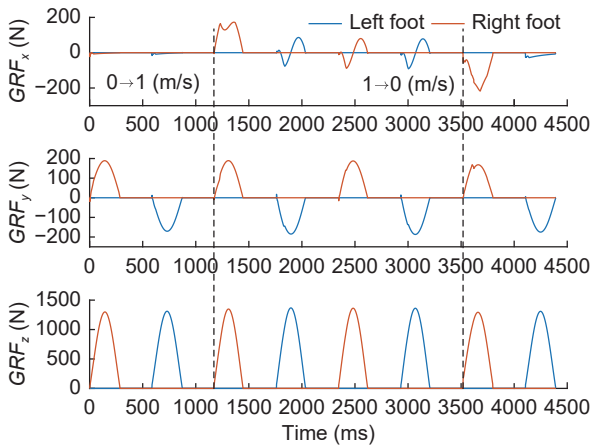


Figure 16: Ground reaction forces of Kangaroo while running at different paces.

6.5. Response to Noise and Uncertainty

To better justify the proposed method's robustness against real-world challenges, such as disturbances, signal noises, imprecision, and delays, we perform an additional verification. The supplemental video already includes the system's response to force disturbances, undetected height changes, and corner-stepping conditions. Here, we take into account unideal actuation characteristics, delays, modeling errors, and sensor noise. The new system diagram, incorporating noise and uncertainty, is shown in Fig. 17. The applied noise and uncertainties are random and generated in run-time through "`std::random_device`" and "`std::mt19937`" in C++.

We again examine running at different velocities and during transitions under the influence of the newly introduced real-world conditions. Note that the overall control system, in terms of the whole-body control, trajectories, and control gains, has been kept the same, and no additional tuning has been done. The system's response is provided in Fig. 18. Compared to the ideal behaviors shown in Fig. 15, Fig. 18 exhibits a noisier behavior, which is more pronounced in the running direction, v_x . The actual periodic running velocity varies between the consecutive steps, and transitioning performance decays under the given conditions. Nevertheless, longitudinal velocity RMSE = 0.085m/s, lateral velocity RMSE = 0.021m/s, confirming stability under disturbances. Furthermore, state-based

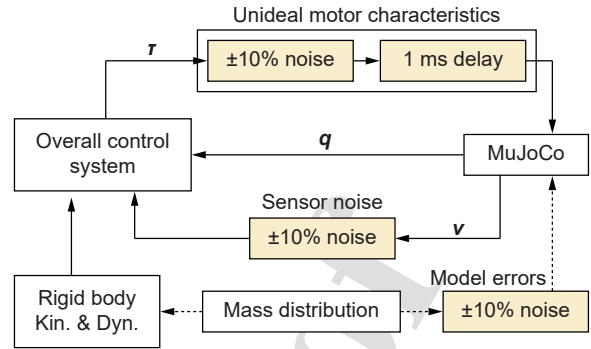


Figure 17: A diagram regarding the applied noise. The generalized velocity readings (body frame and joint velocities), motor torque commands, and mass distribution are subject to signal noise, delay, and uncertainty.

phase transitioning of the spring-mass model enhances the disturbance rejection performance as the flight and stance times are not precise under disturbances and uncertainties.

7. Discussion

Two main outcomes of this paper are step adaptation through the spring-mass model and reactive limb swing control.

7.1. Step Adaptation

The original spring-mass model is a velocity-based biomimetic running model constructed around periodic trajectories [1, 12, 13]. Although it exhibits highly dynamic behaviors over a compelling set of continuous ground reaction force patterns, step adaptation has not yet been achieved on a humanoid model. We demonstrate that, while retaining the advantages of the periodic spring-mass running model, our method can induce sudden changes in velocity (see Figures 15A-16) and reject disturbances including physical perturbations, model and sensing imprecisions, and varying surface conditions (see Figures 1, 17, and 18). The rapid trajectory switching capability is employed by the step adaptation policy (see Algorithms 1-4), enabling step adaptation in random environments (see Figures 14-15C). Although requiring a trajectory library, the library generation is entirely automatic and tuning-free, taking only a few seconds to calculate for hundreds of trajectories. It is scalable up to thousands of trajectories while still satisfying real-time constraints (see Table 1).

7.2. Reactive Limb Swing Control

The spring-mass model falls short in capturing multibody and angular momentum effects when controlling a full-scale humanoid robot [12, 13, 47]. As it only covers the CoM control and the foot placement point, the remaining limb swing and posture control are not considered in planning and must be handled separately. The limb swing is important both for angular momentum control and preparation for the next step. In cases of frequent trajectory switching and disturbances, where force profiles, stance time, flight time, and foot placement conditions vary, a different limb-swing behavior is required to maintain

1 postural stability. We address this problem through a reactive
 2 limb-swing task, eliminating the need for trajectory-specific
 3 tuning. We demonstrate that the reactive limb swing task uti-
 4 lizes a backward leg swing for angular momentum compensa-
 5 tion in the case of the Kangaroo (without arms) and employs
 6 both leg and arm swing in the case of the Unitree G1, thereby
 7 validating the framework’s adaptability to kinematic variability.
 8 Along with tuning-free CoM trajectory switching capabilities,
 9 the limb swing task also enhances disturbance rejection perfor-
 10 mance at the whole-body control level, as it is related to overall
 11 postural stabilization while tracking the given CoM trajectory.

14 8. Conclusion

15 This study develops an overall control framework for highly
 16 maneuverable robots to navigate through random environments.
 17 First, a carefree 3D spring-mass trajectory library is generated
 18 by varying the running velocity, apex height, and leg stiffness.
 19 Then, an apex-to-apex deadbeat control gain library, which ma-
 20 nipulates leg angles and length parameters for disturbance re-
 21 jection or trajectory switching purposes, is obtained through a
 22 model that resembles the full-order model’s constrained and ac-
 23 tively controlled CoM dynamics well. The total offline library
 24 calculation process takes around 4.5 seconds for 315 trajecto-
 25 ries and can even be done a few seconds before running the
 26 robot. Then, we develop a step adaptation policy that enables
 27 the robot to run in random environments. We show that running
 28 through random stepping stones and jumping over obstacles is
 29 possible with at least two-step and one-step planning, respec-
 30 tively. Furthermore, the step adaptation is performed consid-
 31 ering the kinematic and dynamic feasibility, as well as conver-
 32 gence conditions. The online step-adaptation policy can handle
 33 thousands of trajectories while still satisfying the real-time con-
 34 ditions. Mapping the spring-mass trajectories to a full-size hu-
 35 manoid robot model is achieved through a conventional whole-
 36 body control with additional reactive-limb swing control for-
 37 mulation. The reactive limb swing enables the preservation of
 38 postural whole-body stability under a wide variety of different
 39 spring-mass trajectories and enhances the disturbance rejection
 40 performance. We show the inclusiveness and the robustness of
 41 the proposed framework through various challenging and agile
 42 behaviors, such as running through randomly generated step-
 43 ping stones, jumping over random obstacles, performing highly
 44 dynamic slalom motions, changing the running direction sud-
 45 denly with a random leg, and rejecting significant disturbances
 46 and uncertainties through the MuJoCo physics simulator. To
 47 further justify robustness against real-world challenges, we pro-
 48 vide additional simulations with unideal actuation characteris-
 49 tics, delays, modeling errors, and sensor noise. All the afore-
 50 mentioned behaviors are performed using a single library and
 51 the same set of whole-body control parameters, without requir-
 52 ing additional tuning.

57 9. Supplemental Media

58 The presented and additional results are provided in the sup-
 59 plemental video with the following content:

- The actively controlled template model runs through ran-
 domly generated stepping stones.
- Starting from stationary jogging, Unitree G1 accelerates to
 $1m/s$ and comes back to stop.
- Starting from stationary jogging, Kangaroo first acceler-
 ates to $1m/s$, then to $2m/s$, and comes back to stop.
- Kangaroo runs at $2m/s$ velocity and performs slalom
 movement with 10 degree rotation per step. Then it per-
 forms sudden $\pm\pi/4$ radian turns with random legs.
- Kangaroo runs at $1m/s$ velocity and jumps over randomly
 generated objects.
- Kangaroo runs through randomly generated stepping
 stones.
- Force disturbance rejection performance in all directions
 while running at $1m/s$ velocity.
- Running through unobserved ground height differences
 and edge stepping.

Acknowledgements

This project has received funding from the European Re-
 search Council (ERC) under the European Union’s Horizon
 2020 research and innovation programme (grant agreement No.
 819358).

References

- [1] R. Blickhan, The spring-mass model for running and hop-
 ping, *Journal of biomechanics* 22 (11-12) (1989) 1217–
 1227.
- [2] D. E. Koditschek, R. J. Full, M. Buehler, Mechanical as-
 pects of legged locomotion control, *Arthropod structure
 & development* 33 (3) (2004) 251–272.
- [3] P. Holmes, R. J. Full, D. Koditschek, J. Guckenheimer,
 The dynamics of legged locomotion: Models, analyses,
 and challenges, *SIAM review* 48 (2) (2006) 207–304.
- [4] H. Geyer, A. Seyfarth, R. Blickhan, Compliant leg be-
 haviour explains basic dynamics of walking and running,
Proceedings of the Royal Society B: Biological Sciences
 273 (1603) (2006) 2861–2867.
- [5] J. Hurst, Walk this way: To be useful around people,
 robots need to learn how to move like we do, *IEEE Spec-
 trum* 56 (3) (2019) 30–51.
- [6] U. Saranli, W. Schwind, D. Koditschek, Toward the con-
 trol of a multi-jointed, monoped runner, in: *IEEE Inter-
 national Conference on Robotics and Automation*, Vol. 3,
 1998, pp. 2676–2682.
- [7] A. Seyfarth, H. Geyer, M. Günther, R. Blickhan, A
 movement criterion for running, *Journal of Biomechanics*
 35 (5) (2002) 649–655.

- [8] M. M. Ankarali, U. Saranlı, Stride-to-stride energy regulation for robust self-stability of a torque-actuated dissipative spring-mass hopper, *Chaos: An Interdisciplinary Journal of Nonlinear Science* 20 (3) (2010) 033121.
- [9] G. Secer, U. Saranlı, Control of planar spring–mass running through virtual tuning of radial leg damping, *IEEE Transactions on Robotics* 34 (5) (2018) 1370–1383.
- [10] M. H. Raibert, *Legged robots that balance*, MIT press, 1986.
- [11] C. Hubicki, A. Abate, P. Clary, S. Rezazadeh, M. Jones, A. Peekema, J. Van Why, R. Domres, A. Wu, W. Martin, H. Geyer, J. Hurst, Walking and running with passive compliance: Lessons from engineering: A live demonstration of the atrias biped, *IEEE Robotics & Automation Magazine* 25 (3) (2018) 23–39.
- [12] P. M. Wensing, D. E. Orin, High-speed humanoid running through control with a 3d-slip model, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 5134–5140.
- [13] S. Sovukluk, J. Engelsberger, C. Ott, Highly maneuverable humanoid running via 3d slip+foot dynamics, *IEEE Robotics and Automation Letters* 9 (2) (2024) 1131–1138.
- [14] F. M. Smaldone, N. Scianca, L. Lanari, G. Oriolo, From walking to running: 3d humanoid gait generation via mpc, *Frontiers in Robotics and AI* 9 (2022).
- [15] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, R. Tedrake, Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot, *Autonomous robots* 40 (2016) 429–455.
- [16] H. Dai, A. Valenzuela, R. Tedrake, Whole-body motion planning with centroidal dynamics and full kinematics, in: *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 295–302.
- [17] W.-L. Ma, A. Hereid, C. M. Hubicki, A. D. Ames, Efficient hzd gait generation for three-dimensional underactuated humanoid running, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 5819–5825.
- [18] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, M. Inaba, Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation, in: *IEEE International Conference on Robotics and Automation*, 2024, pp. 13107–13114.
- [19] S. H. Jeon, S. Heim, C. Khazoom, S. Kim, Benchmarking potential based rewards for learning humanoid locomotion, in: *IEEE International Conference on Robotics and Automation*, 2023, pp. 9204–9210.
- [20] X. Meng, Z. Yu, X. Chen, Z. Huang, C. Dong, F. Meng, Online running-gait generation for bipedal robots with smooth state switching and accurate speed tracking, *Biomimetics* 8 (1) (2023).
- [21] R. Tajima, D. Honda, K. Suga, Fast running experiments involving a humanoid robot, in: *2009 IEEE international conference on robotics and automation*, 2009, pp. 1571–1576.
- [22] T. Takenaka, T. Matsumoto, T. Yoshiike, S. Shirokura, Real time motion generation and control for biped robot -2nd report: Running gait pattern generation-, in: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1092–1099.
- [23] Q. Zhang, P. Cui, D. Yan, J. Sun, Y. Duan, G. Han, W. Zhao, W. Zhang, Y. Guo, A. Zhang, R. Xu, Whole-body humanoid robot locomotion with human reference, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 11225–11231.
- [24] Y. Sakurai, T. Kamimura, Y. Sakamoto, S. Nishii, K. Sato, Y. Fujiwara, A. Sano, Bipedal robot running: human-like actuation timing using fast and slow adaptations, *Advanced Robotics* 38 (8) (2024) 577–588.
- [25] F. Al-Hafez, G. Zhao, J. Peters, D. Tateo, Locomujoco: A comprehensive imitation learning benchmark for locomotion, in: *6th Robot Learning Workshop, NeurIPS*, 2023.
- [26] C. Tessler, Y. Guo, O. Nabati, G. Chechik, X. B. Peng, Maskedmimic: Unified physics-based character control through masked motion inpainting, *ACM Trans. Graph.* 43 (6) (Nov. 2024).
- [27] R. J. Griffin, G. Wiedebach, S. McCrory, S. Bertrand, I. Lee, J. Pratt, Footstep planning for autonomous walking over rough terrain, in: *IEEE-RAS International Conference on Humanoid Robots*, 2019, pp. 9–16.
- [28] J. Engelsberger, P. Kozłowski, C. Ott, A. Albu-Schäffer, Biologically inspired deadbeat control for running: From human analysis to humanoid control and back, *IEEE Transactions on Robotics* 32 (4) (2016) 854–867.
- [29] G. Mesesan, R. Schuller, J. Engelsberger, C. Ott, A. Albu-Schäffer, Unified motion planner for walking, running, and jumping using the three-dimensional divergent component of motion, *IEEE Transactions on Robotics* (2023) 1–21.
- [30] H. Geyer, Simple models of legged locomotion based on compliant limb behavior, Ph.D. thesis (2005).
- [31] M. Ernst, H. Geyer, R. Blickhan, Spring-legged locomotion on uneven ground: a control approach to keep the running speed constant, in: *Mobile Robotics: Solutions and Challenges*, World Scientific, 2010, pp. 639–644.

- [32] P.-B. Wieber, Holonomy and nonholonomy in the dynamics of articulated motion, in: *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*, Springer, 2006, pp. 411–425.
- [33] D. E. Orin, A. Goswami, S.-H. Lee, Centroidal dynamics of a humanoid robot, *Autonomous robots* 35 (2013) 161–176.
- [34] P. M. Wensing, D. E. Orin, Improved computation of the humanoid centroidal dynamics and application for whole-body control, *International Journal of Humanoid Robotics* 13 (01) (2016) 1550039.
- [35] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, A. Del Prete, Optimization-based control for dynamic legged robots, *IEEE Transactions on Robotics* (2023).
- [36] B. Henze, M. A. Roa, C. Ott, Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios, *The International Journal of Robotics Research* 35 (12) (2016) 1522–1543.
- [37] B. Henze, A. Dietrich, C. Ott, An approach to combine balancing with hierarchical whole-body control for legged humanoid robots, *IEEE Robotics and Automation Letters* 1 (2) (2016) 700–707.
- [38] J. Engelsberger, A. Dietrich, G.-A. Mesesan, G. Garofalo, C. Ott, A. O. Albu-Schäffer, Mptc-modular passive tracking controller for stack of tasks based control frameworks, *Robotics: Science and Systems (RSS 2020)* (2020).
- [39] L. Sentis, O. Khatib, A whole-body control framework for humanoids operating in human environments, in: *IEEE International Conference on Robotics and Automation*, 2006, pp. 2641–2648.
- [40] P. M. Wensing, D. E. Orin, Generation of dynamic humanoid behaviors through task-space control with conic optimization, in: *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3103–3109.
- [41] O. Khatib, M. Jorda, J. Park, L. Sentis, S.-Y. Chung, Constraint-consistent task-oriented whole-body robot formulation: Task, posture, constraints, multiple contacts, and balance, *The International Journal of Robotics Research* 41 (13-14) (2022) 1079–1098.
- [42] C. Ott, A. Dietrich, A. Albu-Schäffer, Prioritized multi-task compliance control of redundant manipulators, *Automatica* 53 (2015) 416–423.
- [43] P. M. Wensing, Optimization and control of dynamic humanoid running and jumping, Ph.D. thesis, The Ohio State University (2014).
- [44] J. Engelsberger, Combining reduced dynamics models and whole-body control for agile humanoid locomotion, Ph.D. thesis, Technische Universität München (2016).
- [45] S. Sovukluk, J. Engelsberger, C. Ott, Whole body control formulation for humanoid robots with closed/parallel kinematic chains: Kangaroo case study, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023, pp. 10390–10396.
- [46] S. Sovukluk, G. Zambella, T. Egle, C. Ott, Experimental comparison of whole-body control formulations for humanoid robots in task acceleration and task force spaces, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2025, pp. 20146–20153.
- [47] S. Sovukluk, R. Schuller, J. Engelsberger, C. Ott, Realtime limb trajectory optimization for humanoid running through centroidal angular momentum dynamics, in: *IEEE International Conference on Robotics and Automation*, 2025, pp. 404–410.
- [48] E. Todorov, T. Erez, Y. Tassa, Mujoco: A physics engine for model-based control, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.
- [49] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiroux, O. Stasse, N. Mansard, The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives, in: *IEEE International Symposium on System Integrations*, 2019.
- [50] S. Agarwal, K. Mierle, T. C. S. Team, Ceres solver (10 2023).
URL <https://github.com/ceres-solver/ceres-solver>

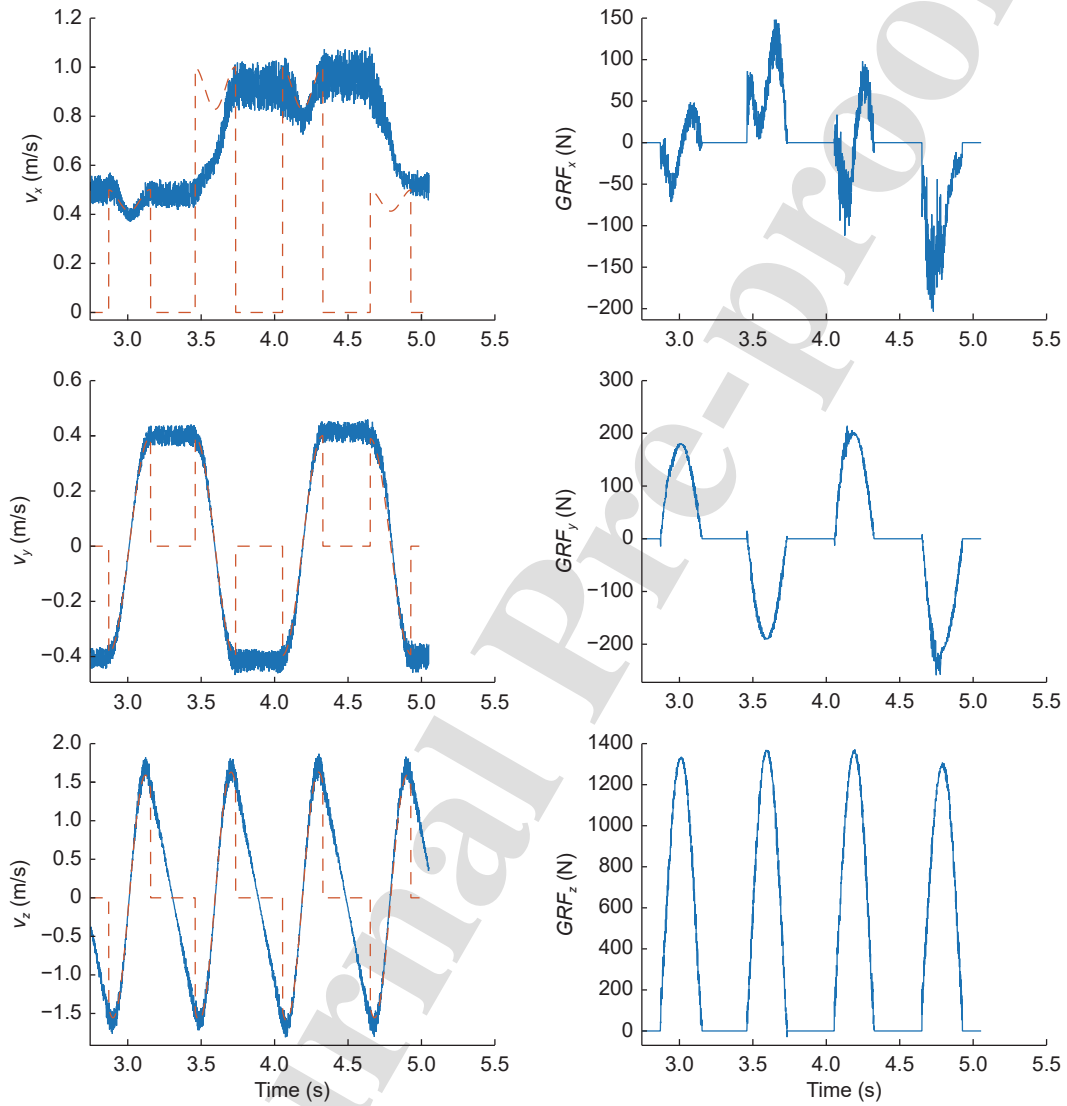


Figure 18: Kangaroo's center of mass velocities (left) and estimated ground reaction forces (right) under the influence of noise described in Fig. 17.

Declaration of Interest Statement

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The author is an Editorial Board Member/Editor-in-Chief/Associate Editor/Guest Editor for this journal and was not involved in the editorial review or the decision to publish this article.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: