Capturability-based Pattern Generation for Walking with Variable Height

Stéphane Caron, Adrien Escande, Leonardo Lanari, Bastien Mallein

Capturability-based Pattern Generation for Walking with Variable Height

Stéphane Caron, Adrien Escande, Leonardo Lanari, and Bastien Mallein

Abstract—**Capturability analysis of the linear inverted pendulum (LIP) model enabled walking with constrained height based on the *capture point*. We generalize this analysis to the variable-height inverted pendulum (VHIP) and show how it enables 3D walking over uneven terrains based on *capture inputs*. Thanks to a tailored optimization scheme, we can compute these inputs fast enough for real-time model predictive control. We implement this approach as open-source software and demonstrate it in dynamic simulations.**

Index Terms—**Bipedal walking, Capturability, Uneven terrain**

I. INTRODUCTION

Capturability quantifies the ability of a system to come to a stop. For a humanoid walking in the linear inverted pendulum (LIP) mode, it is embodied by the *capture point*, the point on the ground where the robot should step in order to bring itself to a stop [1]. In recent years, one of the main lines of research in LIP-based studies has explored the question of walking by feedforward planning and feedback control of the capture point [2], [3], [4], [5], [6], [7].

The LIP owes its tractability to two assumptions: no angular-momentum variation around the center of mass (CoM), and a holonomic constraint on the CoM height. As a consequence of the latter, a majority of LIP-based walking controllers assume a flat terrain. Removing this holonomic constraint from the LIP leads to the *variable-height inverted pendulum* (VHIP) model, for which our understanding is at an earlier stage. Previous studies [8], [9], [10] focused on its balance control for planar motions (sagittal and vertical only). In a preliminary version of this work [11], we extended the analysis from 2D to 3D balance control. In the present work, we bridge the gap from balancing to walking.

Our contribution is three-fold. First, we provide a necessary and sufficient condition for the capturability of the VHIP model (Section II). Second, we show how to turn this condition into an optimization problem (Section III) for which we develop a tailored optimization scheme (Section IV). Finally, we adapt this optimization into a model-predictive walking pattern

Stéphane Caron is with the Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), CNRS–University of Montpellier, Montpellier, France.

Adrien Escande is with the CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/RL, Japan.

Leonardo Lanari is with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Rome, Italy (e-mail: lanari@diag.uniroma1.it).

Bastien Mallein is with the Laboratoire Analyse, Géométrie et Applications (LAGA), CNRS–Paris 13 University, Villetaneuse, France.

Corresponding author: stephane.caron@lirmm.fr.

generator for rough terrains (Section V) that we demonstrate in dynamic simulations (Section VI).

II. CAPTURABILITY OF INVERTED PENDULUM MODELS

The critical part of the dynamics for a walking biped lies in the Newton-Euler equation that drives its unactuated floating-base coordinates:

$$\begin{bmatrix} \ddot{\boldsymbol{c}} \\ \dot{\mathbf{L}}_{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\boldsymbol{f} \\ \boldsymbol{\tau_c} \end{bmatrix} + \begin{bmatrix} \boldsymbol{g} \\ \mathbf{0} \end{bmatrix} \tag{1}$$

where $\boldsymbol{c}$ is the position of the center of mass (CoM) of the robot, $\boldsymbol{g}$ is the gravity vector (also written $\boldsymbol{g} = -g\boldsymbol{e}_z$ with g the gravitational constant), m is the total robot mass and $\mathbf{L}_{\boldsymbol{c}}$ is the angular momentum around $\boldsymbol{c}$. The net contact wrench $(\boldsymbol{f}, \boldsymbol{\tau_c})$ consists of the resultant $\boldsymbol{f}$ of external contact forces and their moment $\boldsymbol{\tau_c}$ around the CoM.

A. Inverted pendulum models

The linear inverted pendulum (LIP) [12] model is based on two constraints: a constant angular momentum around the center of mass, and a constant CoM height with respect to a reference plane:

$$\dot{\mathbf{L}}_{\boldsymbol{c}} = \mathbf{0} \tag{2}$$
$$\boldsymbol{n} \cdot (\boldsymbol{c} - \boldsymbol{o}) = h \tag{3}$$

where $\boldsymbol{n}$, $\boldsymbol{o}$ and h are respectively the normal vector, reference point and reference height that define the CoM motion plane. As a consequence of these two assumptions, the Newton-Euler equation (1) simplifies[1] to:

$$\ddot{\boldsymbol{c}} = \omega_{\mathrm{LIP}}^2 (\boldsymbol{c} - \boldsymbol{r}) + \boldsymbol{g} \tag{4}$$

where $\omega_{\mathrm{LIP}} := \sqrt{g/h}$ is a constant and $\boldsymbol{r}$ is the center of pressure (CoP), or zero-tilting moment point (ZMP) [14] when there are multiple contacts. A strong limitation of the LIP is the holonomic constraint (3) on the CoM, which can be kinematically problematic in scenarios such as stair climbing. One line of research sought to overcome this by constraining the CoM to parametric or piecewise-planar surfaces [15], [16], but the next question of deciding such surfaces based on terrain topology has not attracted a lot of attention so far.

The variable-height inverted pendulum (VHIP) model strips away the holonomic CoM constraint altogether [17], [18]. Its equation of motion is:

$$\ddot{\boldsymbol{c}} = \lambda(\boldsymbol{c} - \boldsymbol{r}) + \boldsymbol{g} \tag{5}$$

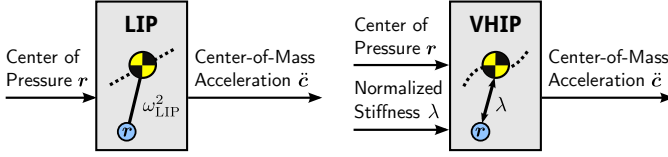[1] See *e.g.* [13] for a reminder of the steps of this derivation.

Fig. 1. Inputs and output of the *linear inverted pendulum* (LIP) and *variable-height inverted pendulum* (VHIP) models. In the former, the stiffness coefficient is fixed to a constant ω_{LIP}^2, while it becomes an additional input λ of the latter.

where $\lambda > 0$ is now a time-varying stiffness[2] coefficient. The two control inputs of this system are the center of pressure $\boldsymbol{r}$ and the stiffness λ, as shown in Figure 1.

B. Feasibility conditions

To be *feasible*, the CoP $\boldsymbol{r}$ must belong to the contact area $\mathcal{C}$ under the supporting foot. This area is also time-varying but changes only discretely. The transition from one support contact to the next is called a *contact switch*. A trajectory with N contact switches is called an N-step trajectory.

We assume that all contact areas are planar and polygonal. Let us denote by $\boldsymbol{o}$ the center of the area $\mathcal{C}$ and by $\boldsymbol{n}$ its normal (such that $\boldsymbol{n} \cdot \boldsymbol{e}_z \neq 0$). The CoP $\boldsymbol{r}$ belongs to the plane of contact if and only if $(\boldsymbol{r} - \boldsymbol{o}) \cdot \boldsymbol{n} = 0$. The *height* of the CoM $\boldsymbol{c}$ above the contact area is the algebraic distance $h(\boldsymbol{c})$ such that $\boldsymbol{c} - h(\boldsymbol{c})\boldsymbol{e}_z$ belongs to the contact plane:

$$h(\boldsymbol{c}) := \frac{(\boldsymbol{c} - \boldsymbol{o}) \cdot \boldsymbol{n}}{(\boldsymbol{e}_z \cdot \boldsymbol{n})} \tag{6}$$

Note how, when walking on a horizontal floor, $\boldsymbol{e}_z$ and $\boldsymbol{n}$ are aligned and h is simply the z coordinate of the center of mass.

To be *feasible*, the stiffness λ must be non-negative by unilaterality of contact. We furthermore impose that $\lambda \in [\lambda_{\min}, \lambda_{\max}]$ so that contact pressure is never exactly zero and remains bounded. Note that we do not model Coulomb friction conditions here: having found in a previous work that CoP feasibility constraints are usually more stringent than friction constraints when walking over uneven terrains [19], we assume sufficient friction in the present study.

An input function $t \mapsto (\lambda(t), \boldsymbol{r}(t))$ is *feasible* when both $\lambda(t)$ and $\boldsymbol{r}(t)$ are feasible at all times t. A general control problem is to find a feasible input function such that the resulting output trajectory $\boldsymbol{c}(t)$ has certain properties. For the locomotion problem of "getting somewhere", we will focus on the property of converging to a desired location.

C. Capture inputs and capture trajectories

A natural choice of the pendulum state consists of its CoM position and velocity $\boldsymbol{x} = (\boldsymbol{c}, \dot{\boldsymbol{c}})$.

Definition 1 (Static equilibrium). *A state $\boldsymbol{x} = (\boldsymbol{c}, \dot{\boldsymbol{c}})$ is a static equilibrium when its velocity $\dot{\boldsymbol{c}}$ is zero and can remain zero with suitable constant controls $(\lambda_{\mathrm{f}}, \boldsymbol{r}_{\mathrm{f}})$.*

---

[2] All quantities being normalized by mass, we call λ a stiffness although its unit is s^{-2} and not $\mathrm{kg.s}^{-2}$. Similarly, we will refer to frequencies (unit: s^{-1}) as dampings.

Static equilibria, also called *capture states* [1], are the targets of capturability analysis, the desired locations that the CoM should converge to. A static equilibrium is characterized by its CoM position $\boldsymbol{c}_{\mathrm{f}}$ and the contact $\mathcal{C}_{\mathrm{f}}$ upon which it rests at height $h_{\mathrm{f}} = h(\boldsymbol{c}_{\mathrm{f}})$. The only control input $\lambda_{\mathrm{f}}, \boldsymbol{r}_{\mathrm{f}}$ that maintains the pendulum in static equilibrium is such that $\boldsymbol{c}_{\mathrm{f}} = \boldsymbol{r}_{\mathrm{f}} - \boldsymbol{g}/\lambda_{\mathrm{f}}$, that is:

$$\lambda_{\mathrm{f}}(\boldsymbol{c}_{\mathrm{f}}) = \frac{g}{h_{\mathrm{f}}} \qquad \boldsymbol{r}_{\mathrm{f}}(\boldsymbol{c}_{\mathrm{f}}) = \boldsymbol{c}_{\mathrm{f}} - h_{\mathrm{f}}\boldsymbol{e}_z \tag{7}$$

Given an N-step contact sequence, we say that a state $\boldsymbol{x}_{\mathrm{i}}$ is (N-step) *capturable* when there exists a feasible input function $\lambda(t), \boldsymbol{r}(t)$ such that applying Equation (5) from $\boldsymbol{x}_{\mathrm{i}}$ brings the system asymptotically to an equilibrium $\boldsymbol{x}_{\mathrm{f}}$. We call such functions *capture inputs* of the capturable state $\boldsymbol{x}_{\mathrm{i}}$, and denote their set by $\mathcal{I}_{\boldsymbol{x}_{\mathrm{i}}, \boldsymbol{x}_{\mathrm{f}}}$. We call *capture trajectory* the CoM trajectory $\boldsymbol{c}(t)$ resulting from a capture input. In what follows, we use the subscript $\square_{\mathrm{i}}$ to denote the "initial" or instantaneous state of the system, and the subscript $\square_{\mathrm{f}}$ for its "final" or asymptotic state.

By definition, $\boldsymbol{x}_{\mathrm{i}}$ is capturable if and only if there exists $\boldsymbol{x}_{\mathrm{f}}$ such that $\mathcal{I}_{\boldsymbol{x}_{\mathrm{i}}, \boldsymbol{x}_{\mathrm{f}}} \neq \emptyset$. The set $\mathcal{I}_{\boldsymbol{x}_{\mathrm{i}}, \boldsymbol{x}_{\mathrm{f}}}$ contains however several solutions, including mathematical oddities such as functions with ever-increasing frequencies. In what follows, we restrict it to inputs that converge asymptotically:

$$\mathcal{I}_{\boldsymbol{x}_{\mathrm{i}}, \boldsymbol{x}_{\mathrm{f}}}^c = \left\{ (\lambda(t), \boldsymbol{r}(t)) \in \mathcal{I}_{\boldsymbol{x}_{\mathrm{i}}, \boldsymbol{x}_{\mathrm{f}}} : \begin{array}{l} \lim_{t \to \infty} \lambda(t) = \lambda_{\mathrm{f}} \\ \lim_{t \to \infty} \boldsymbol{r}(t) = \boldsymbol{r}_{\mathrm{f}} \end{array} \right\} \tag{8}$$

Property 5 shows that this choice does not cause any loss of generality (see Appendix A).

D. Dichotomy of the components of motion

One of the main findings in the control of the LIP model is to focus on its divergent component of motion (DCM), the capture point. Among other benefits, controlling only the capture point reduces second-order dynamics to first order and maximizes the basin of attraction of feedback controllers [2], [3], [4]. An important step in the analysis of the VHIP model is therefore to identify its DCM. Interestingly, the answer can be found in a study of motorcycle balance [20], which we recall here.

The equation of motion (5) of the VHIP can be interpreted as either nonlinear or linear time-variant, depending on whether one focuses respectively on feedback or open-loop control. We focus on the latter for walking pattern generation. Let us then rewrite this equation as a first-order linear time-variant system:

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \ddot{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \lambda\mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{c} \\ \dot{\boldsymbol{c}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{g} - \lambda\boldsymbol{r} \end{bmatrix} \tag{9}$$

where $\mathbf{I}$ is the 3×3 identity matrix. This equation has the form $\dot{\boldsymbol{x}} = \mathbf{A}(t)\boldsymbol{x} + \boldsymbol{b}(t)$ where the system matrix $\mathbf{A}$ depends on the stiffness input λ, while the forcing term $\boldsymbol{b}$ varies with both inputs λ and $\boldsymbol{r}$.

Hauser *et al.* showed [20] how to obtain an *exponential dichotomy* [21] of the state $\boldsymbol{x}$ (that is, how to decompose it into

convergent and divergent components) by applying a change of coordinates $\boldsymbol{x} = \mathbf{S}\boldsymbol{z}$ with:

$$\mathbf{S} = \frac{1}{\gamma + \omega}\begin{bmatrix} \mathbf{I} & \mathbf{I} \\ -\omega\mathbf{I} & \gamma\mathbf{I} \end{bmatrix} \Longleftrightarrow \mathbf{S}^{-1} = \begin{bmatrix} \gamma\mathbf{I} & -\mathbf{I} \\ \omega\mathbf{I} & +\mathbf{I} \end{bmatrix} \quad (10)$$

The two functions $\gamma(t)$ and $\omega(t)$ are *positive* and of class $\mathcal{C}^1$. We will refer to them as *dampings* in accordance with their (normalized) physical unit. The new state vector $\boldsymbol{z} = \mathbf{S}^{-1}\boldsymbol{x}$ then consists of two components $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ defined by:

$$\boldsymbol{\zeta} = \gamma\boldsymbol{c} - \dot{\boldsymbol{c}} \quad (11)$$

$$\boldsymbol{\xi} = \omega\boldsymbol{c} + \dot{\boldsymbol{c}} \quad (12)$$

They respectively correspond to the *convergent* and *divergent component of motion* (DCM). In the case of a linear inverted pendulum with constant ω_{LIP}, the DCM $\boldsymbol{\xi}$ is simply proportional to the capture point $\boldsymbol{c} + \dot{\boldsymbol{c}}/\omega_{\text{LIP}}$.

Collectively, the state vector $\boldsymbol{z}$ is subject to:

$$\dot{\boldsymbol{z}} = \tilde{\mathbf{A}}\boldsymbol{z} + \tilde{\boldsymbol{b}} \quad (13)$$

$$\tilde{\mathbf{A}} = \mathbf{S}^{-1}(\mathbf{A}\mathbf{S} - \dot{\mathbf{S}}) \quad (14)$$

$$\tilde{\boldsymbol{b}} = \mathbf{S}^{-1}\boldsymbol{b} \quad (15)$$

The calculation of $\tilde{\boldsymbol{b}}$ is straightforward. That of $\tilde{\mathbf{A}}$ yields:

$$\tilde{\mathbf{A}} = \frac{1}{\gamma + \omega}\begin{bmatrix} (\dot{\gamma} - \gamma\omega - \lambda)\mathbf{I} & (\dot{\gamma} + \gamma^2 - \lambda)\mathbf{I} \\ (\dot{\omega} - \omega^2 + \lambda)\mathbf{I} & (\dot{\omega} + \omega\gamma + \lambda)\mathbf{I} \end{bmatrix} \quad (16)$$

To decouple the system, we can eliminate non-diagonal terms in this state matrix by imposing the two following Riccati equations:

$$\dot{\gamma} = \lambda - \gamma^2 \quad (17)$$

$$\dot{\omega} = \omega^2 - \lambda \quad (18)$$

This results in the following state dynamics:

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{\boldsymbol{\zeta}} \\ \dot{\boldsymbol{\xi}} \end{bmatrix} = \begin{bmatrix} -\gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \omega\mathbf{I} \end{bmatrix}\begin{bmatrix} \boldsymbol{\zeta} \\ \boldsymbol{\xi} \end{bmatrix} + \begin{bmatrix} \lambda\boldsymbol{r} - \boldsymbol{g} \\ \boldsymbol{g} - \lambda\boldsymbol{r} \end{bmatrix} \quad (19)$$

The linear time-varying system has thus been decoupled into two linearly independent components $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ that evolve according to their own dynamics, provided that there exists two $\mathcal{C}^1$ positive finite solutions to (17) and (18). A proof of this and a detailed analysis of damping functions are given in Appendix A-B.

Although we defer the detailed analysis of damping functions to this Appendix, its takeaway point is that γ and ω are, in themselves, convergent and divergent components. A parallel can be drawn between the DCM–CoP and ω–λ systems:

- In the LIP, with the CoP restricted to a support area, the CoP is a repulsor of the DCM, and the DCM is controllable if and only if it is above the support area [3].
- In the VHIP, with λ restricted to $[\lambda_{\min}, \lambda_{\max}]$, λ is a repulsor of ω, and ω is controllable if and only if it belongs to $[\sqrt{\lambda_{\min}}, \sqrt{\lambda_{\max}}]$ (Property 6).

This remark is central to the walking pattern generation method in Section III, which reduces three-dimensional capturability to one dimension.

It can be shown that, regardless of the initial state $\boldsymbol{x}_i$ of the system, *any* input function from $\mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}^c$ makes $\boldsymbol{\zeta}$ converge as

well (Property 9), owing it its name of convergent component of motion. From a control perspective, spending additional inputs to control this component is not necessary and can even be wasteful.[3] The main concern of capturability analysis is therefore to prevent the other component $\boldsymbol{\xi}$ from diverging.

E. Boundedness condition

The divergent component of motion $\boldsymbol{\xi}$ corresponding to the damping ω is subject to the differential equation:

$$\dot{\boldsymbol{\xi}} = \omega\boldsymbol{\xi} + \boldsymbol{g} - \lambda\boldsymbol{r} \quad (20)$$

The general solution to this equation is given by:

$$\boldsymbol{\xi}(t) = \left(\boldsymbol{\xi}(0) + \int_0^t e^{-\Omega(\tau)}(\boldsymbol{g} - \lambda(\tau)\boldsymbol{r}(\tau))\mathrm{d}\tau\right)e^{\Omega(t)} \quad (21)$$

where $\Omega(t) = \int_0^t \omega(t)\mathrm{d}t$. In our working assumptions, this integral is well-defined and finite (details in Appendix A-D).

Set aside the particular condition that we are about to discuss, the function $\boldsymbol{\xi}(t)$ diverges as $t \to \infty$, giving $\boldsymbol{\xi}$ its name of divergent component of motion (DCM) [4].[4] However, a careful match between future capture inputs and the initial condition $\boldsymbol{\xi}_i$ can guarantee that $\boldsymbol{\xi}(t)$ converges as well. This choice is known as the *boundedness condition* [22]:

Property 1 (Boundedness condition). *Consider an input function $\lambda(t), \boldsymbol{r}(t)$ such that $\lim_{t\to\infty} \lambda(t) = \lambda_f$ and $\lim_{t\to\infty} \boldsymbol{r}(t) = \boldsymbol{r}_f$. Then, there exists a unique $\boldsymbol{\xi}_i = \omega_i\boldsymbol{c}_i + \dot{\boldsymbol{c}}_i$ such that the solution $\boldsymbol{\xi}$ of (20) with $\boldsymbol{\xi}(0) = \boldsymbol{\xi}_i$ remains finite at all times. This initial condition is given by:*

$$\boxed{\boldsymbol{\xi}_i = \int_0^\infty e^{-\Omega(t)}(\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})\mathrm{d}t} \quad (22)$$

where ω_i is the initial value of the unique bounded solution ω to the Riccati equation $\dot{\omega} = \omega^2 - \lambda$ and Ω is the antiderivative of ω such that $\Omega(0) = 0$.

The proof of this property is given in Appendix A-D. In the familiar setting of the LIP where $\lambda = \omega_{\text{LIP}}^2$ is a constant, taking a constant CoP $\boldsymbol{r}_c$ yields:

$$\omega_{\text{LIP}}\boldsymbol{c}_i + \dot{\boldsymbol{c}}_i = \int_0^\infty (\omega_{\text{LIP}}^2\boldsymbol{r}_c - \boldsymbol{g})e^{-\omega_{\text{LIP}}t}\mathrm{d}t \quad (23)$$

$$= \omega_{\text{LIP}}\boldsymbol{r}_c - \frac{\boldsymbol{g}}{\omega_{\text{LIP}}} \quad (24)$$

Over horizontal coordinates, this equation implies that $\boldsymbol{r}_c^{xy} = \boldsymbol{c}_i^{xy} + \dot{\boldsymbol{c}}_i^{xy}/\omega_{\text{LIP}}$, *i.e.* the CoP is located at the capture point. Over the z coordinate, it yields $\omega_{\text{LIP}} = \sqrt{g/h}$, the known expression of the natural frequency of the LIP. Overall, the boundedness condition characterizes the capturability of the LIP. We will now conclude our capturability analysis by showing how this is also the case for the VHIP.

---

[3] For the LIP, it reduces the basin of attraction of feedback controllers [3].

[4] More specifically, our analysis considers a *time-varying* divergent component of motion [17]. While previous works such as [4], [6], [17] chose to write their DCMs as positions $\boldsymbol{c}(t) + \dot{\boldsymbol{c}}(t)/\omega(t)$, we cast them as velocities here to simplify calculations (consider the derivative of a product uv compared to that of a ratio u/v). The formula of the DCM itself is not a crucial design choice, as we will discuss at the end of this Section.

F. Capturability of the variable-height inverted pendulum

What we have established so far is a necessary condition: if an input function belongs to $\mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}^c$, then it is feasible, converging and satisfies the boundedness condition. The key result of our capturability analysis is that this condition is also sufficient, *i.e.* it *characterizes* capture inputs:

Property 2. *Let* $\boldsymbol{x}_i = (\boldsymbol{c}_i, \dot{\boldsymbol{c}}_i)$ *denote a capturable state and* $\boldsymbol{x}_f = (\boldsymbol{c}_f, \boldsymbol{0})$ *a static equilibrium. Then,* $t \mapsto \lambda(t), \boldsymbol{r}(t)$ *is a capture input from* $\boldsymbol{x}_i$ *to* $\boldsymbol{x}_f$ *if and only if:*

(i) its values $\lambda(t)$ *and* $\boldsymbol{r}(t)$ *are feasible for all* $t \geq 0$,
(ii) $\lim_{t \to \infty} \lambda(t) = \lambda_f(\boldsymbol{c}_f)$ *and* $\lim_{t \to \infty} \boldsymbol{r}(t) = \boldsymbol{r}_f(\boldsymbol{c}_f)$,
(iii) it satisfies the boundedness condition:

$$\int_0^\infty (\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})e^{-\Omega(t)}\mathrm{d}t = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \quad (25)$$

where ω_i *is the initial value of the unique bounded solution* ω *to the Riccati equation* $\dot{\omega} = \omega^2 - \lambda$ *and* Ω *is the antiderivative of* ω *such that* $\Omega(0) = 0$.

A proof of this property is given in Appendix A-E. In the remainder of this manuscript, we will see how this is not only a theoretical but also a practical result with applications to balance control and walking pattern generation.

A noteworthy methodological point here is that the expression of the divergent component of motion is not unique. Rather, *a* DCM is chosen by the roboticist. For example, in [11] we considered a different DCM $\widetilde{\boldsymbol{\xi}} := \omega(\boldsymbol{c} - \boldsymbol{r}) + \dot{\boldsymbol{c}} - \dot{\boldsymbol{r}}$ yielding a boundedness condition written:

$$\int_0^\infty (\ddot{\boldsymbol{r}}(t) - \boldsymbol{g})e^{-\Omega(t)}\mathrm{d}t = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \quad (26)$$

This condition is the same as (25), which can be seen by applying a double integration by parts. In the present work, we chose the DCM from Equation (20) as it makes calculations simpler. We preferred a velocity-based rather than position-based DCM for the same reason, as the differential of a product involves less operations than that of a ratio.

III. BALANCE CONTROL WITH VARIABLE HEIGHT

Let us consider first the problem of balance control, *i.e.* zero-step capturability. The robot pushes against a stationary contact area $\mathcal{C}$ in order to absorb the linear momentum of its initial state $\boldsymbol{x}_i$, eventually reaching a static equilibrium $\boldsymbol{x}_f$. This level of capturability enables push recovery [1], [23], [10], [24] up to post-impact fall recovery in worst-case scenarios [25], [26].

The gist of the method we propose thereafter is to reduce the three-dimensional capturability condition over $\lambda, \boldsymbol{r}$ (Property 2) into a one-dimensional condition over λ. To do so, we couple the evolution of λ and $\boldsymbol{r}$ by a suitably-defined intermediate variable s. The complete pipeline goes as follows:

- Change variable from time t to a new variable s
- Define the CoP evolution $\boldsymbol{r}(s)$ as a function of s
- Reduce capturability to an optimization over $\omega(s)$
- Compute the optimal solution $\omega^*(s)$ of this problem
- Change variable from s to time t

From $\omega^*(t)$, it is then straightforward to compute the full capture input $t \mapsto \lambda^*(t), \boldsymbol{r}^*(t)$ as well as the capture trajectory
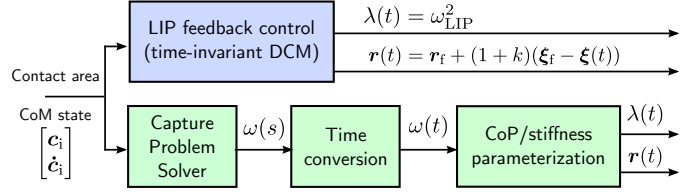

Fig. 2. **Comparison between LIP and VHIP balance control.** With the linear inverted pendulum model, the natural frequency ω_{LIP} is constant and feedback control is realized by the center of pressure $\boldsymbol{r}(t)$ (ankle strategy). With the variable-height inverted pendulum (VHIP), a time-varying natural frequency $\omega(t)$ is computed, then converted to time-varying CoP $\boldsymbol{r}(t)$ (ankle strategy) and pendulum stiffness $\lambda(t)$ (height-variation strategy).

$\boldsymbol{c}^*(t)$. Figure 2 compares this pipeline with the traditional LIP feedback control strategy with time-invariant DCM. Let us now detail each step of this process.

A. Change of variable

Define the adimensional quantity:

$$s(t) = e^{-\Omega(t)} \quad (27)$$

This new variable ranges from $s = 1$ when $t = 0$ to $s \to 0$ when $t \to \infty$. Its time derivatives are:

$$\dot{s}(t) = -\omega(t)s(t) \qquad \ddot{s}(t) = \lambda(t)s(t) \quad (28)$$

Owing to the bijective mapping between t and s, we can define ω, γ and λ as functions of s rather than as functions of t. This approach is *e.g.* common in time-optimal control [27]. Let us denote by $\square'$ derivation with respect to s, as opposed to $\dot{\square}$ for derivation with respect to t. The Riccati equation (18) of ω becomes:

$$\lambda = \omega^2 - \dot{\omega} = \omega^2 - \dot{s}\omega' = \omega(\omega + s\omega') = \omega(s\omega)' \quad (29)$$

Injecting this expression into the time integral (25) of the boundedness condition yields:

$$\int_0^\infty (\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})s(t)\mathrm{d}t = \int_0^1 (\omega(s\omega)'\boldsymbol{r}(s) - \boldsymbol{g})\frac{\mathrm{d}s}{\omega} \quad (30)$$

We can then characterize capture inputs as functions of s:

Property 3. *Let* $\boldsymbol{x}_i = (\boldsymbol{c}_i, \dot{\boldsymbol{c}}_i)$ *denote a capturable state and* $\boldsymbol{x}_f = (\boldsymbol{c}_f, \boldsymbol{0})$ *a static equilibrium. Then,* $s \mapsto \lambda(s), \boldsymbol{r}(s)$ *is a capture input from* $\boldsymbol{x}_i$ *to* $\boldsymbol{x}_f$ *if and only if:*

(i) its values $\lambda(s)$ *and* $\boldsymbol{r}(s)$ *are feasible for all* $s \in [0, 1]$,
(ii) $\lim_{s \to 0} \lambda(s) = \lambda_f(\boldsymbol{c}_f)$ *and* $\lim_{s \to 0} \boldsymbol{r}(s) = \boldsymbol{r}_f(\boldsymbol{c}_f)$,
(iii) it satisfies the boundedness condition:

$$\int_0^1 \boldsymbol{r}(s)(s\omega)'\mathrm{d}s - \boldsymbol{g}\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \quad (31)$$

where ω_i *denotes the initial value (at* $s = 1$*) of the solution* ω *to the differential equation* $\omega(s\omega)' = \lambda$.

With this reformulation, the infinite-time integral has become finite over the $[0, 1]$ interval and the antiderivative Ω has been replaced by ω itself.
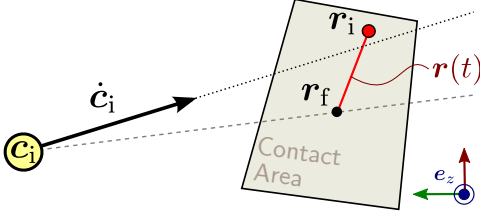
Fig. 3. **Linear time-varying CoP trajectory.** Variations of the center of pressure inside the contact area allow the robot to reorient its velocity towards a capture state without re-stepping.

B. Time-varying CoP strategy

The CoP and gravity terms of the boundedness condition (31) can be separated by projecting them in the (non-orthogonal) basis (e_x, e_y, n):

$$\int_0^1 r^{xy}(s)(s\omega)' \mathrm{d}s = \omega_i c_i^{xy} + \dot{c}_i^{xy} \qquad (32)$$

$$\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \frac{\omega_i h_i + \dot{h}_i}{g} \qquad (33)$$

where $h_i := h(c_i)$ is the initial CoM height and $\dot{h}_i$ its velocity, both known from the initial state c_i. The gravity term (33) only involves $\omega(s)$, but the CoP term (32) involves both $r(s)$ and $\omega(s)$. We reduce it to an integral over $\omega(s)$ by making the CoP move along a line segment[5] from r_i to r_f:

$$r(s) = r_f + (r_i - r_f)f(s\omega) \qquad (34)$$

where f can be any smooth function that satisfies:

- $f(\omega_i) = 1$: the CoP is initially located at r_i,
- $f(0) = 0$: the CoP converges to r_f,
- f *is increasing*: we exclude solutions where the CoP would move back and forth along the line segment,
- f *is integrable*: let F denote its antiderivative $F(x) := \int_0^x f(y)\mathrm{d}y$. It is positive by positivity of f.

The final CoP location r_f is already known from the desired capture state x_f, where the CoM is at the vertical above contact with zero velocity, but the instantaneous CoP r_i is a decision variable. In the example depicted in Figure 3, it will be chosen on the other side of the line $c_i + \mathbb{R}\dot{c}_i$ compared to r_f in order to progressively reorient $\dot{c}(t)$ toward the capture state r_f, similarly to the behavior observed in the LIP with linear capture-point feedback control [3], [5], [6].

Combining Equations (32) and (34) yields:

$$r_i^{xy} = r_f^{xy} + \frac{\omega_i(c_i^{xy} - r_f^{xy}) + \dot{c}_i^{xy}}{F(\omega_i)} \qquad (35)$$

At this stage, the roboticist can explore different CoP strategies via the choice of a function F. We choose a power law parameterized by $\alpha \in (0, 1)$:

$$f(s\omega) = \left(\frac{s\omega}{\omega_i}\right)^{\frac{\alpha}{1-\alpha}} \Rightarrow F(s\omega) = (1-\alpha)\omega_i \left(\frac{s\omega}{\omega_i}\right)^{\frac{1}{1-\alpha}} \qquad (36)$$

[5] Strategies with two control points such as this one are the simplest one can imagine, in the sense that it is in general impossible to realize three-dimensional balance control with a stationary CoP (see Appendix A-F).

With this choice, $F(\omega_i) = (1-\alpha)\omega_i$ and the horizontal coordinates of the instantaneous CoP imposed by the boundedness condition become:

$$r_i^{xy} = r_f^{xy} + \frac{1}{1-\alpha}\left[c_i^{xy} + \frac{\dot{c}_i^{xy}}{\omega_i} - r_f^{xy}\right] \qquad (37)$$

where we recognize the same expression as in capture-point feedback control of the LIP [3], [5], [6]. Note that the three-dimensional position r_i of the CoP is readily available from r_i^{xy} by vertical projection:

$$r_i = r_i^{xy} - h(r_i^{xy})e_z \qquad (38)$$

The current state x_i and target capture state x_f being given, the only decision variable left on the right-hand side of Equation (37) is ω_i. At this point, we have almost reduced the CoP capturability conditions to ω: with respect to Property 3, $r(s)$ converges to r_f (*ii*) and satisfies the boundedness condition by selecting r_i from Equation (37) (*iii*). We now need to make sure that the CoP trajectory is feasible (*i*).

By convexity of the contact area, the CoP trajectory is feasible if and only if both its ends r_i and r_f are in the area. We assume the latter does by construction. For the former, the constraint that r_i belongs to the contact polygon can be described in halfspace representation by a matrix-vector inequality $\mathbf{H}r_i^{xy} \leq p$, with $\mathbf{H}$ an $m \times 2$ matrix and p an m-dimensional vector. For example, a rectangular contact area written in the contact frame (t, b, n) as:

$$\pm t \cdot (r_i - o) \leq X \qquad (39)$$
$$\pm b \cdot (r_i - o) \leq Y \qquad (40)$$

can be reformulated equivalently in the horizontal plane:

$$\pm(b \times e_z)(r_i^{xy} - o^{xy}) \leq X(e_z \cdot n) \qquad (41)$$
$$\pm(t \times e_z)(r_i^{xy} - o^{xy}) \leq Y(e_z \cdot n) \qquad (42)$$

Injecting Equation (37) into inequalities $\mathbf{H}r_i^{xy} \leq p$ yields:

$$[\alpha\mathbf{H}r_f^{xy} + (1-\alpha)p - \mathbf{H}c_i^{xy}]\omega_i \geq \mathbf{H}\dot{c}_i^{xy} \qquad (43)$$

Each line of this vector inequality $u\omega_i \geq v$ provides a lower or upper bound on ω_i depending on the sign of the factor in front of it:

$$\omega_{i,\min} = \max\left(\sqrt{\lambda_{\min}}, \max_j\left\{\frac{v_j}{u_j}, u_j > 0\right\}\right) \qquad (44)$$

$$\omega_{i,\max} = \min\left(\sqrt{\lambda_{\max}}, \min_j\left\{\frac{v_j}{u_j}, u_j < 0\right\}\right) \qquad (45)$$

We have thus reduced the feasibility condition on the CoP to an inequality constraint on ω_i:

$$\forall s, r(s) \in \mathcal{C} \iff \omega_{i,\min} \leq \omega_i \leq \omega_{i,\max} \qquad (46)$$

Overall, the CoP strategy allows us to reduce the three-dimensional capturability condition over $s \mapsto \lambda(s), r(s)$ (Property 3) into a one-dimensional condition over $s \mapsto \lambda(s)$:

Property 4 (1D capturability). *Let* $x_i = (c_i, \dot{c}_i)$ *denote a capturable state and* $x_f = (c_f, 0)$ *a static equilibrium. Under the time-varying CoP strategy:*

$$r(s) = r_f + (r_i - r_f)\left(\frac{s\omega}{\omega_i}\right)^{\frac{\alpha}{1-\alpha}} \qquad \alpha \in (0, 1), \qquad (47)$$

$s \mapsto \lambda(s)$ *is a capture input from* $\boldsymbol{x}_\mathrm{i}$ *to* $\boldsymbol{x}_\mathrm{f}$ *if and only if:*

(i) $\omega_\mathrm{i} \in [\omega_\mathrm{i,min}, \omega_\mathrm{i,max}]$ *and* $\forall s \in [0,1], \lambda(s) \in [\lambda_\mathrm{min}, \lambda_\mathrm{max}]$,
(ii) $\lim_{s \to 0} \lambda(s) = \lambda_\mathrm{f}(\boldsymbol{c}_\mathrm{f})$,
(iii) *it satisfies the boundedness condition:*

$$\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \frac{\omega_\mathrm{i} h_\mathrm{i} + \dot{h}_\mathrm{i}}{g} \tag{48}$$

where ω_i *denotes the initial value (at* $s = 1$*) of the solution* ω *to the differential equation* $\omega(s\omega)' = \lambda$.

C. Formulation as an optimization problem

Let us compute piecewise-constant functions $s \mapsto \lambda(s)$ that satisfy the three conditions from Property 4. We partition the interval $[0,1]$ into $n-1$ fixed segments $0 = s_0 < s_1 < \ldots < s_{n-1} < s_n = 1$ such that $\forall s \in (s_j, s_{j+1}], \lambda(s) = \lambda_j$. Note how the interval is closed to the right ($\lambda(s_{j+1}) = \lambda_j$) but open to the left. Define:

$$\varphi(s) := s^2 \omega^2 \qquad \delta_j := s_{j+1}^2 - s_j^2 \tag{49}$$

The quantity φ represents a squared velocity and is commonly considered in time-optimal retiming [27], while δ_j's can be interpreted as corresponding discretization steps. In practice, we use $s_j = j/n$ so that $\delta_j = (2j+1)/n^2$, which is one choice among many. Remarking that $\varphi' = 2s\lambda$ from the Riccati equation (29), we can compute $\varphi(s)$ for $s \in [s_j, s_{j+1}]$ as:

$$\varphi(s) = \sum_{k=0}^{j-1} \lambda_k \delta_k + \lambda_j(s^2 - s_j^2) = \varphi(s_j) + \lambda_j(s^2 - s_j^2) \tag{50}$$

In what follows, we use the shorthand $\varphi_j := \varphi(s_j)$. The values $\lambda(s)$ and $\omega(s)$ for $s \in (s_j, s_{j+1}]$ can be computed back from φ using Equations (49) and (50):

$$\lambda_j = \frac{\varphi_{j+1} - \varphi_j}{\delta_j} \qquad \omega(s) = \frac{1}{s}\sqrt{\varphi_j + \lambda_j(s^2 - s_j^2)} \tag{51}$$

Owing to this property, we choose the vector $\boldsymbol{\varphi} := \begin{bmatrix} \varphi_1 \ldots \varphi_n \end{bmatrix}$ to be the decision variable of our optimization problem. Note that this vector starts from φ_1, as $\varphi_0 = 0$ by definition.

a) Feasibility (i): noting how $\varphi_n = \omega_\mathrm{i}^2$ from the equation above, both feasibility conditions can be expressed as:

$$\omega_\mathrm{i,min}^2 \leq \varphi_n \leq \omega_\mathrm{i,max}^2 \tag{52}$$
$$\forall j < n, \; \lambda_\mathrm{min}\delta_j \leq \varphi_{j+1} - \varphi_j \leq \lambda_\mathrm{max}\delta_j \tag{53}$$

b) Convergence (ii): $\lambda(s)$ converges to $\lim_{s \to 0}\lambda(s) = \lambda_1 = \varphi_1/\delta_0$. Convergence to λ_f can thus be expressed as:

$$\varphi_1 = \delta_0 \lambda_\mathrm{f} = \frac{\delta_0 g}{h_\mathrm{f}} \tag{54}$$

(Recall that φ_1 corresponds to the last time interval by definition of s.) The parameter h_f corresponds to the CoM height of the capture state.

Capture Problem Equation (59)

Parameters:
- Feasibility bounds $(\lambda_\mathrm{min}, \lambda_\mathrm{max})$ and $(\omega_\mathrm{i,min}, \omega_\mathrm{i,max})$
- Initial height h_i, its derivative $\dot{h}_\mathrm{i}$, and target height h_f
- Discretization steps $\delta_1, \ldots, \delta_n$

$$\underset{\boldsymbol{\varphi} \in \mathbb{R}^n}{\text{minimize}} \; \sum_{j=1}^{n-1} \left[\frac{\varphi_{j+1} - \varphi_j}{\delta_j} - \frac{\varphi_j - \varphi_{j-1}}{\delta_{j-1}} \right]^2 \tag{59a}$$

$$\text{subject to} \; \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} - \frac{h_\mathrm{i}\sqrt{\varphi_n} + \dot{h}_\mathrm{i}}{g} = 0 \tag{59b}$$

$$\omega_\mathrm{i,min}^2 \leq \varphi_n \leq \omega_\mathrm{i,max}^2 \tag{59c}$$

$$\forall j < n, \; \lambda_\mathrm{min}\delta_j \leq \varphi_{j+1} - \varphi_j \leq \lambda_\mathrm{max}\delta_j \tag{59d}$$

$$\varphi_1 = \delta_0 g / h_\mathrm{f} \tag{59e}$$

---

c) Boundedness (iii): the variables φ_j can also be used to express the integral as a finite sum:

$$\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \sum_{j=0}^{n-1} \int_{s_j}^{s_{j+1}} \frac{s\,\mathrm{d}s}{\sqrt{\varphi_j + \lambda_j(s^2 - s_j^2)}} \tag{55}$$

$$= \sum_{j=0}^{n-1} \int_0^{\delta_j} \frac{\mathrm{d}v}{2\sqrt{\varphi_j + \lambda_j v}} \tag{56}$$

$$= \sum_{j=0}^{n-1} \frac{1}{\lambda_j} \left[\sqrt{\varphi_j + \lambda_j \delta_j} - \sqrt{\varphi_j} \right] \tag{57}$$

$$= \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} \tag{58}$$

To complete the optimization problem, we add a regularizing cost function over variations of λ so that the ideal behavior becomes a constant $\lambda = \omega_\mathrm{LIP}^2$, *i.e.* the LIP model. This way, height variations are only added when required. This behavior is *e.g.* depicted in Figure 4, where linear capture trajectories are used until the CoP reaches the boundary of the support area and height variations are used for additional braking.

The complete optimization problem is assembled in Equation (59). We will refer to it as the *capture problem*.

D. Computation and behavior of CoM capture trajectories

Suppose that we know the solution $\boldsymbol{\varphi}^*$ to the capture problem, *e.g.* computed using the solver from Section IV. The first step to return from s to time trajectories is to calculate the times $t_j = t(s_j)$ where the stiffness λ switches from one value to the next. With this definition, $s_n = 1$ corresponds to $t_n = 0$ and the partition $1 = s_n > \ldots > s_0 = 0$ corresponds to the time partition $0 = t_n < t_{n-1} < \ldots < t_1 < \infty$.

Recall how the piecewise-constant values of λ are readily computed from $\boldsymbol{\varphi}$ via Equation (51). On an interval $[t_{j+1}, t_j]$ where $\lambda(t) = \lambda_j$ is constant, we can solve the differential equation $\ddot{s} = \lambda_j s$ to obtain:

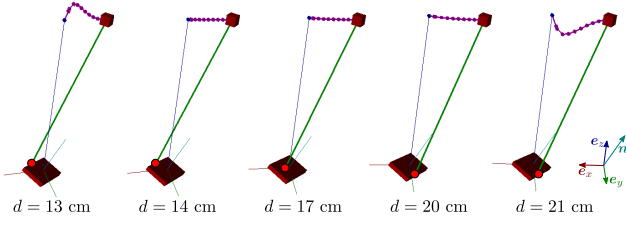$$s(t) = s_{j+1} \left[\cosh(x_j(t)) - \frac{\omega(s_{j+1})}{\sqrt{\lambda_j}} \sinh(x_j(t)) \right] \tag{60}$$

Fig. 4. Zero-step capture trajectories for different contact locations. The CoM starts from the red box with a given initial velocity and needs to come to a stop at the blue line located at distance d. Red discs indicate where the successful capture trajectory (magenta) puts its initial CoP $\boldsymbol{r}_i$. This CoP will then vary on a line segment to its final position $\boldsymbol{r}_f$ at the base of the blue line. The LIP is able to stop over a range $d \in [14, 20]$ cm using only CoP variations, which corresponds to the ankle strategy. The VHIP is able to stop over a wider range thanks to the addition of the height-variation strategy.

where $x_j(t) = \sqrt{\lambda_j}(t - t_{j+1})$. Solving for the boundary condition $s(t_j) = s_j$ yields the next time $t_j = t(s_j)$:

$$t(s_j) = t_{j+1} + \frac{1}{\sqrt{\lambda_j}} \log\left(\frac{\sqrt{\varphi_{j+1}} + \sqrt{\lambda_j} s_{j+1}}{\sqrt{\varphi_j} + \sqrt{\lambda_j} s_j} \right) \quad (61)$$

By backward recursion, we can thus compute the time partition $0 = t_n < t_{n-1} < \ldots < t_1 < \infty$. The set of values (λ_j, t_j) characterizes $\lambda(t)$ as a function of time.

Given both $s(t)$ from Equation (60) and $\omega(s)$ from Equation (51), we can compute the CoP trajectory:

$$\boldsymbol{r}(t) = \boldsymbol{r}_f + (\boldsymbol{r}_i - \boldsymbol{r}_f) \left(\frac{s(t)\omega(s(t))}{\sqrt{\varphi_n}} \right)^{\frac{\alpha}{1-\alpha}} \quad (62)$$

Finally, once both components $\lambda(t)$ and $\boldsymbol{r}(t)$ of the capture input are known, we can compute the CoM capture trajectory by forward integration of the VHIP equation of motion (5). Figure 4 shows various such trajectories for zero-step capture scenarios with the same initial state $\boldsymbol{x}_i$ but different contact locations.

IV. OPTIMIZATION OF CAPTURE PROBLEMS

While the capture problem (59) belongs to the general class of nonconvex optimization, we can leverage its structural properties to solve it much faster than a generic nonconvex problem. This Section presents a dedicated solver that can solve capture problems within tens of *microseconds*. Readers more interested in walking pattern generation can skip directly to Section V.

In what follows, we assume that the reader is already familiar with common knowledge in numerical optimization, including the active-set method for quadratic programming (QP) and the sequential quadratic programming (SQP) method for nonlinear optimization. An overview of this background is provided for reference in Appendix B.

A. Problem reformulation

The capture problem (59) has a linear least squares cost, linear constraints and *a single one-dimensional nonlinear*

equality constraint. Its objective (59a) can be written $\|\mathbf{J}\boldsymbol{\varphi}\|^2$ where the cost matrix $\mathbf{J}$ is the $(n-1) \times n$ matrix given by:

$$\mathbf{J} = \begin{bmatrix} -d_0 - d_1 & d_1 & & & \\ d_1 & -d_1 - d_2 & d_2 & & \\ & & \ddots & & \\ & & & d_{n-2} & -d_{n-2} - d_{n-1} & d_{n-1} \end{bmatrix}$$

with $d_j = \delta_j^{-1}$. Equation (59b) rewrites to $b(\boldsymbol{\varphi}) = 0$ with:

$$b(\boldsymbol{\varphi}) = \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} - \frac{h_i \sqrt{\varphi_n} + \dot{h}_i}{g} \quad (63)$$

where $h_i > 0$ and $\dot{h}_i \in \mathbb{R}$ are problem parameters, *i.e.* constant during the optimization. Linear constraints (59c)–(59e) have the form:

$$\boldsymbol{l} \leq \mathbf{C}\boldsymbol{\varphi} \leq \boldsymbol{u} \quad (64)$$

where the two vectors $\boldsymbol{l} \in \mathbb{R}^{n+1}$ and $\boldsymbol{u} \in \mathbb{R}^{n+1}$ are also problem parameters, and $\mathbf{C}$ is the $(n+1) \times n$ matrix:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_Z \\ \boldsymbol{e}_n^T \end{bmatrix} \text{ where } \mathbf{C}_Z = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}$$

with $\boldsymbol{e}_n$ is the last column of the $n \times n$ identity matrix. Equality constraints are specified by letting $l_j = u_j$.

Solutions to the capture problem (59) can be approximated by solving:

$$\underset{\boldsymbol{\varphi} \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2} \|\mathbf{J}\boldsymbol{\varphi}\|^2 + \frac{\mu^2}{2} \|b(\boldsymbol{\varphi})\|^2 \quad (65a)$$

$$\text{subject to } \boldsymbol{l} \leq \mathbf{C}\boldsymbol{\varphi} \leq \boldsymbol{u} \quad (65b)$$

which presents the advantage of having only linear constraints, and whose solution tends to the original solution as μ goes to infinity. This reformulation is reminiscent of penalty-based methods, where the penalty parameter μ would be adapted during successive iterations. Yet, a fixed parameter (typically $\mu = 10^6$) is enough to get a precise solution in practice, so that we do not need to adapt this parameter.

B. Application of an SQP approach

We apply the SQP method to the reformulation (65). Let us denote by $f(\boldsymbol{\varphi})$ the objective (65a) of the problem and $\boldsymbol{j} := \nabla_{\boldsymbol{\varphi}} b$ the gradient of the nonlinear constraint. The Lagrangian of problem (65) is

$$\mathcal{L}(\boldsymbol{\varphi}, \boldsymbol{\lambda}_-, \boldsymbol{\lambda}_+) = f(\boldsymbol{\varphi}) + \boldsymbol{\lambda}_-^T(\boldsymbol{l} - \mathbf{C}\boldsymbol{\varphi}) + \boldsymbol{\lambda}_+^T(\mathbf{C}\boldsymbol{\varphi} - \boldsymbol{u})$$

with $\boldsymbol{\lambda}_-, \boldsymbol{\lambda}_+ \in \mathbb{R}^{n+1}$ the corresponding Lagrange multipliers.

Let us index by $\square_k$ the value of any quantity at iteration k of the SQP method. The Hessian matrix at iteration k is then:

$$(\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \mathcal{L})_k = \mathbf{J}^T\mathbf{J} + \mu^2 \boldsymbol{j}_k \boldsymbol{j}_k^T + \mu^2 b_k (\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 b)_k \quad (66)$$

We adopt the Gauss-Newton approximation $\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \mathcal{L}_k \approx \mathbf{J}^T\mathbf{J} + \mu^2 \boldsymbol{j}_k \boldsymbol{j}_k^T$, a classical approach for nonlinear least squares. It is particularly well-suited to the present case, as $\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \mathcal{L}$ is exactly equal to $\mathbf{J}^T\mathbf{J} + \mu^2 \boldsymbol{j}\boldsymbol{j}^T$ when the boundedness condition

$b = 0$ is satisfied. Under this approximation, the problem for one iteration of the SQP method becomes:

$$\underset{\boldsymbol{p}\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\left\|\mathbf{J}\boldsymbol{p} + \mathbf{J}\boldsymbol{\varphi}_k\right\|^2 + \frac{\mu^2}{2}\left\|\boldsymbol{j}_k^T\boldsymbol{p} + b_k\right\|^2 \quad (67\text{a})$$

$$\text{subject to } \boldsymbol{l}_k' \le \mathbf{C}\boldsymbol{p} \le \boldsymbol{u}_k' \quad (67\text{b})$$

where $\boldsymbol{l}_k' := \boldsymbol{l} - \mathbf{C}\boldsymbol{\varphi}_k$ and $\boldsymbol{u}_k' := \boldsymbol{u} - \mathbf{C}\boldsymbol{\varphi}_k$. If $\boldsymbol{\varphi}_k$ satisfies the linear constraints (65b), then $\boldsymbol{p} = 0$ is a feasible point for the problem (67). This problem is a linear least squares with inequality constraints (LSI), a particular case of QP, that we can solve using the active-set method (Algorithm 1). Adopting $\boldsymbol{d}$ and j for the step and iteration number of the QP (keeping $\boldsymbol{p}$ and k for the SQP), an iteration of the active-set method solves in this case:

$$\underset{\boldsymbol{d}\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\left\|\mathbf{J}(\boldsymbol{d} + \boldsymbol{p}_j + \boldsymbol{\varphi}_k)\right\|^2 + \frac{\mu^2}{2}\left\|\boldsymbol{j}_k^T(\boldsymbol{d} + \boldsymbol{p}_j) + b_k\right\|^2 \quad (68\text{a})$$

$$\text{subject to } \mathbf{C}_{\mathcal{W}_j}\boldsymbol{d} = 0 \quad (68\text{b})$$

with $\mathcal{W}_j$ the set of active constraints at the current iteration j and $\mathbf{C}_{\mathcal{W}_j}$ the corresponding matrix.

This can be solved in two steps using the nullspace approach. First, compute a matrix $\mathbf{N}_{\mathcal{W}} \in \mathbb{R}^{n\times n-r}$ whose columns form a basis of the nullspace of $\mathbf{C}_{\mathcal{W}}$, r being the rank of $\mathbf{C}_{\mathcal{W}}$. The vector $\boldsymbol{d}$ is then solution of the problem if and only if $\boldsymbol{d} = \mathbf{N}_{\mathcal{W}}\boldsymbol{z}$ for some $\boldsymbol{z} \in \mathbb{R}^{n-r}$. The problem can thus be rewritten as an unconstrained least squares:

$$\underset{\boldsymbol{z}\in\mathbb{R}^{n-r}}{\text{minimize}} \quad \frac{1}{2}\left\|\begin{bmatrix}\mu\boldsymbol{j}^T\\\mathbf{J}\end{bmatrix}\mathbf{N}_{\mathcal{W}}\boldsymbol{z} + \begin{bmatrix}\mu(\boldsymbol{j}^T\boldsymbol{p}_j + f)\\\mathbf{J}(\boldsymbol{p}_j + \boldsymbol{\varphi})\end{bmatrix}\right\|^2 \quad (69)$$

Second, solve this unconstrained problem: taking $\mathbf{T}$ and $\boldsymbol{u}$ such that the above objective writes $\frac{1}{2}\|\mathbf{T}\boldsymbol{z} + \boldsymbol{u}\|^2$, compute the QR decomposition $\mathbf{T} = \mathbf{QR}$, and solve $\mathbf{QR}\boldsymbol{z} = -\boldsymbol{u}$. The latter is equivalent to $\boldsymbol{z} = -\mathbf{R}^{-1}\mathbf{Q}^T\boldsymbol{u}$ if $\mathbf{R}$ has full rank [28, Chapter 10]. Both of these steps can be significantly tailored to the case of capture problems.

C. Tailored operations

In the SQP, most of the time is spent in solving the underlying LSI: the computation of $\mathbf{N}_{\mathcal{W}}$, the post-multiplication by $\mathbf{N}_{\mathcal{W}}$ to obtain $\mathbf{T}$, the QR decomposition of $\mathbf{T}$ and the computation of the Lagrange multipliers are the main operations, performed each roughly in $O(n^3)$ [29], at least for the first iteration of each LSI.[6] We can reduce this complexity to at most $O(n^2)$ for capture problems:

- The matrix $\mathbf{N}_{\mathcal{W}}$ does not need to be computed explicitly. Rather, the cost matrix $\mathbf{T}$ of the unconstrained least squares problem can be built directly in $O(n)$ operations by taking advantage of the structure of $\mathbf{C}_{\mathcal{W}}$ and $\mathbf{J}$ (Appendix B-D)
- Lagrange multipliers needed to check KKT conditions can be computed in $O(n^2)$ by taking advantage of the structure of $\mathbf{C}_{\mathcal{W}}$ (Appendix B-E)

- The QR decomposition of $\mathbf{T}$ can be carried out in $O(n^2)$ by leveraging the tridiagonal structure of $\mathbf{J}_{\mathcal{W}}$ (Appendix B-F)

The last point to consider is finding an initial pair $(\boldsymbol{\varphi}_0, \boldsymbol{\lambda}_0)$ for the SQP. While in a classical SQP this is done through a so-called *Phase I* which can be almost as costly as running the main loop of the algorithm itself, we can leverage the geometry of our constraints to get such a pair in $O(n)$ (Appendix B-G).

D. Numerical and algorithmic considerations

The implementation of a general-purpose QP or SQP solver is an extensive work due to the numerous numerical difficulties that can arise in practice: active-set methods need to perform a careful selection of their active constraints in order to keep the corresponding matrix well conditioned, while SQPs require several refinements, some of which imply solving additional QPs at each iteration [28]. While the tailored operations we presented reduce the theoretical complexity w.r.t. general-purpose solvers, there are also a number of features of Problem (65) that allow us to stick with a simple, textbook implementation, and contribute to the general speed-up.

On the QP side, the matrix $\mathbf{C}_Z$ is always full rank and well conditioned, while the last row $\boldsymbol{e}_n^T$ of $\mathbf{C}$ is a linear combination of *all* rows from $\mathbf{C}_Z$ ($\boldsymbol{e}_n = \mathbf{C}_Z^T\mathbf{1}_n$). As a consequence, all matrices $\mathbf{C}_{\mathcal{W}}$ are full rank and well conditioned, save for the case where all $n+1$ constraints are active. This case can be easily detected and avoided.[7] It is thus safe to use a basic active-set scheme.

All QR decompositions are performed on matrices with rank deficiency of at most one. As a consequence, it is not necessary to use more involved column-pivoting algorithms, and the rank deficiency can be detected by simply monitoring the bottom-right element of the triangular factor. While we don't prove that the matrices $\mathbf{J}_{\mathcal{W}}$ are well conditioned, we verified this assertion for $n \le 20$ in a systematic way. Even for large values of μ, the QR decomposition of $\mathbf{T}$ is stable as the row with largest norm appears first [30, p. 169].

On the SQP side, the odds are very favorable: constraints are linear and the Gauss-Newton approach offers a good approximation of the Hessian matrix. As a consequence, we observed that the method takes full steps 98.5% of the time in practice, and converges in very few iterations (four on average). Since we start from a feasible point, all subsequent iterates are guaranteed to be feasible and the line search needs only monitor the objective function, in an unconstrained-optimization fashion.

E. Pre-computation of QR decompositions

It is important to note that the matrices $\mathbf{J}$ and $\mathbf{C}$ only depend on the problem size n and partition $s_0, \ldots, s_n$, which are the same across all capture problems that we solve in this work. If n is small enough (say $n \le 20$), we can precompute and store the QR decompositions of *all* possible $\mathbf{J}_{\mathcal{W}} := \mathbf{J}\mathbf{N}_{\mathcal{W}}$. Note that there are $2^{n+1} - 1$ different sets $\mathcal{W}$, including all

---

[6] We could refine these estimates by taking into account the number of active constraints. Note also that subsequent LSI iterations can perform some of these operations in $O(n^2)$.

[7] With the notations of Appendix B-G, this can only happen if a^- or a^+ is equal to 0 or 1. A workaround is then to slightly perturb $\omega_{\text{i,min}}$ or $\omega_{\text{i,max}}$.

Solver	$n = 10$	$n = 15$	$n = 20$	$n = 50$
IPOPT[8]	7.1×10^3	9.4×10^3	1.1×10^4	2.2×10^4
SQP + LSSOL	86 ± 60	130 ± 86	220 ± 160	1700 ± 1700
SQP + cLS	22 ± 12	33 ± 18	54 ± 41	$\mathbf{210 \pm 180}$
SQP + cLS + pre.	$\mathbf{18 \pm 10}$	$\mathbf{25 \pm 14}$	$\mathbf{35 \pm 22}$	–

combinations of up to n active constraints among $n+1$. These pre-computations can be done in a reasonable amount of time thanks to our specialized QR decompositions, and result in even faster resolution times.

F. Performance comparison

In [11], Problem (59) was solved with the state-of-the-art solver IPOPT [31], which is written in Fortran and C. We compare its performances with our tailored SQP approach, implemented in C++.[9] Taking $\mu = 10^6$, the solutions returned by both methods are numerically equivalent (within 10^{-7} of one another, and $|b(\boldsymbol{\varphi})| \approx 10^{-8}$ in both cases). Computation times over representative problems produced during walking pattern generation are reported in Table I, where our approach is denoted *SQP + cLS* (custom least squares), and the abbreviation *pre.* denotes the use of QR pre-computations for $\mathbf{J}_\mathcal{W}$. In practice, we work with $n = 10$, for which our solver is 300–400 times faster than the generic solver IPOPT.

Computation times for QR pre-computations range from 2 ms for $n = 10$, 100 ms for $n = 15$, to 4.9 s for $n = 20$. This is not limiting in practice, as these computations are performed only once at startup. The limit rather lies with memory consumption, which follows an exponential law ranging from 2 MB for $n = 10$ to 5 GB for $n = 20$.

To break down how much of the speed-up is due to the problem reformulation and how much is due to our custom least squares implementation, we also test our SQP method using the state-of-the-art least squares solver LSSOL [32]. This variant is denoted by *SQP + LSSOL*. For $n \geq 25$, SQP + LSSOL starts to fail on some problems, with a failure rate of roughly 25% for $n = 50$. This suggests that the least squares component of our solver is more robust. There are two plausible explanations for this. First, LSSOL assumes all matrices are dense, while we leverage sparsity patterns of $\mathbf{J}$ and $\mathbf{C}$. Second, LSSOL treats all coefficients as floating-point numbers, while knowing that the elements of $\mathbf{C}$ are exactly 1 or -1 allows us to carry out exact computations (most notably for the nullspace and pseudoinverse of $\mathbf{C}_\mathcal{W}$).

V. WALKING PATTERN GENERATION

While zero-step capturability enables push recovery, *one-step* capturability is the minimum price to pay for walking.

We will now see how this next level of capturability can be solved with a pipeline similar to the balance control one:

- Change variable from time t to a new variable s
- Define the CoP $\boldsymbol{r}(s)$ as piecewise-constant function of s
- Parameterize its contact-switch time by a new variable α
- Reduce capturability to an optimization over $\varphi(s, \alpha)$
- Compute the optimal solution $\varphi^*(s, \alpha)$ of this problem
- Change variable from s to time t

From $\varphi^*(t)$, it is then straightforward to compute the full capture input $t \mapsto \lambda^*(t), \boldsymbol{r}^*(t)$ as well as the capture trajectory $\boldsymbol{c}^*(t)$. Let us now detail each step of this pipeline.

A. Time-varying CoP strategy

In a one-step capture setting, the contact area switches instantaneously at a given instant $s_\mathrm{c} \in (0, 1)$:

$$\mathcal{C}(s) = \begin{cases} \mathcal{C}_\mathrm{i} & \text{for } s_\mathrm{c} < s \leq 1 \\ \mathcal{C}_\mathrm{f} & \text{for } 0 < s \leq s_\mathrm{c} \end{cases} \tag{70}$$

Due to this discontinuity, we cannot adopt the same line-segment CoP strategy that we used in Section III for balance control. Let us then adopt a piecewise-constant CoP trajectory:

$$\boldsymbol{r}(s) = \begin{cases} \boldsymbol{r}_\mathrm{i} & \text{for } s_\mathrm{c} < s \leq 1 \\ \boldsymbol{r}_\mathrm{f} & \text{for } 0 < s \leq s_\mathrm{c} \end{cases} \tag{71}$$

This choice yields the following boundedness condition (31):

$$-\boldsymbol{g} \int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \omega_\mathrm{i}(\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\mathrm{i}) + (\boldsymbol{r}_\mathrm{i} - \boldsymbol{r}_\mathrm{f})s_\mathrm{c}\omega(s_\mathrm{c}) + \dot{\boldsymbol{c}}_\mathrm{i} \tag{72}$$

In terms of our optimization variable $\boldsymbol{\varphi}$, the right-hand side of this equation is:

$$(\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\mathrm{i})\sqrt{\varphi_n} + (\boldsymbol{r}_\mathrm{i} - \boldsymbol{r}_\mathrm{f})\sqrt{\varphi(s_\mathrm{c})} + \dot{\boldsymbol{c}}_\mathrm{i} \tag{73}$$

where $\varphi(s_\mathrm{c})$ is a linear combination of φ_j and φ_{j+1} when $s_\mathrm{c} \in [s_j, s_{j+1}]$ from Equation (50).

Let us define an external parameter $\alpha \in (0, 1)$. This time, we parameterize the contact switch s_c of the CoP trajectory by α as follows:

$$s_\mathrm{c} \text{ is the scalar s.t. } \sqrt{\varphi(s_\mathrm{c})} = \alpha\sqrt{\varphi_n} \tag{74}$$

Under this assumption, the right-hand side of the boundedness condition then simplifies to:

$$(\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\alpha)\sqrt{\varphi_n} + \dot{\boldsymbol{c}}_\mathrm{i} \tag{75}$$

where $\boldsymbol{r}_\alpha := \alpha\boldsymbol{r}_\mathrm{f} + (1 - \alpha)\boldsymbol{r}_\mathrm{i}$ is the *equivalent constant CoP* [2] of the time-varying CoP trajectory. In the LIP with constant ω_LIP, Equation (74) rewrites to $\alpha = e^{-\omega_\mathrm{LIP}t_\mathrm{c}}$,[10] which shows how α parameterizes the time of contact switch. This observation still holds for the VHIP, where we will use α as an external variable to optimize over contact switch times.

Taking the dot product of this equation with the normal $\boldsymbol{n}_\mathrm{i}$ of the initial contact $\mathcal{C}_\mathrm{i}$ yields:

$$\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \frac{h_\alpha\sqrt{\varphi_n} + \dot{h}_\mathrm{i}}{g} \qquad h_\alpha := \frac{(\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\alpha) \cdot \boldsymbol{n}_\mathrm{i}}{(\boldsymbol{e}_z \cdot \boldsymbol{n}_\mathrm{i})} \tag{76}$$

---

[8] We only report averages for IPOPT computation times as they lie on a different scale. These averages are higher than those reported in [11] because we evaluate both feasible and unfeasible problems (for reasons made clear in the next section), while all random initial conditions in [11] were zero-step capturable. For $n = 10$ and projecting performance statistics on feasible problems only, IPOPT's computation times decrease to 1600 ± 790 μs.

[9] https://github.com/jrl-umi3218/CaptureProblemSolver

[10] This expression corresponds to the limit of the scalar w' when $\Delta t'_1 \to \infty$ in Equation (25) of [2]. It also plays an important role in the step timing adjustment method from [33].

This constraint is exactly of the form (59b) of the capture problem (59), where the parameter h_i has simply been replaced by h_α.

The horizontal components of Equation (75) give:

$$\boldsymbol{r}_i^{xy} = \boldsymbol{r}_f^{xy} + \frac{1}{1-\alpha}\left[\boldsymbol{c}_i^{xy} + \frac{\dot{\boldsymbol{c}}_i^{xy}}{\omega_i} - \boldsymbol{r}_f^{xy}\right] \qquad (77)$$

This expression is identical to Equation (37). Following the same steps (38)–(46) as in the zero-step setting, we therefore conclude that capture inputs under the CoP strategy (71)–(74) are characterized by the 1D capturability condition (Property 4). In particular, we can compute them using our capture problem solver.

B. External optimization over contact switching time

The contact switching time $t_c(\alpha)$ plays a crucial role in one-step capture trajectories. Given the solution $\boldsymbol{\varphi}_\alpha$ to the capture problem parameterized by α, we can compute it as follows. First, compute s_c using Equation (51):

$$s_c = \sqrt{s_j^2 + \frac{\alpha^2 \varphi_{\alpha,n} - \varphi_{\alpha,j}}{\lambda_{\alpha,j}}} \qquad (78)$$

where $\varphi_\alpha(s_c) = \alpha^2 \varphi_{\alpha,n} \in [\varphi_{\alpha,j}, \varphi_{\alpha,j+1}]$. Then, apply the mapping $t_c = t(s_c)$ provided by Equation (61).

While this mapping is straightforward to compute numerically, its high nonlinearity suggests that introducing time constraints into capture problems would radically affect the design of the dedicated solver from Section IV. Fortunately, computation times achieved by this solver allow us to solve *hundreds* of capture problems per control cycle. We therefore choose to optimize jointly over $\boldsymbol{\varphi}$ and α using a two-level decomposition: an external optimization over $\alpha \in (0,1)$, wrapping an internal optimization where α is fixed and the existing capture problem solver is called to compute $\boldsymbol{\varphi}_\alpha$.

A straightforward way to carry out the external optimization is to test for a large number of values $\alpha \in (0,1)$. This approach is however inefficient in practice as it sends a significant amount of unfeasible problems to the internal optimization. The main cause for such unfeasibility is that whole intervals of values for α result in $\omega_{i,min}(\alpha) > \omega_{i,max}(\alpha)$, for which the CoP constraint (59c) is obviously unfeasible. To avoid this, we propose in Appendix C-A an algorithm to pre-compute the set of intervals $[\alpha_{min}, \alpha_{max}] \subset (0,1)$ on which it is guaranteed that $\omega_{i,min} \leq \omega_{i,max}$.

During bipedal walking, contact switches can only be realized after the free foot has completed its swing trajectory from a previous to a new contact. We therefore need to make sure that $t_c(\alpha)$ is greater than the remaining duration t_{swing} of the swing foot motion. This gives us an additional constraint to be enforced by the external optimization:

$$t_c(\alpha) \geq t_{swing} \qquad (79)$$

where each evaluation of $t_c(\alpha)$ costs the resolution of a full capture problem. We find such solutions by sampling n_α values of α per feasible interval $[\alpha_{min}, \alpha_{max}]$. This approach provides automatic step timings, and tends to output pretty dynamic gaits.

Alternatively, when step timings are provided one may want to enforce an equality constraint:

$$t_c(\alpha) = t_{swing} \qquad (80)$$

In this case, the external optimization implements an approximate gradient descent, calling the capture problem solver multiple times to evaluate $\nabla_\alpha t_c$ and converge to a local optimum on each feasible interval $[\alpha_{min}, \alpha_{max}]$. Although we don't prove it formally, we observed in practice that the mapping $t_c(\alpha)$ seems to be always monotonic (as in the LIP) and the optimum thus found global on its interval.

C. Pattern generation from capture trajectories

We can generate walking from capture trajectories via a two-state strategy [19], [34] combining zero-step and one-step capture problems. The robot starts in a double-support posture, and computes a one-step capture trajectory to transfer its CoM to its first support foot.

- **Single support phase:** swing foot tracks its pre-defined trajectory while the CoM follows a one-step capture trajectory updated in a model predictive control fashion. The phase ends at swing foot touchdown.
- **Double support phase:** the CoM follows a zero-step capture trajectory towards the target contact. Meanwhile, the capture problem solver is called to compute a one-step capture trajectory towards the next contact. The phase ends as soon as such a trajectory is found.

The behavior realized by this state machine is conservative: after touchdown, the robot uses its double support phase to slow down until a next one-step capture trajectory is found. If the next contact is not one-step capturable, the robot stops walking and balances in place. Otherwise, walking continues toward the next pair of footsteps.

A limitation of this walking pattern generator is that double-support phases do not include a continuous CoP transfer from one contact to the next. The one exception to this is when both contacts are coplanar, in which case we can use their convex hull as halfspace representation $(\mathbf{H}, \boldsymbol{p})$ in Equation (43) and thus apply the line-segment CoP strategy from Section III-B. In practice, we let CoP discontinuities occur in the walking pattern during double support and rely on the underlying robot stabilizer to handle them (see Section VI-B).

VI. SIMULATIONS

We validated this walking pattern generator in two successive implementations. First, we evaluate the kinematics of whole-body tracking of the VHIP reference in a prototyping environment. Then, we implement our method as a full-fledged C++ controller and evaluate its performance compared to the state of the art in two dynamic simulators.

A. Kinematics

We implemented our walking pattern generation method in the *pymanoid*[11] prototyping environment. Whole-body inverse kinematics is carried out using a standard quadratic-programming formulation (see *e.g.* [35, Section 1] for a

---

[11] https://github.com/stephane-caron/pymanoid

(a) Elliptic staircase   (b) Regular staircase   (c) Aircraft factory
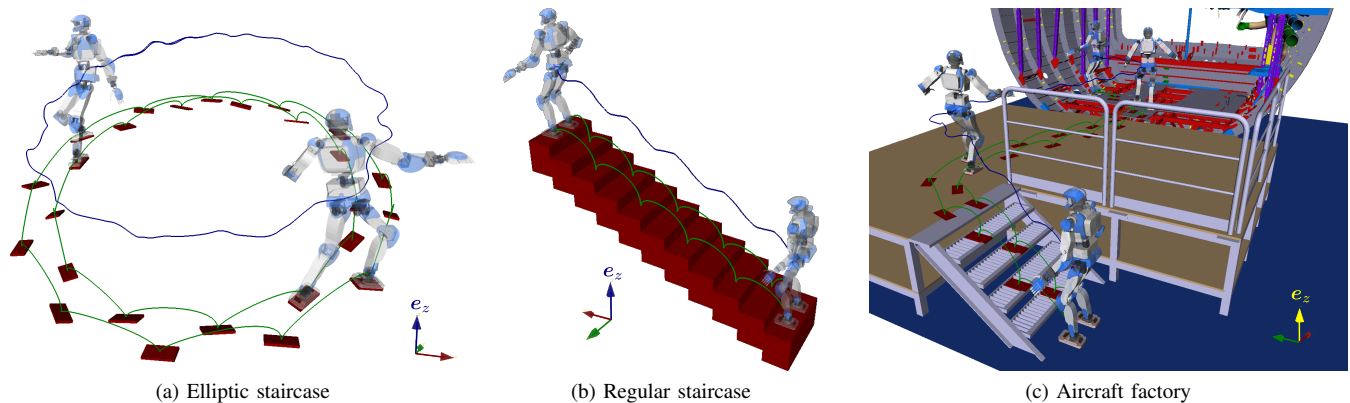
Fig. 5. **Walking pattern generation:** whole-body tracking in three scenarios. The elliptic staircase with randomly-tilted footsteps (a) tests the ability to walk over rough terrains, *i.e.* to adapt to both 3D translation and 3D orientation variations between contacts. The regular staircase (b) has 15-cm high steps; it assesses the behavior of the solution when contacts are close to each other and collision avoidance becomes more stringent. Finally, the aircraft scenario (c) provides a real-life use case where the environment combines flat floors, staircases with 18.5-cm high steps and uneven-ground areas (inside the fuselage). In all three figures, blue and green trajectories respectively correspond to center-of-mass and swing-foot trajectories.

survey). Tracking of the inverted pendulum and swing foot trajectories is realized by the following set of tasks:

Task group	Task	Weight
Foot tracking	Support foot	1
Foot tracking	Swing foot	10^{-3}
VHIP tracking	Center of mass	1×10^{-2}
VHIP tracking	Min. ang. mom. variations	1×10^{-4}
Regularization	Keep upright chest	1×10^{-4}
Regularization	Min. shoulder extension	1×10^{-5}
Regularization	Min. upper-body velocity	5×10^{-6}
Regularization	Reference upright posture	1×10^{-6}

We considered the three scenarios depicted in Figure 5. The first one, an elliptic staircase with randomly-tilted footsteps (Figure 5a), tests the ability to adapt to general uneven terrains. It illustrates the main advantage of the capture problem formulation: it provides real-time model predictive control with (CoP and unilaterality) constraint saturation over rough terrains. Existing methods either enforce constraints on regular terrains such as floors or stairs [36], [37], or do not take constraint saturation into account [17], [6], or compute costly nonconvex optimizations offline [38], [39] (see Section VII for a complete discussion).

We also consider a regular staircase with 15-cm high steps (Figure 5b), and the real-life scenario provided by Airbus Group depicted in Figure 5c. It consists of a 1:1 scale model of an A350 aircraft under construction in a factory environment. To reach its desired workspace configuration, the humanoid has to walk up an industrial-grade staircase (step height 18.5 cm, except the last one which is 14.5 cm), then across a flat floor area and finally inside the fuselage where the ground consists of temporary wooden slabs. All three walking patterns are depicted in Figure 5 and in the accompanying video.

We use the same set of parameters on all scenarios. For capture problems, we choose $n = 10$ discretization steps with a partition $s_i = i/n$ and set $h_{\mathrm{f}} = 0.8$ m, which is a suitable CoM height for HRP-4 at rest with an extended leg. The external optimization for one-step problems samples $n_\alpha = 5$ values of α per feasibility interval, which is sufficient

to find solutions that satisfy the inequality constraint (79) on contact switching time. For the VHIP stiffness feasibility, we set $\lambda_{\min} = 0.1g$ and $\lambda_{\max} = 2g$.

At each control cycle, the capture problem solver is called on both zero-step and one-step capture problems. On a consumer laptop computer, zero-step problems were solved in 0.38 ± 0.13 ms while one-step ones were solved in 2.4 ± 1.1 ms (average and standard deviations over 10,000 control cycles). Note that these computation times reflect both calls to the C++ solver (Section IV) and the external optimization over α implemented in the prototyping environment in Python.

We release the code of this prototype as open source software.[12]

B. Dynamics

To test our method in dynamic simulations and compare it to the existing, we extend the C++ LIP-based stair climbing controller from [40]. This controller consists of two main components: a LIP-based pattern generator by model predictive control (*i.e.* the pattern is produced online in receding horizon) and a stabilizer based on whole-body admittance control. We implement our solution as a second pattern generator in this controller, using the same stabilizer in both cases for comparison.

The planar CoM constraint of the LIP is applicable to small step heights (*e.g.* Kajita *et al.* applied it to 10-cm high steps in [41]) but can become problematic for higher steps. Another option is to constrain the CoM to a non-horizontal inclined plane parallel to the staircase, yet in this case either the CoP is kept constant [42] or the CoP constraint becomes nonlinear and harder to enforce [13]. From experiments on the HRP-4 robot, [40] chose to decompose the CoM constraint in two horizontal plane segments (thus keeping linear CoP inequalities), introducing the vertical height variation at toe liftoff. These segments could be further decomposed for better adaption to terrain, but as of today there is no known algorithm

---

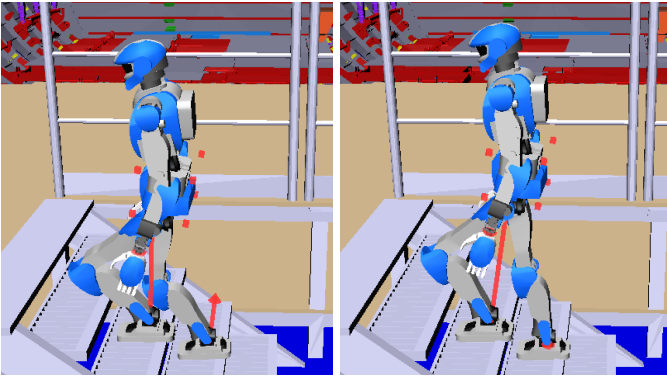[12] https://github.com/stephane-caron/capture-walkgen

Fig. 6. **Dynamic simulations in Choreonoid:** snapshots at the maximum knee flexion time instant. *Left:* LIP-based pattern generation. *Right:* VHIP-based pattern generation. The former causes HRP-4 to bend its knees more, increasing the risk of shank collision with the staircase. We can also observe the more abrupt CoP transfer of the latter, which is specific to our method.
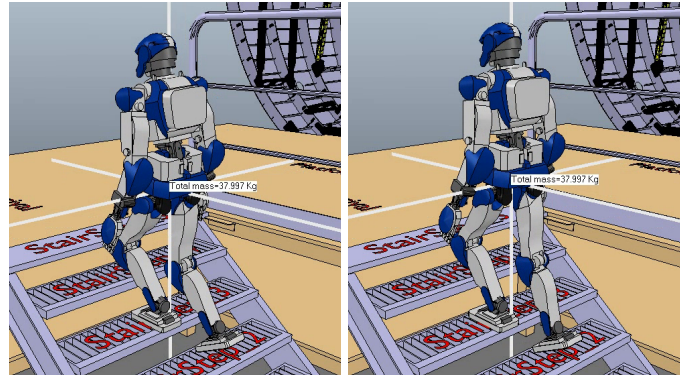

Fig. 7. **Dynamic simulations in V-REP:** snapshots at the toe liftoff time instant. Simulations use the Newton dynamics engine with a time step of 5 ms and one pass per frame. *Left:* LIP-based pattern generation. *Right:* VHIP-based pattern generation. The former causes HRP-4 to bend its knees more, raising the risk of shank collision with the staircase.

for this. Using a VHIP-based pattern generator addresses the question by removing the specification of plane segments altogether.

Thanks to the mc_rtc control framework,[13] we validated the feasibility of our method in two different dynamic simulators:

- **Choreonoid**[14], the most reliable one at our disposal, which implements the projected Gauss-Seidel method [43] and runs in real time.
- **V-REP**[15] with the Newton dynamics engine. For a realistic behavior, we set simulations to a relatively small time step of 5 ms, in which case they did not run in real time.

The same set of controller parameters, and thus the same walking patterns, were applied in both simulators. Testing in two simulators allowed us to validate the robustness of our walking controller to varying environment responses: ground contacts in V-REP with Newton are more flexible, with smoothed and oscillatory contact forces, while those of Choreonoid are more rigid and closer to what is observed on the real robot.

Figures 6 and 7 compare LIP-based and VHIP-based controller simulations at different time instants (full simulation recordings are shown in the accompanying video). The main difference between the two lies in knee flexion-extension (Figure 9): thanks to its reference height going up after the first step, the VHIP pattern brings the robot to knee extension while the LIP one bends down then raises again (Figure 8) owing to its choice of a horizontal rather than inclined LIP plane. In the horizontal plane, DCM-ZMP tracking looks similar to a regular capture-point walking controller (Figure 10) since the variations of $\omega(t)$ required to lift the CoM up are very small: in this stair climbing scenario, ω ranges from 3.536 Hz to 3.550 Hz (Figure 11), as opposed *e.g.* to the elliptic staircase scenario (Figure 5a) where ω ranges from 3.4 Hz to 3.8 Hz.

We release the code of this controller as open source software.[16]

VII. DISCUSSION AND FUTURE WORK

While the present study is coming to an end, our understanding of the variable-height inverted pendulum (VHIP) model is, hopefully, only beginning to unfold. What did we understand so far? First, that capturability of the VHIP is characterized by three properties of its two inputs: their feasibility, asymptotic convergence, and the boundedness condition. These properties can be cast into an optimization problem, the *capture problem*, that we can solve in tens of microseconds. These fast solver times allow us to add an external optimization solving for other nonlinearities such as step timings.

Our overall discussion draws numerous connections with the existing literature. To start with, the exponential dichotomy of the time-varying inverted pendulum was proposed in 2004 by J. Hauser *et al.* [20] to address a question of motorcycle balance. Its application to the linear inverted pendulum can be found in the motion generation framework of the Honda ASIMO humanoid [4]. The LIP itself has been the focus of a large part of the recent literature, in the wake of major works such as [41], [36], [4], [2]. Solutions allowing CoM height variations have therefore been the exception more than the rule. They can be grouped into two categories: pre-planning of CoM height functions, and 2D sagittal capturability.

When CoM height variations $c^z(t)$ are pre-planned [44], [45], [17], [18], the remainder of the system can be controlled in the 2D horizontal plane similarly to the LIP, yet with a time-variant rather than time-invariant equation of motion. Two successful LIP solutions have been generalized following this idea: linear model predictive control [36] was extended into [45], and the time-invariant divergent component of motion [6] was extended into a time-variant counterpart [17].[17]

[13] Developed in the course of the COMANOID and JRP projects, this control framework is available at https://gite.lirmm.fr/multi-contact/mc_rtc upon request and will be released soon.
[14] http://choreonoid.org/en/
[15] http://coppeliarobotics.com/

[16] https://github.com/stephane-caron/capture_walking_controller
[17] Both [45] and [17] use polynomial CoM height functions, which makes it easy to satisfy boundary conditions but yields non-integrable dynamics. Terada and Kuniyoshi [44] proposed a symmetric alternative where the system becomes integrable, yet where enforcing boundary conditions is a nonlinear root finding problem.
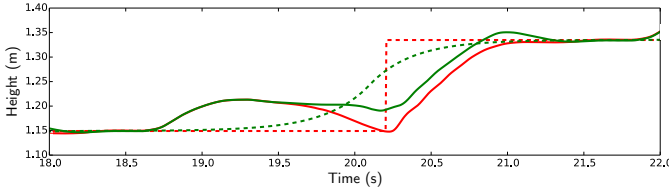
Fig. 8. **CoM height variations** for LIP-based (red) and VHIP-based (green) walking patterns on a representative step. Dashed lines: references from walking patterns. Solid lines: estimates by the controller's CoM observer. Tracking cannot be perfect as walking patterns do not take into account swing foot motions (18.5–20.0 s) and maximum leg extension during double support (20.0–20.2 s) that limit the CoM height kinematically.
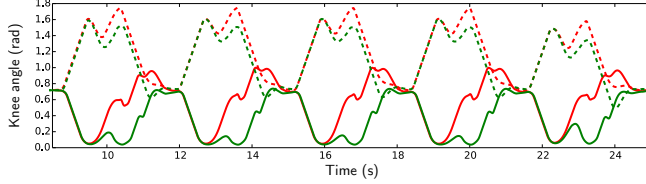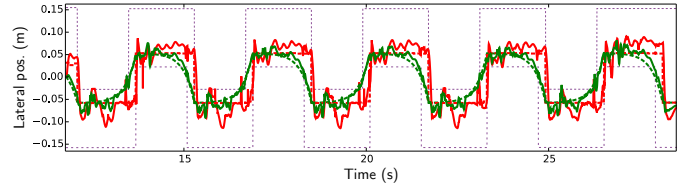

Fig. 10. **Lateral DCM-ZMP tracking for the VHIP** over the complete stair climbing motion. Green: DCM, red: ZMP, dashed: reference, solid: observed. The DCM here is the position $c(t) + \dot{c}(t)/\omega(t)$. References look like a regular capture-point pattern generator as the variations of $\omega(t)$ required to lift the robot up by 18.5 cm are too small to affect the robot's lateral dynamics. The same behavior is observed on sagittal dynamics.
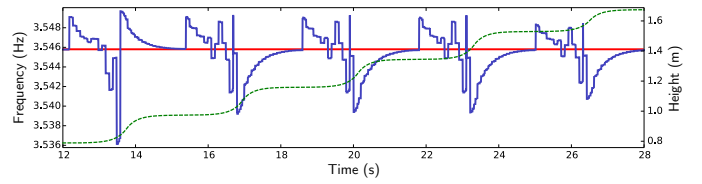

Fig. 9. **Knee angle variations** between LIP-based (red) and VHIP-based (green) whole-body control. Solid lines correspond to the right knee and dashed ones to the left knee. The right knee stays extended longer in the VHIP pattern, curtailing a peak in left-knee flexion observed in the LIP pattern.

Fig. 11. **Time-varying frequency** $\omega(t)$ of the VHIP (blue) compared to its LIP counterpart (red). The VHIP pattern CoM height (Figure 8) is shown for reference in green. The frequency raises when the DCM accelerates upward and falls when the DCM accelerates downward.

Interestingly, in [17] Hopkins *et al.* use the Riccati equation (18) to compute $\omega(t)$ from $c^z(t)$, while in the present study we compute $c(t)$ from $\omega(t)$. More generally, our strategy can be seen as mapping the whole problem onto the damping ω and solving for $\omega(t)$, while the underlying strategy behind those other approaches is to fix ω and map the remainder of the problem onto c^{xy}. Another noteworthy example of the latter can be found in the linearized MPC proposed by Brasseur *et al.* [37], where ω variations are this time abstracted using polyhedral bounds rather than a pre-planned height trajectory.

In this regard, our present study is more akin to works on capturability proposed for the 2D nonlinear inverted pendulum [8], [9], [10]. All of them share a design choice dating back to the seminal work of Pratt and Drakunov [8]: they interpolate CoM trajectories in a 2D vertical plane with a fixed center of pressure (CoP). The key result of [8] is the conservation of the "orbital energy" of a CoM path, a variational principle that we can now interpret as a two-dimensional formulation of the boundedness condition. This principle was later translated into a predictive controller in an equally inspirational study by Koolen *et al.* [10]. Ramos and Hauser [9] also noticed that the capture point, interpreted as *point where to step*, was a function of the CoM path. They proposed a single-shooting method to compute what we would now call 2D capture trajectories.

All of these works hinted at key features of 3D capture trajectories, but applied only to two-dimensional CoM motions in vertical planes. The key to lift this restriction is the 3D boundedness condition, which was first formulated in the case of the LIP by Lanari *et al.* [22] and applied to model predictive control of the LIP in [46]. This condition can be more generally applied to different asymptotic behaviors, including but not restricted to stopping. For instance, infinite stepping is another option [47]. The exploration of these more general asymptotic behaviors is an open question.

Another important choice of the present study is to focus on zero- and one-step capture trajectories. Walking controllers based on one-step capturability have been proposed for both even [48], [33], [34] and uneven terrains [49], [19]. The latter follow a single line of work leading to the present study: [49] finds rough-terrain (even multi-contact) solutions but tends to produce conservatively slow trajectories; [19] discovers dynamic walking patterns, but suffers from numerical instabilities when used in a closed control loop. In our understanding, these instabilities are due to the *direct* transcription of centroidal dynamics, which has proved successful for planning [38], [39] but where closed-loop controllers suffer from frequent switches between local optima [19]. The optimization of capture problems provides an alternative transcription for which we do not observe this numerical sensitivity.

Finally, the last key choice of the present study is the change of variable from t to s. This choice is one possible generalization of the seminal idea by Pratt and Drakunov [8] to make the CoM height a function $c^z(c^x)$ of the 2D CoM abscissa: as it turns out, $c^x(t)$ and $s(t)$ are proportional in their 2D setting [11], although that is not the case any more in 3D. An alternative 3D generalization is to solve for the remaining lateral motion $c^y(t)$ after the sagittal motion has been computed by the 2D method [50]. Both cases, as noted in [50] and in the present study, bear a close connection with time-optimal path parameterization (see *e.g.* [27] for a survey). Future works may explore this connection, and perhaps bring to light computational complexity results regarding the best case performance one can hope for this kind of problems.

The ability to solve capture problems in tens of microseconds opens new perspectives for motion planning and control. For instance, planners can use this tool for fast evaluation of contact reachability, while stepping stabilizers can evaluate

several contact candidates in parallel in reaction to *e.g.* external pushes. These extensions are open to future works.

References

[1] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *IEEE-RAS International Conference on Humanoid Robots*, 2006, pp. 200–207.

[2] T. Koolen, T. de Boer, J. Rebula, A. Goswami, and J. Pratt, "Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models," *The International Journal of Robotics Research*, vol. 31, no. 9, pp. 1094–1113, 2012.

[3] T. Sugihara, "Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 1966–1971.

[4] T. Takenaka, T. Matsumoto, and T. Yoshiike, "Real time motion generation and control for biped robot-1st report: Walking gait pattern generation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1084–1091.

[5] M. Morisawa, S. Kajita, F. Kanehiro, K. Kaneko, K. Miura, and K. Yokoi, "Balance control based on capture point error compensation for biped walking on uneven terrain," in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 734–740.

[6] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[7] R. J. Griffin, G. Wiedebach, S. Bertrand, A. Leonessa, and J. Pratt, "Walking stabilization using step timing and location adjustment on the humanoid robot, atlas," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[8] J. E. Pratt and S. V. Drakunov, "Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 4653–4660.

[9] O. E. Ramos and K. Hauser, "Generalizations of the capture point to nonlinear center of mass paths and uneven terrain," in *IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 851–858.

[10] T. Koolen, M. Posa, and R. Tedrake, "Balance control using center of mass height variation: Limitations imposed by unilateral contact," in *IEEE-RAS International Conference on Humanoid Robots*, 2016.

[11] S. Caron and B. Mallein, "Balance control using both zmp and com height variations: A convex boundedness approach," in *IEEE International Conference on Robotics and Automation*, May 2018.

[12] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2001, pp. 239–246.

[13] S. Caron, Q.-C. Pham, and Y. Nakamura, "Zmp support areas for multi-contact mobility under frictional constraints," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 67–80, Feb. 2017.

[14] M. Vukobratovi and J. Stepanenko, "On the stability of anthropomorphic systems," *Mathematical biosciences*, vol. 15, no. 1, pp. 1–37, 1972.

[15] M. Morisawa, S. Kajita, K. Kaneko, K. Harada, F. Kanehiro, K. Fujiwara, and H. Hirukawa, "Pattern generation of biped walking constrained on parametric surface," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 2405–2410.

[16] Y. Zhao and L. Sentis, "A three dimensional foot placement planner for locomotion in very rough terrains," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*, 2012, pp. 726–733.

[17] M. A. Hopkins, D. W. Hong, and A. Leonessa, "Humanoid locomotion on uneven terrain using the time-varying divergent component of motion," in *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 266–272.

[18] T. Kamioka, T. Watabe, M. Kanazawa, H. Kaneko, and T. Yoshiike, "Dynamic gait transition between bipedal and quadrupedal locomotion," in *IEEE/RSJ International Conf. on Int. Robots and Systems*, 2015.

[19] S. Caron and A. Kheddar, "Dynamic walking over rough terrains by nonlinear predictive control of the floating-base inverted pendulum," in *IEEE/RSJ International Conf. on Intelligent Robots and Systems*, 2017.

[20] J. Hauser, A. Saccon, and R. Frezza, "Achievable motorcycle trajectories," in *IEEE Conf. on Dec. and Control*, vol. 4, 2004, pp. 3944–3949.

[21] W. A. Coppel, "Dichotomies and stability theory," in *Symposium on Differential Equations and Dynamical Systems*, 1971, pp. 160–162.

[22] L. Lanari, S. Hutchinson, and L. Marchionni, "Boundedness issues in planning of locomotion trajectories for biped robots," in *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 951–958.

[23] B. Stephens, "Humanoid push recovery," in *IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 589–595.

[24] K. Yamamoto, "Control strategy switching for humanoid robots based on maximal output admissible set," *Robotics and Autonomous Systems*, vol. 81, pp. 17–32, Jul. 2016.

[25] V. Samy, S. Caron, K. Bouyarmane, and A. Kheddar, "Post-impact adaptive compliance for humanoid falls using predictive control of a reduced model," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2017.

[26] A. Del Prete, S. Tonneau, and N. Mansard, "Zero Step Capturability for Legged Robots in Multi Contact," Dec. 2017, submitted.

[27] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, Jun. 2018.

[28] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer-Verlag New York, 2006.

[29] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[30] Å. Björck, *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996.

[31] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006.

[32] P. E. E. Gill, S. J. Hammarling, W. Murray, M. A. Saunders, and M. H. Wright, "User's guide for lssol (version 1.0)," Standford University, Standord, California 94305, Tech. Rep. 86-1, January 1986.

[33] M. Khadiv, A. Herzog, S. A. A. Moosavian, and L. Righetti, "Step timing adjustment: A step toward generating robust gaits," in *IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 35–42.

[34] T. Sugihara and T. Yamamoto, "Foot-guided agile control of a biped robot through zmp manipulation," in *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, 2017.

[35] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The Int. Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[36] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *IEEE-RAS International Conference on Humanoid Robots*, 2006, pp. 137–142.

[37] C. Brasseur, A. Sherikov, C. Collette, D. Dimitrov, and P.-B. Wieber, "A robust linear MPC approach to online generation of 3D biped walking motion," in *IEEE–RAS Int. Conf. on Humanoid Robots*, 2015.

[38] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 295–302.

[39] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 3555–3561.

[40] S. Caron, A. Kheddar, and O. Tempier, "Stair climbing stabilization of the HRP-4 humanoid robot using whole-body admittance control," in *IEEE International Conference on Robotics and Automation*, May 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01875387

[41] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2003, pp. 1620–1626.

[42] S. Kajita and K. Tani, "Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode," in *IEEE International Conference on Robotics and Automation*, pp. 1405–1411.

[43] S. Nakaoka, S. Hattori, F. Kanehiro, S. Kajita, and H. Hirukawa, "Iterative contact force solver for simulating articulated rigid bodies," in *Annual conference of the Robotics Society of Japan*, 2007, in Japanese.

[44] K. Terada and Y. Kuniyoshi, "Online gait planning with dynamical 3d-symmetrization method," in *IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 222–227.

[45] A. Herdt, N. Perrin, and P.-B. Wieber, "Lmpc based online generation of more efficient walking motions," in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 390–395.

[46] N. Scianca, M. Cognetti, D. De Simone, L. Lanari, and G. Oriolo, "Intrinsically stable MPC for humanoid gait generation," in *IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 101–108.

[47] L. Lanari and S. Hutchinson, "Planning desired center of mass and zero moment point trajectories for bipedal locomotion," in *IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 637–642.

[48] T. Sugihara and Y. Nakamura, "Boundary condition relaxation method for stepwise pedipulation planning of biped robots," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 658–669, 2009.

[49] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3d com accelerations," in *IEEE-RAS International Conference on Humanoid Robots*, 2016.

[50] S. Kajita, M. Benallegue, R. Cisneros, T. Sakaguchi, S. Nakaoka, M. Morisawa, K. Kaneko, and F. Kanehiro, "Biped walking pattern generation based on spatially quantized dynamics," in *IEEE-RAS International Conference on Humanoid Robots*, 2017, pp. 599–605.

[51] S. Brossette, "Viable multi-contact posture computation for humanoid robots using nonlinear optimization on manifolds," Ph.D. dissertation, University of Montpellier, 2016.

[52] A. Escande, "Dedicated optimization for MPC with convex boundedness constraint," CNRS-AIST JRL UMI3218/RL, Tech. Rep., 2018. [Online]. Available: https://github.com/jrl-umi3218/CaptureProblemSolver/

APPENDIX A
MATHEMATICAL COMPLEMENT

In this Appendix, we provide formal proofs for claims made in Section II and verify that all quantities are soundly defined.

A. Restriction to convergent input functions

Property 5. *For every pair of states x_i and $x_f = (c_f, 0)$, $\mathcal{I}_{x_i, x_f}$ is non-empty if and only if $\mathcal{I}_{x_i, x_f}^c$ is non-empty.*

In other words, if there exists a capture input $\lambda(t), r(t)$ steering an initial state x_i to a static equilibrium x_f, then there exists another input $\lambda^c(t), r^c(t)$ accomplishing the same while also converging.

Proof. As $\mathcal{I}_{x_i, x_f}^c \subset \mathcal{I}_{x_i, x_f}$, it is enough to prove that $\mathcal{I}_{x_i, x_f}^c$ is non-empty as soon as $\mathcal{I}_{x_i, x_f}$ is non-empty, *i.e.* that given a input function $\lambda(t), r(t) \in \mathcal{I}_{x_i, x_f}$ we can find another input $\lambda^c(t), r^c(t)$ steering to the same state while converging. To this end, consider the following state-dependent inputs:

$$\sqrt{\bar{\lambda}(x)} = 2 \frac{\sqrt{g(5h - h_f) + \dot{h}^2} - \dot{h}_i}{5h_i - h_f} \tag{81}$$

$$\bar{r}(x) = c + \frac{c - c_f}{4} + \frac{\dot{c}}{\sqrt{\bar{\lambda}(x)}} + \frac{g}{\bar{\lambda}(x)} \tag{82}$$

This definition is chosen so that $\bar{\lambda}(x)$ is the solution of:

$$\left[c - o + \frac{c - c_f}{4} \right] \cdot n + \frac{(\dot{c} \cdot n)}{\sqrt{X}} + \frac{(g \cdot n)}{X} = 0 \tag{83}$$

As a consequence, $(\bar{r}(x_i) - o) \cdot n = 0$ and the state-dependent CoP belongs to the contact area. Moreover, $\bar{\lambda}$ and $\bar{r}$ are continuous functions of x_i in a neighbourhood of x_f, and $\bar{\lambda}(x_f) = \lambda_f(c_f)$ and $\bar{r}(x_f) = r_f(c_f)$. Hence, as long as $x(t)$ is close enough to x_f, both $\bar{\lambda}(x(t))$ and $\bar{r}(x(t))$ are feasible.

Injecting those inputs into (5) yields the nonlinear differential equation:

$$\ddot{c}(t) = -\frac{\bar{\lambda}(x(t))}{4}(c(t) - c_f) - \sqrt{\bar{\lambda}}\dot{c}(t) \tag{84}$$

It is immediate that x_f is an equilibrium for this dynamics. The linearized system around this equilibrium is:

$$\ddot{c}^\ell(t) = -\frac{\lambda_f}{4}(c^\ell(t) - c_f) - \sqrt{\lambda_f}\dot{c}^\ell(t) \tag{85}$$

for which the equilibrium is stable. Therefore the equilibrium x_f of (84) is locally stable: if $x(0)$ is close enough to x_f, then $x(t)$ remains close to x_f and converges toward this limit.

We now consider a generic input function $\lambda(t), r(t) \in \mathcal{I}_{x_i, x_f}$. By definition, the solution of (5) converges to x_f as $t \to \infty$. Then, there exists some time T such that $x(T)$ is close enough to x_f so that, starting from this position, the state-dependent control remains feasible and converges to x_f. We conclude by noting that the input function that switches at time T from λ, r to $\bar{\lambda}, \bar{r}$ belongs to $\mathcal{I}_{x_i, x_f}^c$. $\square$

B. Solutions to the Riccati equations

Let us verify the existence of damping solutions and exhibit some of their properties that will prove useful to characterize those that don't diverge.

Property 6. *Assume that we are given λ such that $\lambda(t) \in [\lambda_{min}, \lambda_{max}]$ at all times t. Then, there exists a unique $\omega_i > 0$ such that the solution ω of (18) with $\omega(0) = \omega_i$ is positive and finite at all times. Moreover, this solution is such that:*

$$\forall t > 0, \quad \omega(t) \in \left[\sqrt{\lambda_{min}}, \sqrt{\lambda_{max}} \right] \tag{86}$$

In other words, there is a one-to-one mapping between the stiffness function $\lambda(t)$ and its non-diverging filtered damping $\omega(t)$.

Proof. Consider first the case of a constant input λ. One can note that the differential equation $\dot{y} = y^2 - \lambda$ has two equilibrium points: one stable $-\sqrt{\lambda}$ and one unstable $\sqrt{\lambda}$. More precisely, given $y_0 \in \mathbb{R}$, the only solution of this equation satisfying $y(0) = y_0$ is given by

$$y(t) = \begin{cases} \frac{\sqrt{\lambda}}{\tanh(\sqrt{\lambda}(T-t))} & \text{if } |y_0| > \sqrt{\lambda} \\ \sqrt{\lambda}\tanh(\sqrt{\lambda}(T-t)) & \text{if } |y_0| < \sqrt{\lambda} \\ y_0 & \text{if } |y_0| = \sqrt{\lambda} \end{cases} \tag{87}$$

The initial condition y_0 settles the behavior of the solution at all times. Define the time $T = \frac{1}{2\sqrt{\lambda}} \log \left| \frac{y_0 - \sqrt{\lambda}}{y_0 + \sqrt{\lambda}} \right|$, then:

- If $0 \le y_0 < \sqrt{\lambda}$, then $\lim_{t \to \infty} y(t) = -\sqrt{\lambda}$ and the solution y becomes negative after time T.
- If $y_0 = \sqrt{\lambda}$, then $y(t) = \sqrt{\lambda}$ for all $t > 0$.
- If $y_0 > \sqrt{\lambda}$, then $\lim_{t \to T} y(t) = +\infty$: the solution explodes in finite time.

Let us move now to the general case where $\lambda(t)$ is time-varying, and denote by ω a non-negative, non-explosive solution to (18). If $\omega(t_0) > \sqrt{\lambda_{max}}$ at some time $t_0 > 0$, then choosing the solution y to $\dot{y} = y^2 - \lambda_{max}$ with $y(t_0) = \omega(t_0)$, we observe that, as long as $0 \le y(t) \le \omega(t)$,

$$\dot{\omega}(t) - \dot{y}(t) = \omega^2(t) - y^2(t) + \lambda_{max} - \lambda(t) \ge 0 \tag{88}$$

Therefore, $\omega - y$ is nondecreasing, and $y(t) \leq \omega(t)$ holds until the explosion time T of y. This shows that ω explodes in finite time, in contradiction with the hypothesis. Similarly, if $\omega(t_0) < \sqrt{\lambda_{\min}}$ for some $t_0 > 0$, we can upper-bound $\omega(t)$ by the solution to $\dot{y} = y^2 - \lambda_{\min}$ such that $y(t_0) = \omega(t_0)$, which becomes negative in finite time, once again contradicting the hypothesis. The bounds (86) must therefore hold.

We now prove the uniqueness of the non-negative non-exploding solution of (18). Suppose that one could find two such solutions, ω_1 and ω_2. As observed above, these functions remain in the interval $[\sqrt{\lambda_{\min}}, \sqrt{\lambda_{\max}}]$. Consider without loss of generality that $\omega_1(0) > \omega_2(0)$. Then, as long as $\omega_1(t) > \omega_2(t)$,

$$\dot{\omega}_1 - \dot{\omega}_2 = (\omega_1 - \omega_2)(\omega_1 + \omega_2) \geq 2\sqrt{\lambda_{\min}}(\omega_1 - \omega_2) \quad (89)$$

As a consequence, $\omega_1(t) - \omega_2(t) \geq (\omega_1(0) - \omega_2(0))e^{2\sqrt{\lambda_{\min}}t}$ at all times, showing that the two functions cannot be bounded at the same time.

To finally prove the existence of the solution, we observe there exists a unique $\overline{\omega}(0) \in [\sqrt{\lambda_{\min}}, \sqrt{\lambda_{\max}}]$ such that for all $t > 0$, we have $\overline{y}_t(0) > \omega(0) > \underline{y}_t(0)$ where $\overline{y}_t$ and $\underline{y}_t$ are the solutions of (18) with conditions $\overline{y}_t(t) = \sqrt{\lambda_{\max}}$ and $\underline{y}_t(t) = \sqrt{\lambda_{\min}}$. The solution ω starting from this value $\omega(0)$ remains within bounds by construction (the time it crosses $\sqrt{\lambda_{\min}}$ or $\sqrt{\lambda_{\max}}$ is greater than any finite time t). $\quad\square$

Let us now turn to the other damping γ. While there is a unique solution ω corresponding to a given λ, there are many different non-negative finite functions γ that satisfy (17). As a matter of fact, each choice of $\gamma(0) > 0$ yields an admissible solution:

Property 7. *Assume that we are given λ such that $\lambda(t) \in [\lambda_{\min}, \lambda_{\max}]$ at all times t. For all $\gamma(0) > 0$, the solution γ of* (17) *is non-negative and finite at all times. Moreover:*

$$\sqrt{\lambda_{\min}} \leq \liminf_{t\to\infty} \gamma(t) \leq \limsup_{t\to\infty} \gamma(t) \leq \sqrt{\lambda_{\max}} \quad (90)$$

Proof. The existence and the uniqueness of the solution on a maximal interval are consequences of the Cauchy-Lipschitz theorem. As in the previous proof, we can compare γ with the functions $\underline{y}$ and $\overline{y}$, respectively solutions to $\dot{y} = \lambda_{\min} - y^2$ and $\dot{y} = \lambda_{\max} - y^2$ with $\underline{y}(0) = \overline{y}(0) = \gamma(0)$. Then, $\underline{y}(t) \leq \gamma(t) \leq \overline{y}(t)$ at all times t, and the rest of the proof is a consequence of $\lim_{t\to\infty} \underline{y}(t) = \sqrt{\lambda_{\min}}$ and $\lim_{t\to\infty} \overline{y}(t) = \sqrt{\lambda_{\max}}$. $\quad\square$

An interesting consequence of these two properties is the following asymptotic behavior:

Corollary 8. *If $\lim_{t\to\infty} \lambda(t) = \lambda_{\mathrm{f}}$, then*

$$\lim_{t\to\infty} \omega(t) = \lim_{t\to\infty} \gamma(t) = \sqrt{\lambda_{\mathrm{f}}}. \quad (91)$$

Proof. We only consider the case of ω, the proof for γ following the same derivation. By definition of the limit, for any $\epsilon > 0$, there exists $t_0 > 0$ large enough so that $\forall t > t_0, |\lambda(t) - \lambda_{\mathrm{f}}| < \epsilon$. Next, remark that the time-shifted function $\widetilde{\omega}(t) := \omega(t + t_0)$ is a solution of the equation $\dot{\widetilde{\omega}}(t) = \widetilde{\omega}(t)^2 - \widetilde{\lambda}$, where $\forall t > 0, \widetilde{\lambda}(t) = \lambda(t + t_0) \in [\lambda_{\mathrm{f}} - \epsilon, \lambda_{\mathrm{f}} + \epsilon]$. Property 6 then shows that $\omega(t + t_0) \in [\sqrt{\lambda_{\mathrm{f}} - \epsilon}, \sqrt{\lambda_{\mathrm{f}} + \epsilon}]$ for all $t > 0$. As a consequence, for any $\epsilon > 0$:

$$\sqrt{\lambda_{\mathrm{f}} - \epsilon} \leq \liminf_{t\to\infty} \omega(t) \leq \limsup_{t\to\infty} \omega(t) \leq \sqrt{\lambda_{\mathrm{f}} + \epsilon} \quad (92)$$

Taking the limit as $\epsilon \to 0$, we conclude that the limit of ω as $t \to \infty$ exists and is equal to $\sqrt{\lambda_{\mathrm{f}}}$. $\quad\square$

We conclude from the above properties that the solutions $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ to Equation (19) are well-defined.

C. Convergent component of motion

The component $\boldsymbol{\zeta}$ corresponding to the damping γ is subject to the differential equation:

$$\dot{\boldsymbol{\zeta}} = -\gamma\boldsymbol{\zeta} + (\lambda\boldsymbol{r} - \boldsymbol{g}) \quad (93)$$

The general solution to this equation is given by:

$$\boldsymbol{\zeta}(t) = \left(\boldsymbol{\zeta}(0) + \int_0^t e^{\Gamma(\tau)}(\lambda(\tau)\boldsymbol{r}(\tau) - \boldsymbol{g})\mathrm{d}\tau\right)e^{-\Gamma(t)} \quad (94)$$

where Γ is the antiderivative of γ such that $\Gamma(0) = 0$, *i.e.* $\Gamma(t) = \int_0^t \gamma(t)\mathrm{d}t$. It satisfies the following two identities:

$$\frac{\mathrm{d}e^{\Gamma}}{\mathrm{d}t} = \gamma e^{\Gamma} \qquad \frac{\mathrm{d}^2 e^{\Gamma}}{\mathrm{d}t^2} = (\dot{\gamma} + \gamma^2)e^{\Gamma} = \lambda e^{\Gamma} \quad (95)$$

The asymptotic behavior of $\boldsymbol{\zeta}$ is tied to that of the two inputs λ and $\boldsymbol{r}$ of the inverted pendulum:

Property 9. *Consider an input function $\lambda(t), \boldsymbol{r}(t)$ such that $\lim_{t\to\infty} \lambda(t) = \lambda_{\mathrm{f}}$ and $\lim_{t\to\infty} \boldsymbol{r}(t) = \boldsymbol{r}_{\mathrm{f}}$, and let γ denote any solution to* (17). *Then, the solution $\boldsymbol{\zeta}$ of* (93) *satisfies:*

$$\lim_{t\to\infty} \boldsymbol{\zeta}(t) = \sqrt{\lambda_{\mathrm{f}}}\left(\boldsymbol{r}_{\mathrm{f}} - \frac{\boldsymbol{g}}{\lambda_{\mathrm{f}}}\right) = \sqrt{\lambda_{\mathrm{f}}}\boldsymbol{c}_{\mathrm{f}} \quad (96)$$

Proof. By Corollary 8, $\lim_{t\to\infty} \gamma(t) = \sqrt{\lambda_{\mathrm{f}}}$, therefore its antiderivative $\Gamma(t)$ satisfies $\lim_{t\to\infty} \Gamma(t)/t = \sqrt{\lambda_{\mathrm{f}}}$ as well. In particular, $\Gamma(t)$ diverges to ∞, so that:

$$\boldsymbol{\zeta}(t) \underset{t\to\infty}{\sim} e^{-\Gamma(t)}\int_0^t e^{\Gamma(\tau)}(\lambda(\tau)\boldsymbol{r}(\tau) - \boldsymbol{g})\mathrm{d}\tau \quad (97)$$

where the notation $f \sim g$ means that the ratio f/g goes to 1 as $t \to \infty$. Applying l'Hôpital's rule, we conclude that:

$$\begin{aligned}\lim_{t\to\infty} \boldsymbol{\zeta}(t) &= \lim_{t\to\infty} \frac{\int_0^t e^{\Gamma(\tau)}(\lambda(\tau)\boldsymbol{r}(\tau) - \boldsymbol{g})\mathrm{d}\tau}{e^{\Gamma(t)}} \quad (98)\\ &= \lim_{t\to\infty} \frac{e^{\Gamma(t)}(\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})}{\gamma(t)e^{\Gamma(t)}} = \frac{(\lambda_{\mathrm{f}}\boldsymbol{r}_{\mathrm{f}} - \boldsymbol{g})}{\sqrt{\lambda_{\mathrm{f}}}} \quad\square\end{aligned}$$

D. Divergent component of motion

The general solution (21) of the divergent component of motion is based on the antiderivative $\Omega(t) = \int_0^t \omega(t)\mathrm{d}t$. Recalling from Property 6 that $\omega \in [\sqrt{\lambda_{\min}}, \sqrt{\lambda_{\max}}]$, we see that Ω grows at least linearly. Therefore, as long as λ and $\boldsymbol{r}$ remain bounded, the integral $\int_0^\infty e^{-\Omega(\tau)}(\boldsymbol{g} - \lambda(\tau)\boldsymbol{r}(\tau))\mathrm{d}\tau$ is well-defined and finite. Let us now prove Property 1.

Proof. The proof is very similar to that of Property 9. The solution (21) with $\boldsymbol{\xi}(0) = \boldsymbol{\xi}_{\mathrm{i}}$ from Equation (22) becomes:

$$\boldsymbol{\xi}(t) = e^{\Omega(t)}\int_t^\infty e^{-\Omega(\tau)}(\lambda(\tau)\boldsymbol{r}(\tau) - \boldsymbol{g})\mathrm{d}\tau \quad (99)$$

Applying l'Hôpital's rule, we conclude by Corollary 8 that:

$$\lim_{t\to\infty} \boldsymbol{\xi}(t) = \lim_{t\to\infty} \frac{e^{-\Omega(t)}(\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})}{\omega(t)e^{-\Omega(t)}} = \frac{\lambda_{\mathrm{f}}\boldsymbol{r}_{\mathrm{f}} - \boldsymbol{g}}{\sqrt{\lambda_{\mathrm{f}}}} \quad (100)$$

That is to say, similarly to the convergent component of motion, the solution $\boldsymbol{\xi}$ of (20) converges to $\sqrt{\lambda_{\mathrm{f}}}\boldsymbol{c}_{\mathrm{f}}$. $\quad\square$

17

E. Proof of Property 2

Proof of the ⇒ implication. Let $\lambda(t), \boldsymbol{r}(t)$ denote a capture input from $\mathcal{I}^c_{\boldsymbol{x}_i,\boldsymbol{x}_f}$, with $\boldsymbol{x}(t)$ the smooth trajectory resulting from this input via the equation of motion (5). Its boundary values $\boldsymbol{x}(0) = \boldsymbol{x}_i$ and $\boldsymbol{x}(\infty) = \boldsymbol{x}_f$ being bounded, this trajectory must be bounded as well. As $\boldsymbol{\zeta}(t)$ is always bounded by Property 9, this in turns implies that its the divergent component $\boldsymbol{\xi}(t)$ is bounded, and must therefore satisfy Equation (25) by Property 1. Next, let us denote by $\lambda_f, \boldsymbol{r}_f$ the limits of $\lambda(t), \boldsymbol{r}(t)$ as time goes to infinity. Using Properties 9 and 1, the two components converge to:

$$\lim_{t\to\infty} \boldsymbol{\zeta}(t) = \lim_{t\to\infty} \boldsymbol{\xi}(t) = \sqrt{\lambda_f}\left(\boldsymbol{r}_f - \frac{\boldsymbol{g}}{\lambda_f}\right) \quad (101)$$

Recalling from Corollary 8 that γ and ω converge to $\sqrt{\lambda_f}$, we can take the limit in the mapping (10)–(12):

$$\lim_{t\to\infty} \boldsymbol{c}(t) = \lim_{t\to\infty} \frac{\boldsymbol{\zeta}(t) + \boldsymbol{\xi}(t)}{\gamma(t) + \omega(t)} = \boldsymbol{r}_f - \frac{\boldsymbol{g}}{\lambda_f} = \boldsymbol{c}_f \quad (102)$$

Therefore, $\lambda_f = \lambda_f(\boldsymbol{c}_f)$ and $\boldsymbol{r}_f = \boldsymbol{r}_f(\boldsymbol{c}_f)$. □

Proof of the ⇐ implication. Reciprocally, assuming (*i*)–(*iii*), Equation (101) holds again by Properties 9–1 and Corollary 8. Furthermore,

$$\lim_{t\to\infty} \dot{\boldsymbol{c}}(t) = \lim_{t\to\infty} \frac{-\omega(t)\boldsymbol{\zeta}(t) + \gamma(t)\boldsymbol{\xi}(t)}{\gamma(t) + \omega(t)} = \boldsymbol{0} \quad (103)$$

Thus, the pendulum driven by $\lambda(t), \boldsymbol{r}(t)$ converges to the static equilibrium $\boldsymbol{x}_f$. □

F. Note on fixed-CoP strategies

When the CoP input is stationary, *i.e.* in a point-foot model, the Gram determinant $G = ((\boldsymbol{c}-\boldsymbol{r})\times\dot{\boldsymbol{c}})\cdot\boldsymbol{g}$ becomes invariant.

Short proof. Take the cross-product of Equation (5) with $\ddot{\boldsymbol{c}}$. Then, the scalar product of the resulting expression with $\boldsymbol{g}$ yields $((\boldsymbol{c} - \boldsymbol{r}) \times \ddot{\boldsymbol{c}}) \cdot \boldsymbol{g} = 0$. Conclude by noting that this formula is the time derivative of $((\boldsymbol{c} - \boldsymbol{r}) \times \dot{\boldsymbol{c}}) \cdot \boldsymbol{g}$. □

There are two possible outcomes: either $G = 0$, which means the three vectors are coplanar and the robot may stabilize using a 2D strategy [8], [9], [10]; or $G \neq 0$ and it is impossible to bring the system to an equilibrium where $\dot{\boldsymbol{c}} = 0$. This shows simultaneously two properties: first, that sagittal 2D balance control is the most general solution for point-foot models, and second, that these models have a very limited ability to balance, as they need to re-step at the slightest lateral change in linear momentum. The ability of flat-footed bipeds to absorb these perturbations (to some extent) without stepping comes from continuous CoP variations.

APPENDIX B
NUMERICAL OPTIMIZATION COMPLEMENT

In this Appendix, we first recall terminology and state-of-the-art algorithms for numerical optimization. We essentially rewrite treatment from [28] for double-sided inequality constraints. We then detail all tailored operations mentioned in Section IV.

A. Definitions and notations

Consider the optimization problem:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \; f(\boldsymbol{x}) \quad (104a)$$

$$\text{subject to } \boldsymbol{l} \leq \boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{u} \quad (104b)$$

where f and $\boldsymbol{h}$ are smooth functions, f being 1-dimensional and $\boldsymbol{h}$ m-dimensional. Lower and upper bound constraints are represented by vectors $\boldsymbol{l}, \boldsymbol{u} \in \mathbb{R}^m$, with equality constraints specified by taking $l_j = u_j$. A point $\boldsymbol{x}$ is *feasible* if it satisfies all constraints. For a given $\boldsymbol{x}$, we say the jth constraint is *active* at its lower (resp. upper) bound when $h_j(\boldsymbol{x}) = l_j$ (resp. $h_j(\boldsymbol{x}) = u_j$). We denote by:

$$\mathcal{E} := \{j \in [1, m], l_j = u_j\} \quad (105)$$

$$\mathcal{A}(\boldsymbol{x})^- := \{j \notin \mathcal{E}, h_j(\boldsymbol{x}) = l_j\} \quad (106)$$

$$\mathcal{A}(\boldsymbol{x})^+ := \{j \notin \mathcal{E}, h_j(\boldsymbol{x}) = u_j\} \quad (107)$$

These three sets are disjoint. For a set of indexes $\mathcal{S}$ and a matrix $\mathbf{M}$, we define $\mathbf{M}_\mathcal{S}$ the matrix made of the rows of $\mathbf{M}$ whose indexes are in $\mathcal{S}$ (this notation also applies to vectors).

The *Lagrangian* of the problem is defined as $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}^-, \boldsymbol{\lambda}^+) := f(\boldsymbol{x}) + \boldsymbol{\lambda}^{-T}(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{l}) + \boldsymbol{\lambda}^{+T}(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{u})$ where $\boldsymbol{\lambda}^-, \boldsymbol{\lambda}^+ \in \mathbb{R}^m$ are the *Lagrange multipliers*, $\boldsymbol{\nabla}_{\boldsymbol{x}} g$ and $\boldsymbol{\nabla}^2_{\boldsymbol{xx}} g$ are respectively the gradient and Hessian of a function g with respect to $\boldsymbol{x}$. We note $\boldsymbol{\lambda} := \boldsymbol{\lambda}^- + \boldsymbol{\lambda}^+$. We can work with it instead of $\boldsymbol{\lambda}^-$ and $\boldsymbol{\lambda}^+$ (see [51, §4.3.5]).

The *Karush–Kuhn–Tucker* (KKT) conditions give necessary conditions on $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ for $\boldsymbol{x}$ to be a minimizer of Problem (104) (see [28, chap. 12]). They are often used as termination conditions in solvers.

B. Active-set method for Quadratic Programming

When the objective f is quadratic, $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} + \boldsymbol{q}^T\boldsymbol{x}$, with $\mathbf{Q}$ symmetric positive semidefinite and $\boldsymbol{h}$ linear, $\boldsymbol{h}(\boldsymbol{x}) = \mathbf{C}\boldsymbol{x}$ for some matrix $\mathbf{C}$, Problem (104) is a (convex) Quadratic Program with Inequality constraints (QPI). One of the main approaches to solve it is the *active-set* method. This method iteratively discovers the set of constraints active at the solution[18] by solving at each iteration k the following Quadratic Program with only Equality constraints (QPE):

$$\underset{\boldsymbol{p}\in\mathbb{R}^n}{\text{minimize}} \; \frac{1}{2}\boldsymbol{p}^T\mathbf{Q}\boldsymbol{p} + (\mathbf{Q}\boldsymbol{x}_k + \boldsymbol{q})^T\boldsymbol{p} \quad (108a)$$

$$\text{subject to } \mathbf{C}_{\mathcal{W}_k}\boldsymbol{p} = 0 \quad (108b)$$

where $\mathcal{W}_k$ is a set of indexes. The solution $\boldsymbol{p}^*$ to this QPE is used to determine the next iterate $\boldsymbol{x}_{k+1}$.

Unlike QPIs, QPEs admit analytical solutions as their KKT conditions reduce to a linear system. For a given $\boldsymbol{x}_k$, we can retrieve $\boldsymbol{\lambda}$ with:

$$\boldsymbol{\lambda}_{\mathcal{A}_k} = -\mathbf{C}^{\dagger T}_{\mathcal{A}_k}\boldsymbol{\nabla}^T_{\boldsymbol{x}}f(\boldsymbol{x}_k), \qquad \lambda_i = 0, \forall i \notin \mathcal{A}_k \quad (109)$$

where $\square^\dagger$ denotes the Moore-Penrose pseudo-inverse and $\mathcal{A}_k = \mathcal{A}(\boldsymbol{x}_k)^- \cup \mathcal{A}(\boldsymbol{x}_k)^+ \cup \mathcal{E}$ is the active set at $\boldsymbol{x}_k$, *i.e.* the set of constraints that are active at this point.

The active-set method for convex QPIs is given in Algorithm 1. See [28, chap. 16] for more details on this method.

---

[18] We ignore here for the sake of simplicity a subtlety arising when active constraints are linearly dependent.

Algorithm 1 Active-set algorithm for convex QPI

> Given a feasible point $\boldsymbol{x}_0$
> Let $\mathcal{W}_0^- = \mathcal{A}(\boldsymbol{x}_0)^-$, $\mathcal{W}_0^+ = \mathcal{A}(\boldsymbol{x}_0)^+$
> **for** $k = 0, 1, 2, \ldots$ **do**
>   Compute $\boldsymbol{p}$ from (108) with $\mathcal{W}_k = \mathcal{W}_k^- \cup \mathcal{W}_k^+ \cup \mathcal{E}$
>   **if** $\boldsymbol{p} = 0$ **then**
>   Compute $\boldsymbol{\lambda}$ using Equation (109)
>   **if** $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ verify the KKT conditions **then**
>   **return** the solution $\boldsymbol{x}_{\mathrm{f}} = \boldsymbol{x}_k$
>   **else**
>   Choose j such that λ_j violates the KKT conditions
> $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, $\mathcal{W}_{k+1}^- = \mathcal{W}_k^- \setminus \{j\}$, $\mathcal{W}_{k+1}^+ = \mathcal{W}_k^+ \setminus \{j\}$
>   **end if**
>   **else**
>   Find the largest $\alpha \leq 1$ such that $\boldsymbol{x}_k + \alpha \boldsymbol{p}$ is feasible.
> $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \boldsymbol{p}$
>   **if** some constraints have been activated doing so **then**
>   Let j be the index of one of them
>   Obtain $\mathcal{W}_{k+1}^-$ and $\mathcal{W}_{k+1}^+$ from $\mathcal{W}_k^-$ and $\mathcal{W}_k^+$ by adding j to the appropriate set
>   **else**
> $\mathcal{W}_{k+1}^- = \mathcal{W}_k^-$, $\mathcal{W}_{k+1}^+ = \mathcal{W}_k^+$.
>   **end if**
>   **end if**
> **end for**

C. Sequential Quadratic Programming

Sequential quadratic programming (SQP) is an iterative optimization technique for solving general constrained problems such as (104). At each iteration k, a QP approximation of (104) is formed and solved:

$$\underset{\boldsymbol{p} \in \mathbb{R}^n}{\text{minimize}} \; f(\boldsymbol{x}_k) + \boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k)^T \boldsymbol{p} + \frac{1}{2} \boldsymbol{p}^T \mathbf{B}_k \boldsymbol{p} \tag{110a}$$

$$\text{subject to } \boldsymbol{l} - \boldsymbol{h}(\boldsymbol{x}_k) \leq \boldsymbol{\nabla}_{\boldsymbol{x}} \boldsymbol{h}(\boldsymbol{x}_k)^T \boldsymbol{p} \leq \boldsymbol{u} - \boldsymbol{h}(\boldsymbol{x}_k) \tag{110b}$$

where $\mathbf{B}_k$ can be $\boldsymbol{\nabla}_{\boldsymbol{xx}}^2 \mathcal{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ or, for faster computations, some positive-definite approximation of it.

Algorithm 2 Line search SQP

> Given a stepping parameter $\tau \in (0, 1)$
> Choose $(\boldsymbol{x}_0, \boldsymbol{\lambda}_0)$
> **while** the KKT conditions are not satisfied **do**
>   Compute $\boldsymbol{p}$ from (110)
>   Let $\boldsymbol{\lambda}$ be the corresponding multiplier and $\boldsymbol{p}_\lambda = \boldsymbol{\lambda} - \boldsymbol{\lambda}_k$
> $\alpha = 1$
>   **while** $\alpha \boldsymbol{p}$ does not yield an acceptable step **do**
> $\alpha = \tau \alpha$
>   **end while**
> $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \boldsymbol{p}$, $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha \boldsymbol{p}_\lambda$
> **end while**

The outline of the SQP method is given in Algorithm 2. There are several criteria for assessing whether a step is acceptable, see [28, chap. 18] for details.

D. Cost matrix of the unconstrained least squares problem

Consider the active set $\mathcal{W}$ for a given SQP iteration. Starting at the first constraint, count the number a_0 of consecutive active constraints (possibly 0 if the first constraint is not active), then j_1 the number of following consecutive inactive constraints, a_1 the number of following active constraints, *etc.* The set $\mathcal{W}$ is then fully described by the sequence $(a_0, j_1, a_1, j_2, a_2, \ldots, j_p, a_p)$, where only a_0 and a_p can be 0. Note that $\sum_k a_k + \sum_k j_k = n + 1$, and let us define $n_{\mathcal{W}} := \sum_k a_k$. For example, if $\mathcal{W} = \{1, 2, 6, 9, 10, 11, 13, 14\}$ for $n = 15$ optimization variables, we get the sequence $(2, 3, 1, 2, 3, 1, 2, 2, 0)$ and $n_{\mathcal{W}} = 8$. The constraint matrix $\mathbf{C}_{\mathcal{W}}$ is then the $n_{\mathcal{W}} \times n$ matrix:

$$\mathbf{C}_{\mathcal{W}} = \begin{bmatrix} \mathbf{C}_0 & & & & & \\ & \mathbf{0}_{a_1, j_1 - 1} & \mathbf{C}_1 & & & \\ & & & \mathbf{0}_{a_2, j_2 - 1} & \mathbf{C}_2 & \\ & & & & & \ddots \\ & & & & & & \mathbf{0}_{a_p, j_p - 1} & \mathbf{C}_p \end{bmatrix} \tag{111}$$

where $\mathbf{0}_{m,q}$ is the $m \times q$ zero matrix, while $\mathbf{C}_0$, $\mathbf{C}_k$ ($k \in [1, p-1]$) and $\mathbf{C}_p$ are respectively $a_0 \times a_0$, $a_k \times (a_k + 1)$ and $a_p \times a_p$ matrices ($\mathbf{C}_0$ and $\mathbf{C}_p$ can be empty) of the form:

$$\mathbf{C}_0 = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \; \mathbf{C}_k = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \; \mathbf{C}_p = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & 1 \end{bmatrix} \tag{112}$$

Denoting by $\mathbf{1}_a$ the vector of size a filled with ones, the nullspace projection matrix for the active set $\mathcal{W}$ is:

$$\mathbf{N}_{\mathcal{W}} = \begin{bmatrix} \mathbf{0}_{a_0, i_1 - 1} & & & & & \\ \mathbf{I}_{i_1 - 1} & & & & & \\ & \mathbf{1}_{a_1 + 1} & & & & \\ & & \mathbf{I}_{i_2 - 1} & & & \\ & & & \mathbf{1}_{a_2 + 1} & & \\ & & & & \ddots & \\ & & & & & \mathbf{I}_{i_p - 1} \\ & & & & & \mathbf{0}_{a_p, i_p - 1} \end{bmatrix} \tag{113}$$

Noting that $\mathbf{C}_k \mathbf{1}_{a_k + 1} = \mathbf{0}_{a_k, 1}$, we can directly verify that $\mathbf{C}_{\mathcal{W}} \mathbf{N}_{\mathcal{W}} = \mathbf{0}$. The matrix $\mathbf{N}_{\mathcal{W}}$ is n by $n - n_{\mathcal{W}}$ and full column rank. It is thus a basis of the nullspace of $\mathbf{C}_{\mathcal{W}}$.

Computing the product $\mathbf{M} \mathbf{N}_{\mathcal{W}}$ for a given matrix $\mathbf{M}$ does not actually require to perform any multiplication: multiplying by $\mathbf{1}$ amounts simply to the summation of columns of $\mathbf{M}$. Likewise, $\mathbf{N}_{\mathcal{W}} \boldsymbol{z}$ just requires to copy the elements of $\boldsymbol{z}$. It is thus not necessary to form $\mathbf{N}_{\mathcal{W}}$, and $\mathbf{T}$ can be obtained by $\sum a_k = n_{\mathcal{W}}$ vector additions. Taking into account the tridiagonal structure of $\mathbf{J}$, this can be done in $O(n)$.

E. Computation of Lagrange multipliers

The computation of the Lagrange multipliers, needed to check KKT conditions, relies on the pseudoinverse of $\mathbf{C}_{\mathcal{W}}$ (see

e.g. Equation (109)). Due to its block structure, expressing the latter is done by finding the pseudoinverse for each $\mathbf{C}_k$:

$$\mathbf{C}_{\mathcal{W}}^{\dagger} = \begin{bmatrix} \mathbf{C}_0^{-1} & & & & & \\ & \mathbf{0}_{i_1-1,a_1} & & & & \\ & \mathbf{C}_1^{\dagger} & & & & \\ & & \mathbf{0}_{i_2-1,a_2} & & & \\ & & \mathbf{C}_2^{\dagger} & & & \\ & & & \ddots & & \\ & & & & \mathbf{0}_{i_p-1,a_p} & \\ & & & & \mathbf{C}_p^{-1} & \end{bmatrix} \tag{114}$$

where $\square^{\dagger}$ denotes the pseudoinverse. It can be verified that

$$\mathbf{C}_0^{-1} = \begin{bmatrix} 1 & & \\ \vdots & \ddots & \\ 1 & \cdots & 1 \end{bmatrix}, \quad \mathbf{C}_p^{-1} = \begin{bmatrix} -1 & \cdots & -1 & 1 \\ & \ddots & \vdots & 1 \\ & & -1 & 1 \\ & & & 1 \end{bmatrix} \tag{115}$$

$$\mathbf{C}_k^{\dagger} = \frac{1}{a_k+1} \begin{bmatrix} -a_k & -(a_k-1) & -(a_k-2) & \cdots & -1 \\ 1 & -(a_k-1) & -(a_k-2) & \cdots & -1 \\ 1 & 2 & -(a_k-2) & \cdots & -1 \\ 1 & 2 & 3 & \ddots & \vdots \\ \vdots & & & \ddots & -1 \\ 1 & 2 & 3 & \cdots & a_k \end{bmatrix} \tag{116}$$

The computation of Lagrange multipliers can thus be done in $O(n^2)$ without forming the pseudoinverse explicitly.

F. QR decomposition of the cost matrix $\mathbf{T}$

The decomposition can be performed in two steps: first the QR decomposition $\mathbf{J}_{\mathcal{W}} = \mathbf{Q}_{\mathcal{W}} \mathbf{R}_{\mathcal{W}}$ of $\mathbf{J}_{\mathcal{W}} := \mathbf{J} \mathbf{N}_{\mathcal{W}}$, followed by the QR decomposition $\begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{R}_{\mathcal{W}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_1^T \\ \mathbf{Q}_2 \end{bmatrix} \mathbf{R}$. Combining these two yields $\mathbf{T}$ by:

$$\begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{J} \mathbf{N}_{\mathcal{W}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{Q}_{\mathcal{W}} \end{bmatrix} \begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{R}_{\mathcal{W}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{Q}_{\mathcal{W}} \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1^T \\ \mathbf{Q}_2 \end{bmatrix} \mathbf{R}$$
$$= \begin{bmatrix} \boldsymbol{q}_1^T \\ \mathbf{Q}_{\mathcal{W}} \mathbf{Q}_2 \end{bmatrix} \mathbf{R} \tag{117}$$

The matrix $\begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{R}_{\mathcal{W}} \end{bmatrix}$ is upper Hessenberg, so that its QR decomposition is computed in $O(n^2)$ [29, Chapter 5].

The decomposition of $\mathbf{J}_{\mathcal{W}}$ can be achieved in $O(n)$ by taking advantage of its structure. To avoid going through several corner cases, we sketch informally how this is done with the help of the example in Figure 12 (more details can be found in [52]). Because the sum of three non-zero elements on any row (ignoring the first) of $\mathbf{J}$ is zero, a careful study reveals that $\mathbf{J}_{\mathcal{W}}$ is made of p tridiagonal blocks, one for each group of consecutive inactive constraints. Blocks j and $j+1$ are separated by $a_j - 1$ rows of zeros, and the last column of the first block is aligned with the first column of the second block (Figure 12, left). We can perform QR decompositions for each blocks separately and denote by $\tilde{\mathbf{Q}}_{\mathcal{W}}$ the product of all orthogonal matrices. All blocks, except possibly the last one, have a rank equal to their row size minus one, so that
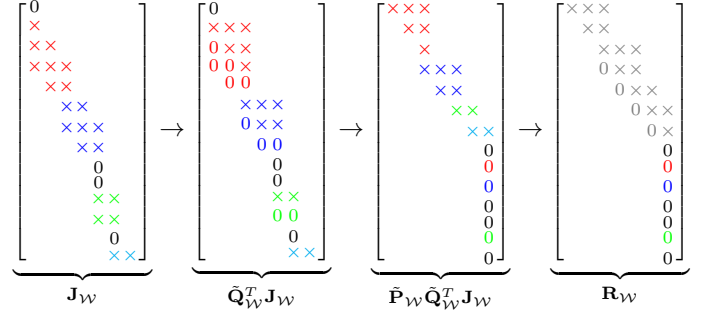

Fig. 12. **QR decomposition for** $n = 15$ **and** $\mathcal{W} = \{1, 2, 6, 9, 10, 11, 13, 14\}$. Cross symbols $\times$ stand for non-zero elements. Left: block structure of $\mathbf{J}_{\mathcal{W}}$, with one color per block. Middle-left: performing QR decomposition for each block. Middle-right: permuting all zero rows to the bottom. Right: completing the QR decomposition.

the triangular factor of the decomposition has zeros on its last line (Figure 12, middle left). Multiplying by a permutation matrix $\mathbf{P}_{\mathcal{W}}$, all zero rows can be moved to the bottom, and we get a quasi-tridiagonal matrix (Figure 12, middle right). The latter can be made triangular with a last tridiagonal QR decomposition in $O(n)$ (Figure 12, right).

G. Search for a feasible initial point

Denoting by $\boldsymbol{l}_Z$ and $\boldsymbol{u}_Z$ the bounds corresponding to $\mathbf{C}_Z$, the set $\mathcal{Z} := \{\boldsymbol{\varphi} \in \mathbb{R}^n, \boldsymbol{l}_Z \leq \mathbf{C}_Z \boldsymbol{\varphi} \leq \boldsymbol{u}_Z\}$ is a zonotope equal to

$$\mathcal{Z} = \mathbf{L}_Z \boldsymbol{l}_Z + \mathbf{L}\,\mathrm{diag}(\boldsymbol{u}_Z - \boldsymbol{l}_Z)\,[0,1]^n \tag{118}$$

where $\mathbf{L}_Z := \mathbf{C}_Z^{-1}$ is the $n \times n$ lower triangular matrix with all coefficients equal to 1. Feasible points for the whole problem are those in $\mathcal{Z}$ such that $\omega_{i,\min}^2 \leq \varphi_n \leq \omega_{i,\max}^2$.

Consider the point:

$$\boldsymbol{\varphi}_i(a) := \mathbf{L}_Z \boldsymbol{l}_Z + \sum a(u_i - l_i) \mathbf{L}_i \tag{119}$$

where $\mathbf{L}_i$ it the ith column of $\mathbf{L}$. This point is in $\mathcal{Z}$ for any value $a \in [0,1]$. Its last component $\varphi_{i,n}(a)$ is an increasing linear function $\varphi_{i,n}(a) = s_l + a s_d$ with $s_l = \sum_j l_j$ and $s_d = \sum_j (u_j - l_j) \geq 0$. Let us denote by a^- and a^+ the two values such that $\varphi_{i,n}(a^-) = \omega_{i,\min}^2$ and $\varphi_{i,n}(a^+) = \omega_{i,\max}^2$. The linear constraints of the capture problem are then feasible *if and only if* $[a^-, a^+] \cap [0,1] \neq \emptyset$. In this case, any a in this intersection yields a feasible point $\boldsymbol{\varphi}_i(a)$ for the problem, for instance the middle value $a_m := \frac{1}{2}(\max(a^-, 0) + \min(a^+, 1))$. We can therefore initialize our SQP with:

$$\boldsymbol{\varphi}_0 = \boldsymbol{\varphi}_i(a_m) \qquad \boldsymbol{\lambda}_0 = 0 \tag{120}$$

which is guaranteed to be a feasible point.

APPENDIX C
EXTERNAL OPTIMIZATION DETAILS

A. Feasibility of outer-optimization problems

There are three ways a bad choice of α can yield an unfeasible capture problem (59):

(a) The bounds of the inequality (59c) are such that $\omega_{i,min} > \omega_{i,max}$. Recall that these bounds represent feasibility of the CoP $\boldsymbol{r}_i$, which involves α by Equation (77).

(b) The intersection between the nonlinear equality constraint (59b) and the polytope (59c)–(59d) is empty. The influence of α on this comes from Equation (76).

(c) The right cylinder given by the linear constraint (59c) does not interesect the zonotope (59d). The role of α in this comes once again from its influence on $\boldsymbol{r}_i$ by Equation (77).

Case (c) can be caught efficiently before solving the capture problem (Appendix B-G). While anticipating (b) is still an open question, case (a) can be avoided altogether thanks to a more careful treatment of CoP inequality constraints.

Recall from Equation (43) that:

$$[\alpha \mathbf{H} \boldsymbol{r}_f^{xy} + (1-\alpha)\boldsymbol{p} - \mathbf{H}\boldsymbol{c}_i^{xy}]\,\omega_i \geq \mathbf{H}\dot{\boldsymbol{c}}_i^{xy} \qquad (121)$$

Let us rewrite this inequality as:

$$(\boldsymbol{u} - \alpha\boldsymbol{v})\omega_i \geq \boldsymbol{w} \qquad (122)$$

To avoid corner cases, let us extend the three vectors $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ with two additional lines:

- *Line 1*: $u = 1$, $v = 0$ and $w = \omega_{i,min}$
- *Line 2*: $u = -1$, $v = 0$ and $w = -\omega_{i,max}$

Next, note that the two sets $\mathcal{A}_{min}(\alpha) := \{i, u_i - \alpha v_i \geq 0\}$ and $\mathcal{A}_{max}(\alpha) := \{i, u_i - \alpha v_i \leq 0\}$ are such that:

$$\omega_{i,min}(\alpha) = \max\left\{\frac{w_i}{u_i - \alpha v_i}, i \in \mathcal{A}_{min}(\alpha)\right\} \qquad (123)$$

$$\omega_{i,max}(\alpha) = \min\left\{\frac{w_i}{u_i - \alpha v_i}, i \in \mathcal{A}_{max}(\alpha)\right\} \qquad (124)$$

Using a technique reminiscent of Fourier-Motzkin elimination, a necessary and sufficient condition for $\omega_{i,min} \leq \omega_{i,max}$ is then that, for all pairs $(i,j) \in \mathcal{A}_{min}(\alpha) \times \mathcal{A}_{max}(\alpha)$,

$$u_i w_j - u_j w_i \leq \alpha(v_i w_j - v_j w_i) \qquad (125)$$

These inequalities are of the form $\widetilde{\boldsymbol{u}}\alpha \geq \widetilde{\boldsymbol{v}}$ and can therefore be reduced, similarly to Equations (44)–(45), into a single interval $[\alpha_{min}, \alpha_{max}]$. In this interval, it is guaranteed that $\omega_{i,min} \leq \omega_{i,max}$ and failure case (a) is avoided.

The subtlety to notice here is that the index sets $\mathcal{A}_\square(\alpha)$ change when α crosses the roots u_i/v_i. (Note that there are few such roots in practice, *e.g.* at most six with rectangular foot soles.) We take this phenomenon into account in the overall Algorithm 3.

Algorithm 3 Computation of α feasibility intervals

Input: vectors $\boldsymbol{u}, \boldsymbol{v}$ and $\boldsymbol{w}$
Output: set $\mathcal{I}$ of feasible intervals $[\alpha_{min}, \alpha_{max}]$

$\quad \mathcal{I} \leftarrow \emptyset$
$\quad \mathcal{R} \leftarrow \{r_j = u_j/v_j | r_j \in (0,1)\}$
$\quad$ **for** (r_j, r_{j+1}) consecutive roots in $\text{SORT}(\mathcal{R})$ **do**
$\quad\quad (\alpha_{min}, \alpha_{max}, \alpha) \leftarrow (r_{2j}, r_{2j+1}, \frac{1}{2}(r_{2j} + r_{2j+1}))$
$\quad\quad$ Compute index sets $\mathcal{A}_{min}(\alpha)$ and $\mathcal{A}_{max}(\alpha)$
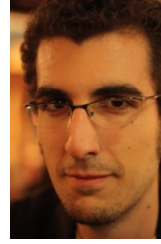$\quad\quad$ Reduce $[\alpha_{min}, \alpha_{max}]$ using (125) with $\mathcal{A}_{min}(\alpha)$, $\mathcal{A}_{max}(\alpha)$
$\quad\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{[\alpha_{min}, \alpha_{max}]\}$
$\quad$ **end for**
$\quad$ **return** $\mathcal{I}$

Stéphane Caron is a CNRS researcher at LIRMM, CNRS–University of Montpellier, France. He received the M.S. degree from the École Normale Supérieure of Paris, and the Ph.D. in 2016 from the University of Tokyo, Japan. He was then involved in the COMANOID project, which developed industrial locomotion and manipulation applications of humanoids that were demonstrated on-site at Airbus. His research interests include whole-body control, force control and balance control for legged robots.

Adrien Escande received the M.S. degree in 2005 from École des Mines de Paris, France, and the Ph.D. degree in 2008 in robotics from Université d'Évry Val-d'Essonne, France, after spending three years at the CNRS–AIST Joint Robotics Laboratory (JRL), Tsukuba, Japan. He then worked as a research scientist in CEA–LIST at Fontenay-aux-Roses, France, until the end of 2012 and is now back at JRL. His research interests include whole-body planning and control for humanoid robots and numerical optimization for robotics.

Leonardo Lanari received the Ph.D. degree in control engineering from Sapienza University of Rome, Italy, in 1992.

He is currently with the Department of Computer, Control and Management Engineering (DIAG), Sapienza University of Rome, as an Associate Professor of automatic control. His research interests are in the general area of planning and control of robotic systems, with an emphasis on humanoid control and robots with elastic joints and links.

Bastien Mallein is *maître de confrences* at the Laboratoire Analyse, Géométrie et Applications (LAGA). He was born in Beauvais (France), and studied Mathematics at the École Normale Supérieure of Paris from 2008 to 2012. He obtained a PhD in Probability from UPMC in 2015, under the supervision of Zhan Shi. In 2016, he was a postdoctoral researcher at the Universität Zürich. He became avare of robotics during a hike in southern Tarn.