Capturability-based Analysis, Optimization and Control of 3D Bipedal Walking

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

Abstract—**Capturability analysis of the linear inverted pendulum model (LIPM) enabled walking over even terrains based on the *capture point*. We generalize this analysis to the inverted pendulum model (IPM) 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 a real-time control loop. We implement this approach as open-source software and demonstrate it in simulations.**

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

I. Introduction

Capturability quantifies the ability of a system to come to a stop at a given location. For a humanoid walking in the linear inverted pendulum mode (LIPM), 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]. More generally, the robot may stop at a given location after N steps, in which case we say that its current state is *N-step capturable* [2]. In recent years, one of the main lines of research in LIPM-based walking has explored the question of walking from N-step capturable states by feedforward planning and feedback control of the capture point [2], [3], [4], [5], [6], [7].

The LIPM owes its tractability to two strong assumptions: the absence of angular-momentum variation around the center of mass, and a holonomic constraint on the height of the center of mass. As a consequence of the latter, a majority of LIPM-based walking controllers assume a flat terrain. Removing this holonomic constraint from the LIPM yields the inverted pendulum model (IPM), for which our understanding is at an earlier stage. Previous study on the capturability of the IPM [8], [9], [10] focused on zero-step capturability for planar motions (sagittal and vertical, non-lateral), a stage that is sufficient for balance control but not for walking.[1]

In the present study, we extend capturability of the IPM to 3D motions and take it to the stage of bipedal walking. This advance is grounded in three developments:

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.

[1] In a preliminary version of this work [11], we generalized the analysis from 2D to 3D balance control. This corresponds to the content of Section III of this manuscript, all other contributions being original material.

- **Analysis:** a mathematical characterization of IPM capturability into *capture inputs* (Section II)
- **Optimization:** a numerical method tailored to the computation of these capture inputs (Section IV)
- **Control:** the combination of zero-step (Section III) and one-step (Section V) capture inputs to realize bipedal walking over uneven terrains (Section VI)

We reserve for Section VII a broader discussion on related and future works.

II. Analysis of the inverted pendulum model

The inverted pendulum model retains the LIPM assumption of no angular-momentum variation around the center of mass (CoM), but strips away the holonomic constraint on CoM height. Its equation of motion is:

$$\ddot{\boldsymbol{c}}(t) = \lambda(t)(\boldsymbol{c}(t) - \boldsymbol{r}(t)) + \boldsymbol{g} \tag{1}$$

where $\boldsymbol{c}$ is the center of mass of the robot and $\boldsymbol{g}$ is the gravity vector, also written $\boldsymbol{g} = -g\boldsymbol{e}_z$ with g the gravitational constant and $\boldsymbol{e}_z$ the vertical of the inertial frame. The two control inputs of the system are the center of pressure (CoP) $\boldsymbol{r}$ and the stiffness λ.

A. 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 a finite number of times, *i.e.* $\mathcal{C}(t) = \mathcal{C}_0$ for $t \in [0, t_1)$, $\mathcal{C}_1$ for $t \in [t_1, t_2)$, ..., and $\mathcal{C}_N$ for $t \geq t_N$. The transition from one contact to the next is called a contact switch. A trajectory whose contact sequence $\mathcal{C}(t)$ contains N 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. 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 $\bar{z}(\boldsymbol{c})$ such that $\boldsymbol{c} - \bar{z}(\boldsymbol{c})\boldsymbol{e}_z$ belongs to the contact plane. It is readily computed as:

$$\bar{z}(\boldsymbol{c}) \stackrel{\text{def}}{=} \frac{(\boldsymbol{c} - \boldsymbol{o}) \cdot \boldsymbol{n}}{(\boldsymbol{e}_z \cdot \boldsymbol{n})} \tag{2}$$

Note how, when walking on a horizontal floor, $\boldsymbol{e}_z$ and $\boldsymbol{n}$ are aligned and $\bar{z}$ 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 $\lambda \in [\lambda_{\min}, \lambda_{\max}]$ to account for two phenomena: a minimum pressure $\lambda_{\min} > 0$ below which contact is uncertain, and a maximum achievable contact force. 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 [12], 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 time 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.

B. N-step capturability

A natural choice of the IPM state consists of its CoM position and velocity $\boldsymbol{x} = \begin{bmatrix} \boldsymbol{c} & \dot{\boldsymbol{c}} \end{bmatrix}$.

Definition 1 (Static equilibrium). *A state is a static equilibrium when its velocity is zero and can remain zero with suitable constant controls.*

Static equilibria, sometimes called *capture states* [1], are the asymptotic targets of capturability analysis, the desired locations that the CoM should converge to. A static equilibrium is characterized by its CoM position $\boldsymbol{c}_f$ and the stationary contact $\mathcal{C}_f$ upon which it is realized. Let us use the shorthand $\bar{z}_f \stackrel{\text{def}}{=} \bar{z}(\boldsymbol{c}_f)$ for the static CoM height above $\mathcal{C}_f$. The only control input $\lambda_f, \boldsymbol{r}_f$ that maintains the system in this equilibrium is such that $\boldsymbol{c}_f = \boldsymbol{r}_f - \boldsymbol{g}/\lambda_f$, that is:

$$\lambda_f(\boldsymbol{c}_f) = \frac{g}{\bar{z}_f} \qquad \boldsymbol{r}_f(\boldsymbol{c}_f) = \boldsymbol{c}_f - \bar{z}_f \boldsymbol{e}_z \qquad (3)$$

Given an N-step contact sequence $\mathcal{C}(t)$ from $\mathcal{C}_0$ to $\mathcal{C}_N$, we say that a state $\boldsymbol{x}_i$ is (N-step) *capturable* when there exists a feasible input function $\lambda(t), \boldsymbol{r}(t)$ such that applying Equation (1) from $\boldsymbol{x}_i$ brings the system asymptotically to an equilibrium $\boldsymbol{x}_f$. We will call such functions *capture inputs* of the capturable state $\boldsymbol{x}_i$, and denote their set by $\mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$. We will call *capture trajectory* the CoM trajectory $\boldsymbol{c}(t)$ resulting from a capture input. In what follows, we will use the subscript $\square_i$ to denote the "initial" or instantaneous state of the system, and the subscript $\square_f$ for its "final" or asymptotic state.

By definition, $\boldsymbol{x}_i$ is capturable if and only if there exists $\boldsymbol{x}_f$ such that $\mathcal{I}_{\boldsymbol{x}, \boldsymbol{x}_f} \neq \emptyset$. The set $\mathcal{I}_{\boldsymbol{x}, \boldsymbol{x}_f}$ is however rather large and contains mathematical oddities. In what follows, we restrict it to inputs that converge asymptotically:

$$\mathcal{I}^c_{\boldsymbol{x}_i, \boldsymbol{x}_f} = \left\{ (\lambda(t), \boldsymbol{r}(t)) \in \mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f} : \begin{array}{l} \lim_{t \to \infty} \lambda(t) = \lambda_f \\ \lim_{t \to \infty} \boldsymbol{r}(t) = \boldsymbol{r}_f \end{array} \right\} \quad (4)$$

The following property shows that this simplification does not affect capturability.

Property 1. *For every pair of states $\boldsymbol{x}_i$ and $\boldsymbol{x}_f$, $\mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$ is non-empty if and only if $\mathcal{I}^c_{\boldsymbol{x}_i, \boldsymbol{x}_f}$ is non-empty.*

In other words, if there exists a capture input $\lambda(t), \boldsymbol{r}(t)$ steering an initial state $\boldsymbol{x}_i$ to a static equilibrium $\boldsymbol{x}_f$, then there exists another input $\lambda^c(t), \boldsymbol{r}^c(t)$ accomplishing the same while also converging. A proof of this property following from controllability of the inverted pendulum is given in Appendix A.

C. Dichotomy of the components of motion

We can rewrite the IPM equation of motion (1) equivalently as a first-order linear time-variant system:

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \ddot{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \lambda\mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{c} \\ \dot{\boldsymbol{c}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{g} - \lambda\boldsymbol{r} \end{bmatrix} \qquad (5)$$

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 $\boldsymbol{r}$. In the present analysis, we focus on open-loop control rather than feedback, thus stressing the linear time-variant rather than nonlinear aspect of Equation (1).

Let us now decouple $\boldsymbol{x}$ into its convergent and divergent components of motion [13]. First, apply the time-varying change of coordinates $\boldsymbol{x} = \mathbf{S}\boldsymbol{z}$ with:

$$\mathbf{S} = \frac{1}{\gamma + \omega} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ -\omega\mathbf{I} & \gamma\mathbf{I} \end{bmatrix} \iff \mathbf{S}^{-1} = \begin{bmatrix} \gamma\mathbf{I} & -\mathbf{I} \\ \omega\mathbf{I} & +\mathbf{I} \end{bmatrix} \quad (6)$$

Here, we assume that $\gamma(t)$ and $\omega(t)$ are *positive* functions of time, of class $\mathcal{C}^1$. We will refer to them as pendulum *dampings* in accordance with their physical unit. The new state vector $\boldsymbol{z}$ consists of two components $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ defined by:

$$\boldsymbol{\zeta} = \gamma\boldsymbol{c} - \dot{\boldsymbol{c}} \qquad (7)$$
$$\boldsymbol{\xi} = \omega\boldsymbol{c} + \dot{\boldsymbol{c}} \qquad (8)$$

and is subject to $\dot{\boldsymbol{z}} = \tilde{\mathbf{A}}\boldsymbol{z} + \tilde{\boldsymbol{b}}$ where:

$$\tilde{\mathbf{A}} = \mathbf{S}^{-1}(\mathbf{A}\mathbf{S} - \dot{\mathbf{S}}) \qquad (9)$$
$$\tilde{\boldsymbol{b}} = \mathbf{S}^{-1}\boldsymbol{b} \qquad (10)$$

While the calculation of $\tilde{\boldsymbol{b}}$ is straightforward, developing that of $\tilde{\mathbf{A}}$ yields:

$$\tilde{\mathbf{A}} = \frac{1}{\gamma + \omega} \begin{bmatrix} (\dot{\gamma} - \gamma\omega - \lambda)\mathbf{I} & (\dot{\gamma} + \gamma^2 - \lambda)\mathbf{I} \\ (\dot{\omega} - \omega^2 + \lambda)\mathbf{I} & (\dot{\omega} + \omega\gamma + \lambda)\mathbf{I} \end{bmatrix} \quad (11)$$

We eliminate non-diagonal terms in this state matrix by imposing Riccati equations on the two damping factors:

$$\dot{\gamma} = \lambda - \gamma^2 \qquad (12)$$
$$\dot{\omega} = \omega^2 - \lambda \qquad (13)$$

This results in the following dynamics for the state $\boldsymbol{z}$:

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{\boldsymbol{\zeta}} \\ \dot{\boldsymbol{\xi}} \end{bmatrix} = \begin{bmatrix} -\gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \omega\mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\zeta} \\ \boldsymbol{\xi} \end{bmatrix} + \begin{bmatrix} \lambda\boldsymbol{r} - \boldsymbol{g} \\ \boldsymbol{g} - \lambda\boldsymbol{r} \end{bmatrix} \quad (14)$$

Therefore, the system can now be 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 (12) and (13).

D. Solutions to the Riccati equations

In this subsection, we verify the existence of damping solutions and exhibit some of their properties that will prove useful to characterize those that don't diverge.

Property 2. *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 (13) *with $\omega(0) = \omega_i$ is non-negative and finite at all times. Moreover, this solution is such that:*

$$\forall t > 0, \quad \omega(t) \in [\sqrt{\lambda_{min}}, \sqrt{\lambda_{max}}] \tag{15}$$

In other words, there is a one-to-one mapping between the stiffness function $\lambda(t)$ and its non-diverging filtered damping $\omega(t)$. This property is cornerstone to our following development, as our solutions to (1), and henceforth N-step capturability conditions, will be written in terms of ω. Using Property 2, these equations can be translated into their counterparts in the physically relevant quantity λ.

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

$$y(t) = \begin{cases} \sqrt{\lambda}\dfrac{e^{-2\sqrt{\lambda}t}\frac{y_0+\sqrt{\lambda}}{y_0-\sqrt{\lambda}}+1}{e^{-2\sqrt{\lambda}t}\frac{y_0+\sqrt{\lambda}}{y_0-\sqrt{\lambda}}-1} & \text{if } y_0 \notin [-\sqrt{\lambda}, \sqrt{\lambda}] \\[4mm] \sqrt{\lambda}\dfrac{e^{-2\sqrt{\lambda}t}\frac{y_0+\sqrt{\lambda}}{y_0-\sqrt{\lambda}}-1}{e^{-2\sqrt{\lambda}t}\frac{y_0+\sqrt{\lambda}}{y_0-\sqrt{\lambda}}+1} & \text{if } y_0 \in (-\sqrt{\lambda}, \sqrt{\lambda}) \\[4mm] y_0 & \text{if } y_0 = \pm\sqrt{\lambda}. \end{cases} \tag{16}$$

The initial condition y_0 settles the behavior of the solution at all times. Define the time $T = \frac{1}{2\sqrt{\lambda}}\log\frac{y_0+\sqrt{\lambda}}{y_0-\sqrt{\lambda}}$, 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 (13). If $\omega(t_0) > \sqrt{\lambda_{max}}$ at some time $t_0 > 0$, then choosing y as the solution to $\dot{y} = y^2 - \lambda_{max}$ with $y(t_0) = \omega(t_0)$, we observe that, as long as $y(t) \le \omega(t)$,

$$\dot{\omega} - \dot{\lambda} = \omega^2 - y^2 \ge 0 \tag{17}$$

Therefore, $\omega - \lambda$ is nondecreasing, and $y(t) \le \omega(t)$ 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 bound from above $\omega(t)$ by a solution to $\dot{y} = y^2 - \lambda_{min}$ that becomes negative in finite time, contradicting once again the hypothesis. The bounds (15) must therefore hold.

We now prove the uniqueness of the non-negative non-exploding solution of (13). 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 *w.l.o.g.* that $\omega_1(0) > \omega_2(0)$. Then, as long as $\omega_1(t) > \omega_2(t)$,

$$\dot{\omega}_1 - \dot{\omega}_2 = (\omega_1 - \omega_2)(\omega_1 + \omega_2) \ge 2\sqrt{\lambda_{min}}(\omega_1 - \omega_2) \tag{18}$$

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

To finally prove the existence of the solution, we observe there exists a unique $\omega(0) \in [\sqrt{\lambda_{min}}, \sqrt{\lambda_{max}}]$ 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 (13) 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). $\square$

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 (12). As a matter of fact, each choice of $\gamma(0) > 0$ yields an admissible solution:

Property 3. *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* (12) *is non-negative and finite at all times. Moreover:*

$$\sqrt{\lambda_{min}} \le \liminf_{t\to\infty} \gamma(t) \le \limsup_{t\to\infty} \gamma(t) \le \sqrt{\lambda_{max}} \tag{19}$$

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 $\dot{\underline{y}} = \lambda_{min} - \underline{y}^2$ and $\dot{\overline{y}} = \lambda_{max} - \overline{y}^2$ with $\underline{y}(0) = \overline{y}(0) = \gamma(0)$. Then, $\underline{y}(t) \le \gamma(t) \le \overline{y}(t)$ 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 $\lim_{t\to\infty} \overline{y}(t) = \sqrt{\lambda_{max}}$. $\square$

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

Corollary 4. *If $\lim_{t\to\infty} \lambda(t) = \lambda_f$, then*

$$\lim_{t\to\infty} \omega(t) = \lim_{t\to\infty} \gamma(t) = \sqrt{\lambda_f}. \tag{20}$$

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 $\forall t > t_0, |\lambda(t) - \lambda_f| < \epsilon$. Next, remark that the time-shifted function $\widetilde{\omega}(t) \stackrel{\text{def}}{=} \omega(t+t_0)$ is solution of the equation $\dot{\widetilde{\omega}}(t) = \widetilde{\omega}(t)^2 - \widetilde{\lambda}$, where $\forall t > 0, \widetilde{\lambda}(t) = \lambda(t + t_0) \in [\lambda_f - \epsilon, \lambda_f + \epsilon]$. Property 2 then shows that $\omega(t + t_0) \in [\sqrt{\lambda_f - \epsilon}, \sqrt{\lambda_f + \epsilon}]$ for all $t > 0$. As a consequence, for any $\epsilon > 0$:

$$\sqrt{\lambda_f - \epsilon} \le \liminf_{t\to\infty} \omega(t) \le \limsup_{t\to\infty} \omega(t) \le \sqrt{\lambda_f + \epsilon} \tag{21}$$

Taking arbitrary small $\epsilon \to 0$, we conclude that the limit of ω as $t \to \infty$ exists and is equal to $\sqrt{\lambda_f}$. $\square$

We conclude from the above properties that the solutions $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ to Equation (14) are well-defined, and we now study their properties.

E. Convergent component of motion

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

$$\dot{\boldsymbol{\zeta}} = -\gamma\boldsymbol{\zeta} + (\lambda\boldsymbol{r} - \boldsymbol{g}) \tag{22}$$

The general solution to this equation is given by:

$$\boldsymbol{\zeta}(t) = \left(\boldsymbol{\zeta}(0) + \int_0^t e^{\Gamma(\tau)}(\lambda(\tau)\boldsymbol{r}(\tau) - \boldsymbol{g})\mathrm{d}\tau\right)e^{-\Gamma(t)} \tag{23}$$

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:

$$\frac{\mathrm{d}e^\Gamma}{\mathrm{d}t} = \gamma e^\Gamma \qquad \frac{\mathrm{d}^2 e^\Gamma}{\mathrm{d}t^2} = (\dot\gamma + \gamma^2)e^\Gamma = \lambda e^\Gamma \qquad (24)$$

The asymptotic behavior of ζ is tied to that of the two inputs λ and r of the inverted pendulum:

Property 5. *Consider an input function $\lambda(t), r(t)$ such that $\lim_{t\to\infty} \lambda(t) = \lambda_\mathrm{f}$ and $\lim_{t\to\infty} r(t) = r_\mathrm{f}$, and let γ denote any solution to (12). Then, the solution ζ of (22) satisfies:*

$$\lim_{t\to\infty} \zeta(t) = \sqrt{\lambda_\mathrm{f}}\left(r_\mathrm{f} - \frac{g}{\lambda_\mathrm{f}}\right) = \sqrt{\lambda_\mathrm{f}}\,c_\mathrm{f} \qquad (25)$$

Proof. By Corollary 4, $\lim_{t\to\infty} \gamma(t) = \sqrt{\lambda_\mathrm{f}}$, therefore its antiderivative satisfies $\lim_{t\to\infty} \Gamma(t)/t = \sqrt{\lambda_\mathrm{f}}$ as well. In particular, $\Gamma(t)$ diverges to ∞, so that:

$$\zeta(t) \underset{t\to\infty}{\sim} e^{-\Gamma(t)} \int_0^t e^{\Gamma(\tau)}(\lambda(\tau)r(\tau) - g)\mathrm{d}\tau \qquad (26)$$

Applying l'Hôpital's rule, we conclude that:

$$\lim_{t\to\infty} \zeta(t) = \lim_{t\to\infty} \frac{\int_0^t e^{\Gamma(\tau)}(\lambda(\tau)r(\tau) - g)\mathrm{d}\tau}{e^{\Gamma(t)}} \qquad (27)$$
$$= \lim_{t\to\infty} \frac{e^{\Gamma(t)}(\lambda(t)r(t) - g)}{\gamma(t)e^{\Gamma(t)}} = \frac{(\lambda_\mathrm{f}r_\mathrm{f} - g)}{\sqrt{\lambda_\mathrm{f}}} \qquad \square$$

This property warrants the name *convergent* component of motion [4] given to ζ: regardless of the state of the system, *any* input function that converges will make ζ converge as well. From a control perspective, this component does not require as much attention as its divergent counterpart.

F. Divergent component of motion

The component ξ corresponding to the damping ω is subject to the differential equation:

$$\dot\xi = \omega\xi + g - \lambda r \qquad (28)$$

The general solution to this equation is given by:

$$\xi(t) = \left(\xi(0) + \int_0^t e^{-\Omega(\tau)}(g - \lambda(\tau)r(\tau))\mathrm{d}\tau\right)e^{\Omega(t)} \qquad (29)$$

where Ω is the antiderivative of ω such that $\Omega(0) = 0$, i.e. $\Omega(t) = \int_0^t \omega(t)\mathrm{d}t$. It satisfies the following two identities

$$\frac{\mathrm{d}e^{-\Omega}}{\mathrm{d}t} = -\omega e^{-\Omega} \qquad \frac{\mathrm{d}^2 e^{-\Omega}}{\mathrm{d}t^2} = (\omega^2 - \dot\omega)e^{-\Omega} = \lambda e^{-\Omega} \qquad (30)$$

Note how $\omega \in [\sqrt{\lambda_{\min}}, \sqrt{\lambda_{\max}}]$ implies that Ω grows at least linearly. Therefore, as long as λ and r remain bounded, the integral $\int_0^\infty e^{-\Omega(\tau)}(g - \lambda(\tau)r(\tau))\mathrm{d}\tau$ is well-defined and finite. In general, set aside the particular condition that we are about to discuss, the function $\xi(t)$ diverges as $t \to \infty$, owning ξ its name of *divergent* component of motion (DCM) [4].[2]

---

[2] More specifically, our analysis considers a *time-varying* divergent component of motion [14]. While previous works such as [4], [6], [14] chose to write their DCMs as positions $c + \dot c/\omega$, 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.

Similarly to what we saw in Property 2 for the damping function $\omega(t)$, a careful choice of the initial condition $\xi(0)$ can guarantee that $\xi(t)$ converges as well. This choice is known as the *boundedness condition* [15]:

Property 6 (Boundedness condition). *Consider an input function $\lambda(t), r(t)$ such that $\lim_{t\to\infty} \lambda(t) = \lambda_\mathrm{f}$ and $\lim_{t\to\infty} r(t) = r_\mathrm{f}$, and let ω denote the non-negative finite solution of (13). Then, there exists a unique ξ_i such that the solution ξ of (28) with $\xi(0) = \xi_\mathrm{i}$ remains finite at all times. Moreover, this initial condition is given by:*

$$\boxed{\xi_\mathrm{i} = \int_0^\infty e^{-\Omega(t)}(\lambda(t)r(t) - g)\mathrm{d}t} \qquad (31)$$

Then, the solution ξ of (28) satisfies:

$$\lim_{t\to\infty} \xi(t) = \sqrt{\lambda_\mathrm{f}}\left(r_\mathrm{f} - \frac{g}{\lambda_\mathrm{f}}\right) = \sqrt{\lambda_\mathrm{f}}\,c_\mathrm{f} \qquad (32)$$

Proof. The proof is very similar to that of Property 5. The solution (29) with $\xi(0) = \xi_\mathrm{i}$ from Equation (31) becomes:

$$\xi(t) = e^{\Omega(t)} \int_t^\infty e^{-\Omega(\tau)}(\lambda(\tau)r(\tau) - g)\mathrm{d}\tau \qquad (33)$$

Applying l'Hôpital's rule, we conclude by Corollary 4 that:

$$\lim_{t\to\infty} \xi(t) = \lim_{t\to\infty} \frac{e^{-\Omega(t)}(\lambda(t)r(t) - g)}{\omega(t)e^{-\Omega(t)}} = \frac{\lambda_\mathrm{f}r_\mathrm{f} - g}{\sqrt{\lambda_\mathrm{f}}} \qquad \square$$

Recalling from Equation (8) that $\xi = \omega c + \dot c$, we note how Equation (31) is another condition (distinct from Proposition 2) that the initial damping value ω_i must satisfy so that $\xi(t)$ remains bounded:

$$\omega_\mathrm{i}c_\mathrm{i} + \dot c_\mathrm{i} = \int_0^\infty e^{-\Omega(t)}(\lambda(t)r(t) - g)\mathrm{d}t \qquad (34)$$

The left-hand side of this equation involves the instantaneous state of the system, whereas its right-hand side involves all its future inputs λ and r.

In the familiar setting where the LIPM is used for constant-height locomotion over a flat floor, $e_z = n$ and λ is kept equal to a constant ω_c^2. Assuming a constant CoP r_c, Equation (36) boils down to:

$$\omega_\mathrm{c}c_\mathrm{i} + \dot c_\mathrm{i} = \int_0^\infty (\omega_\mathrm{c}^2 r_\mathrm{c} - g)e^{-\omega_\mathrm{c}t}\mathrm{d}t = \omega_\mathrm{c}r_\mathrm{c} - \frac{g}{\omega_\mathrm{c}} \qquad (35)$$

Over the x and y coordinates, this equation implies that $r_\mathrm{c}^{xy} = c_\mathrm{i}^{xy} + \dot c_\mathrm{i}^{xy}/\omega_\mathrm{c}$, i.e. the CoP is located at the capture point. Over the z coordinate, it yields $\omega_\mathrm{c} = \sqrt{g/\bar z_\mathrm{i}}$, the well-known expression of the natural frequency of the LIPM.

G. Characterization of capture states and capture inputs

We can now state the main result of this Section: a practical characterization of the set $\mathcal{I}_{x_\mathrm{i}, x_\mathrm{f}}^c$ of capture inputs.

Property 7. *Let $x_\mathrm{i} = \begin{bmatrix} c_\mathrm{i} & \dot c_\mathrm{i} \end{bmatrix}$ denote a capturable state and $x_\mathrm{f} = \begin{bmatrix} c_\mathrm{f} & 0 \end{bmatrix}$ a static equilibrium. Then, $t \mapsto \lambda(t), r(t)$ is a capture input from x_i to x_f if and only if:*

(i) its values $\lambda(t)$ and $r(t)$ are feasible for all $t \geq 0$,
(ii) $\lim_{t\to\infty} \lambda(t) = \lambda_\mathrm{f}(c_\mathrm{f})$ and $\lim_{t\to\infty} r(t) = r_\mathrm{f}(c_\mathrm{f})$,

(iii) it satisfies the boundedness condition:

$$\int_0^\infty (\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})e^{-\Omega(t)}\mathrm{d}t = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \qquad (36)$$

where ω_i is the initial value of the unique bounded solution ω associated with λ (Property 2) and Ω is the antiderivative of ω such that $\Omega(0) = 0$.

This property is at once a characterization of both capturable states and their capture inputs: by Property 1, a state is capturable if and only if there exists a target equilibrium $\boldsymbol{x}_f$ and an input function that satisfies (i)–(iii).

Proof of the $\Rightarrow$ 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 (1). 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 5, this in turns implies that its the divergent component $\boldsymbol{\xi}(t)$ is bounded, and must therefore satisfy Equation (36) by Property 6. 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 5 and 6, the two components converge to:

$$\lim_{t\to\infty} \boldsymbol{\zeta}(t) = \lim_{t\to\infty} \boldsymbol{\xi}(t) = \sqrt{\lambda_f}\left(\boldsymbol{r}_f - \frac{\boldsymbol{g}}{\lambda_f}\right) \qquad (37)$$

Recalling from Corollary 4 that γ and ω converge to $\sqrt{\lambda_f}$, we can take the limit in the mapping (6)–(8):

$$\lim_{t\to\infty} \boldsymbol{c}(t) = \lim_{t\to\infty} \frac{\boldsymbol{\zeta}(t) + \boldsymbol{\xi}(t)}{\gamma(t) + \omega(t)} = \boldsymbol{r}_f - \frac{\boldsymbol{g}}{\lambda_f} = \boldsymbol{c}_f \qquad (38)$$

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

Proof of the $\Leftarrow$ implication. Reciprocally, assuming (i)–(iii), Equation (37) holds again by Properties 5–6 and Corollary 4. Furthermore,

$$\lim_{t\to\infty} \dot{\boldsymbol{c}}(t) = \lim_{t\to\infty} \frac{-\omega(t)\boldsymbol{\zeta}(t) + \gamma(t)\boldsymbol{\xi}(t)}{\gamma(t) + \omega(t)} = \boldsymbol{0} \qquad (39)$$

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

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 $\widetilde{\boldsymbol{\xi}} \stackrel{\text{def}}{=} \omega(\boldsymbol{c} - \boldsymbol{r}) + \dot{\boldsymbol{c}} - \dot{\boldsymbol{r}}$ yielding a boundedness condition written:

$$\int_0^\infty (\ddot{\boldsymbol{r}}(\tau) - \boldsymbol{g})e^{-\Omega(\tau)}\mathrm{d}\tau = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \qquad (40)$$

This condition is the same as (36), which can be seen by applying a double integration by parts using Equation (30).

H. Timeless parameterization

The infinite-time integral from condition (iii) of Property 7 raises computational questions. We can however reduce it to a finite integral by defining the adimensional quantity $s(t) = e^{-\Omega(t)}$. This new variable ranges from $s = 1$ when $t = 0$ to $s = 0$ when $t \to \infty$. Its time derivatives are:

$$\dot{s}(t) = -\omega(t)s(t) \qquad \ddot{s}(t) = \lambda(t)s(t) \qquad (41)$$

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 [16]. Let us denote by $\square'$ derivation with respect to s, as opposed to $\dot{\square}$ for derivation with respect to t. The Riccati equation (13) of ω becomes:

$$\lambda = \omega^2 - \dot{\omega} = \omega^2 - \dot{s}\omega' = \omega(\omega + s\omega') = \omega(s\omega)' \qquad (42)$$

Injecting this expression into the time integral (36) of the boundedness condition yields:

$$\int_0^\infty (\lambda(t)\boldsymbol{r}(t) - \boldsymbol{g})s(t)\mathrm{d}t = \int_0^1 (\omega(s\omega)'\boldsymbol{r}(s) - \boldsymbol{g})\frac{\mathrm{d}s}{\omega} \qquad (43)$$

We can now rephrase our characterization of capture inputs as functions of s rather than time t.

Property 8. *Let $\boldsymbol{x}_i = \begin{bmatrix} \boldsymbol{c}_i & \dot{\boldsymbol{c}}_i \end{bmatrix}$ denote a capturable state and $\boldsymbol{x}_f = \begin{bmatrix} \boldsymbol{c}_f & \dot{\boldsymbol{c}}_f \end{bmatrix}$ 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:

$$\int_0^1 \boldsymbol{r}(s)(s\omega)'\mathrm{d}s - \boldsymbol{g}\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \qquad (44)$$

where ω_i denotes the initial value (at $s = 1$) of the unique bounded solution ω associated with λ (Property 2).

This characterization lays the foundation upon which we can now compute capture inputs in practical situations.

III. ZERO-STEP CAPTURABILITY OF THE 3D IPM

Let us consider *zero-step* capturability, where the contact $\mathcal{C}$ is already established, and used to absorb the momentum of the initial state $\boldsymbol{x}_i$ until a static equilibrium $\boldsymbol{x}_f$ is reached. This level of capturability enables push recovery [1], [17], [10] up to post-impact fall recovery in worst-case scenarios [18], [19].

Taking the dot product of the boundedness condition (44) with the normal $\boldsymbol{n}$ of the contact area $\mathcal{C}$ yields:

$$\int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \frac{\omega_i \bar{z}_i + \dot{\bar{z}}_i}{g} \qquad (45)$$

where $\bar{z}_i \stackrel{\text{def}}{=} \bar{z}(\boldsymbol{c}_i)$ is the initial CoM height. Meanwhile, the horizontal components of the condition are:

$$\int_0^1 \boldsymbol{r}^{xy}(s)(s\omega)'\mathrm{d}s = \omega_i \boldsymbol{c}_i^{xy} + \dot{\boldsymbol{c}}_i^{xy} \qquad (46)$$

Condition (45) involves gravity but does not depend on the CoP input, while its counterpart (46) includes the CoP but does not depend on gravity.

A. Example of the fixed-CoP strategy

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.[3]

---

[3] *Short proof:* take the cross-product of Equation (1) with $\ddot{\boldsymbol{c}}$, then the scalar product of the result with $\boldsymbol{g}$ to obtain $((\boldsymbol{c} - \boldsymbol{r}) \times \ddot{\boldsymbol{c}}) \cdot \boldsymbol{g} = 0$. Conclude by noting that this expression is the time derivative of $((\boldsymbol{c} - \boldsymbol{r}) \times \dot{\boldsymbol{c}}) \cdot \boldsymbol{g}$.

There are then 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 $\dot{c} = 0$. 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.

For now, let us consider a stationary CoP input $r(s) = r_f(c_f)$ and assume that $G = 0$, *i.e.* the three vectors $\dot{c}_i$, $c_i - r_f$ and g are coplanar. The CoP component (46) of the boundedness condition becomes:

$$r_f^{xy} = c_f^{xy} = c_i^{xy} + \frac{\dot{c}_i^{xy}}{\omega_i} \tag{47}$$

where we recognize the well-know expression of the capture point, save for the fact that ω_i is now the initial value of a time-varying function. This value is given by:

$$\omega_i = \frac{\dot{c}_i^x}{c_f^x - c_i^x} = \frac{\dot{c}_i^y}{c_f^y - c_i^y} \tag{48}$$

The right-hand side of Equation (45) is fully determined by the initial state x_i and desired equilibrium c_f. In these conditions, the conditions for 3D capturability given by Property 8 boil down to 1D conditions over λ:

(i) $\forall s \in [0, 1], \lambda(s) \in [\lambda_{min}, \lambda_{max}]$
(ii) $\lim_{s \to 0} \lambda(s) = \lambda_f(c_f)$
(iii) $\omega(1) = \omega_i$ and $\int_0^1 \frac{ds}{\omega(s)} = \frac{1}{g}(\omega_i \bar{z}_i + \dot{z}_i)$

where ω is the bounded solution to the Riccati equation (42) $\lambda = \omega(s\omega)'$. We call this problem the *1D capture problem*. Its main difficulty lies in the resolution of the integral equality constraint over the inverse of ω.

B. Formulation of a first optimization problem

Let us partition the interval $[0, 1]$ into n fixed segments $0 = s_0 < s_1 < \ldots < s_{n-1} < s_n = 1$. We compute solutions to the 1D problem *(i)–(iii)* where $\lambda(s)$ is piecewise constant over this partition, that is, $\forall s \in (s_j, s_{j+1}], \lambda(s) = \lambda_j$. Define:

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

The quantity φ represents a squared velocity (41) and is commonly considered in time-optimal retiming (see *e.g.* [20]). Remarking that $\varphi' = 2s\lambda$ from the Riccati equation (42), we can directly 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 $\varphi_j \stackrel{\text{def}}{=} \varphi(s_j)$. The values $\lambda(s)$ and $\omega(s)$ for $s \in (s_j, s_{j+1}]$ can be computed back from φ using the two equations above:

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

In particular, $\lim_{s \to 0} \lambda(s) = \lambda_1 = \varphi_1/\delta_0$. Convergence to λ_f thus requires that $\varphi_1 = \delta_0 \lambda_f$. In this setting, λ_f is a free parameter, *i.e.* it is not determined by the boundedness condition. We set it as $\lambda_f = g/\bar{z}_f$ where $\bar{z}_f$ corresponds to the CoM height of the robot at rest with an extended leg (*e.g.* $\bar{z}_f = 0.8$ m is a suitable value for the HRP-4 model).

We can now calculate the left-hand side of the gravity component of the boundedness condition (45):

$$\int_0^1 \frac{ds}{\omega(s)} = \sum_{j=0}^{n-1} \int_{s_j}^{s_{j+1}} \frac{s \, ds}{\sqrt{\varphi_j + \lambda_j(s^2 - s_j^2)}} \tag{52}$$

$$= \sum_{j=0}^{n-1} \int_0^{\delta_j} \frac{dv}{2\sqrt{\varphi_j + \lambda_j v}} \tag{53}$$

$$= \sum_{j=0}^{n-1} \frac{1}{\lambda_j}\left[\sqrt{\varphi_j + \lambda_j \delta_j} - \sqrt{\varphi_j}\right] \tag{54}$$

$$= \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} \tag{55}$$

Altogether, these expressions allow us to formulate all conditions *(i)–(iii)* as equality and inequality constraints over the vector $\varphi \stackrel{\text{def}}{=} [\varphi_1 \ldots \varphi_n]$. Note that this vector starts from φ_1, as $\varphi_0 = 0$ by definition, and that $\omega_i = \sqrt{\phi_n}$ from Equation (49). Adding a regularizing cost function over variations of λ to these constraints yields the following optimization problem:

$$\underset{\varphi \in \mathbb{R}^n}{\text{minimize}} \sum_{j=1}^{n-1}\left[\frac{\varphi_{j+1} - \varphi_j}{\delta_j} - \frac{\varphi_j - \varphi_{j-1}}{\delta_{j-1}}\right]^2 \tag{56a}$$

$$\text{subject to} \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} - \frac{\bar{z}_i\sqrt{\varphi_n} + \dot{z}_i}{g} = 0 \tag{56b}$$

$$\varphi_n = \omega_i^2 \tag{56c}$$

$$\forall j < n, \ \lambda_{min}\delta_j \leq \varphi_{j+1} - \varphi_j \leq \lambda_{max}\delta_j \tag{56d}$$

$$\varphi_1 = \delta_0 \lambda_f = \delta_0 g/\bar{z}_f \tag{56e}$$

From Property 8, this problem produces capture inputs that steer the system to the desired static equilibrium at height $\bar{z}_f$ above the fixed CoP. It is "almost" a quadratic program: it has a quadratic cost function and linear constraints, except for Equation (56b) which is a nonlinear equality constraint.

C. Time-varying CoP strategy

Let us now turn to time-varying centers of pressure. Equation (46) admits a simple solution when the CoP input follows a straight line from r_i to r_f:

$$r(s) = r_f + (r_i - r_f)f(s\omega) \tag{57}$$

where f denotes 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 its supporting line,
- f *is integrable*: let F denote its antiderivative such that $F(0) = 0$. It is positive by monotonicity of f.
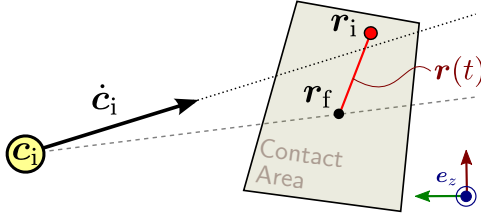
Fig. 1. **Linear time-varying CoP trajectory.** Variations of the center of pressure inside the contact area allow the robot to adjust its divergent component of motion without re-stepping.

Figure 1 depicts the linear CoP trajectory resulting from this choice. Intuitively, the initial CoP r_i 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 r_f, similarly to the behavior observed in the LIPM with linear capture-point feedback control [3], [5], [6].

We should now ensure that our choice of CoP input satisfies all conditions (*i*)–(*iii*) from Property 8. The convergence condition (*ii*) is satisfied from the properties we impose on f. Also, we choose the target CoP r_f as the center of the contact area, *i.e.* the point furthest away from inequality constraints.

To fulfill the boundedness condition (*iii*), the initial CoP r_i must satisfy Equation (46). Its horizontal coordinates are therefore:

$$r_i^{xy} = r_f^{xy} + \frac{\omega_i(c_i^{xy} - r_f^{xy}) + \dot{c}_i^{xy}}{F(\omega_i)} \tag{58}$$

The current state and target equilibrium being given, the only free variable on the right-hand side is ω_i. The complete world coordinates of r_i are readily available from r_i^{xy} by vertical projection:

$$r_i = r_i^{xy} - \bar{z}(r_i^{xy})e_z \tag{59}$$

To fulfill the feasibility condition (*i*), we need only ensure that both ends r_i and r_f of the CoP segment belong to the polygonal contact area. The latter does by definition. For the former, the constraint that r_i belongs to the contact polygon can be described in halfspace-representation by a matrix-vector inequality $\mathbf{F}r_i^{xy} \leq p$, with $\mathbf{F}$ an $m \times 2$ matrix and p an m-dimensional vector. Representing the contact polygon in the horizontal plane rather than in the contact frame makes it more convenient to work with the boundedness condition (58), and is possible thanks to Equation (59). For example, a rectangular contact area written in the contact frame (t, b, n) as:

$$\pm t \cdot (r_i - o) \leq X \tag{60}$$
$$\pm b \cdot (r_i - o) \leq Y \tag{61}$$

can be reformulated equivalently in the horizontal plane:

$$\pm (b \times e_z)(r_i^{xy} - o^{xy}) \leq X(e_z \cdot n) \tag{62}$$
$$\pm (t \times e_z)(r_i^{xy} - o^{xy}) \leq Y(e_z \cdot n) \tag{63}$$

Injecting Equation (58) into inequalities $\mathbf{F}r_i^{xy} \leq p$ yields:

$$[\mathbf{F}(c_i^{xy} - r_f^{xy})]\omega_i + [\mathbf{F}r_f^{xy} - p]F(\omega_i) \leq -\mathbf{F}\dot{c}_i \tag{64}$$

Capture Problem

Parameters:

- Feasibility bounds $(\lambda_{min}, \lambda_{max})$ and $(\omega_{i,min}, \omega_{i,max})$
- Initial height $\bar{z}_i$, its derivative $\dot{\bar{z}}_i$, and target height $\bar{z}_f$
- Number of steps n and discretization steps $\delta_1, \ldots, \delta_n$

$$\underset{\varphi \in \mathbb{R}^n}{\text{minimize}} \sum_{j=1}^{n-1} \left[\frac{\varphi_{j+1} - \varphi_j}{\delta_j} - \frac{\varphi_j - \varphi_{j-1}}{\delta_{j-1}} \right]^2 \tag{70a}$$

$$\text{subject to} \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} - \frac{\bar{z}_i\sqrt{\varphi_n} + \dot{\bar{z}}_i}{g} = 0 \tag{70b}$$

$$\omega_{i,min}^2 \leq \varphi_n \leq \omega_{i,max}^2 \tag{70c}$$

$$\forall j < n, \ \lambda_{min}\delta_j \leq \varphi_{j+1} - \varphi_j \leq \lambda_{max}\delta_j \tag{70d}$$

$$\varphi_1 = \delta_0\lambda_f = \delta_0 g/\bar{z}_f \tag{70e}$$

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

$$f(s\omega) = \left(\frac{s\omega}{\omega_i}\right)^{\frac{\alpha}{1-\alpha}} \implies F(\omega_i) = (1 - \alpha)\omega_i \tag{65}$$

With this choice, the horizontal coordinates of r_i become:

$$r_i^{xy} = \frac{1}{1 - \alpha}\left[c_i^{xy} + \frac{\dot{c}_i^{xy}}{\omega_i} - \alpha r_f^{xy}\right] \tag{66}$$

This choice has the advantage of making inequality (64) linear:

$$[\alpha\mathbf{F}r_f^{xy} + (1 - \alpha)p - \mathbf{F}c_i^{xy}]\omega_i \geq \mathbf{F}\dot{c}_i \tag{67}$$

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:

$$\omega_{i,min} = \max\left(\sqrt{\lambda_{min}}, \{v_j/u_j, u_j > 0\}\right) \tag{68}$$

$$\omega_{i,max} = \min\left(\sqrt{\lambda_{max}}, \{v_j/u_j, u_j < 0\}\right) \tag{69}$$

We can now update our optimization problem to this more general setting. Inequalities $0 < \omega_{i,min} \leq \omega_i \leq \omega_{i,max}$ yield linear inequality constraints on the last optimization variable, written as $\omega_{i,min}^2 \leq \varphi_n \leq \omega_{i,max}^2$. Wrapping up this development, we obtain the *capture problem* (70). This problem embodies all conditions (*i*)–(*iii*) from Property 8 and therefore produces capture inputs that steer the system to the desired static equilibrium $c_f = r_f + \bar{z}_f e_z$.

D. Computation and behavior of CoM capture trajectories

Computing the capture trajectory $c(t)$ corresponding to the solution φ of a capture problem can be done by backward recursion. This operation won't be necessary in our model predictive controller, which only requires the initial control inputs λ_i and r_i computed from φ via Equations (51) and (66). However, it shows how to recover time functions from their counterparts in s.

First, let us calculate the times $t_j = t(s_j)$ where the stiffness λ switches from one value to the next. Recall how
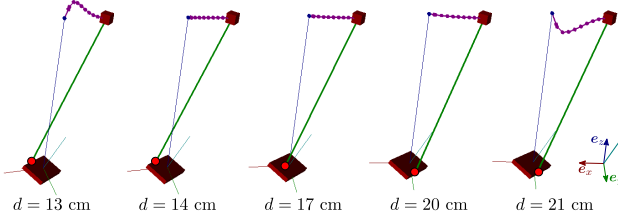
Fig. 2. **Zero-step capture trajectories for different contact locations.** The horizontal distance d from CoM to contact varies while the initial CoM position and velocity are kept constant. Red discs indicate the initial CoP location $\boldsymbol{r}_\mathrm{i}$. The regularization cost keeps the trajectory as close as possible to a LIPM via CoP variations. When there is no linear solution, height variations are resorted to for additional acceleration or braking.

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 (41) with boundary condition $s(t_j) = s_j$ to obtain the next switch time $t_j = t(s_j)$ as:

$$t(s_j) = t_{j+1} + \frac{1}{\sqrt{\lambda_j}} \log\left(\frac{\sqrt{\varphi_{j+1}} + \sqrt{\lambda_j} s_{j+1}}{\sqrt{\varphi_j} + \sqrt{\lambda_j} s_j}\right) \quad (71)$$

This formula allows us to compute iteratively the time partition $0 = t_n < t_{n-1} < \ldots < t_1 < \infty$ of stiffness switches. Finally, we can compute the CoM trajectory iteratively as well by applying the constant-stiffness solution to Equation (1) on each consecutive segment $[t_{j+1}, t_j]$.

As an initial evaluation, we solve the capture problem (70) using the off-the-shelf nonlinear interior-point solver IPOPT [21]. Figure 2 depicts the CoM capture trajectories obtained by applying the above algorithm to optimal capture solutions $\boldsymbol{\varphi}^*$. Observe how their behavior is hierarchical: CoP variations are used first to keep the trajectory as linear as possible; then, additional CoM height variations are resorted to when CoP bounds are saturated.

Starting from various capturable states and contact locations, we observed that the off-the-shelf solver finds solutions in around 1 to 3 ms on a commercial laptop computer (a deeper performance investigation will be carried out in Section IV-F), which is ten times faster than the performance obtained by the same solver on a direct transcription of centroidal dynamics [12]. This observation warrants further investigation into this new optimization problem.

IV. OPTIMIZATION OF CAPTURE PROBLEMS

The ability to compute capture inputs fast is crucial to make the 3D IPM generalization as "cheap" as possible compared to its LIPM counterpart. While the capture problem (70) belongs to the general class of nonconvex optimization, computation times obtained with the off-the-shelf solver suggest that they are actually easier to solve than generic nonconvex problems. We will see that that there are indeed structural properties that we can leverage into a more efficient resolution scheme.

This section presents a dedicated solver for the capture problem (70) which is two to three orders of magnitude faster than a state-of-the-art general-purpose solver. Readers more interested in walking control can take a look at the timings in IV-F and skip the rest of this Section.

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 (70) has a linear least squares cost, linear constraints and *one 1-dimensional nonlinear equality constraint*. Additionally, the cost and linear constraints have specific structures that we can leverage to implement an efficient solver.

Let us start with some notations. The objective (56a) can be written $\|J\boldsymbol{\varphi}\|^2$, where $\boldsymbol{\varphi}$ is the vector of optimization variables φ_1 to φ_N. Equation (70b) rewrites to $b(\boldsymbol{\varphi}) = 0$ with:

$$b(\boldsymbol{\varphi}) = \sum_{j=0}^{n-1} \frac{\delta_j}{\sqrt{\varphi_{j+1}} + \sqrt{\varphi_j}} - \frac{\bar{z}_\mathrm{i}\sqrt{\varphi_n} + \dot{\bar{z}}_\mathrm{i}}{g} \quad (72)$$

where $\bar{z}_\mathrm{i} > 0$ and $\dot{\bar{z}}_\mathrm{i} \in \mathbb{R}$ are problem parameters (*i.e.* constant during the optimization). In both problems, linear constraints have the form:

$$\boldsymbol{l} \leq \mathbf{C}\boldsymbol{\varphi} \leq \boldsymbol{u} \quad (73)$$

where $\boldsymbol{l} \in \mathbb{R}^{n+1}$ and $\boldsymbol{u} \in \mathbb{R}^{n+1}$ are also problem-dependent vectors (set $l_j = u_j$ to specify an equality). The cost matrix $\mathbf{J}$ is the $(n-1) \times n$ matrix given by:

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

where $d_j = \delta_j^{-1}$. Meanwhile, $\mathbf{C}$ is the $(n+1) \times n$ matrix:

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

where $\boldsymbol{e}_n$ is the last column of the $n \times n$ identity matrix.

Solutions to (70) can be approximated by solving:

$$\underset{\boldsymbol{\varphi} \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{J}\boldsymbol{\varphi}\|^2 + \frac{\mu^2}{2}\|b(\boldsymbol{\varphi})\|^2 \quad (74\text{a})$$

$$\text{subject to} \quad \boldsymbol{l} \leq \mathbf{C}\boldsymbol{\varphi} \leq \boldsymbol{u} \quad (74\text{b})$$

which presents the advantage of having only linear constraints, and whose solution tends to the original solution as μ goes to infinity. This problem can be solved very efficiently.

B. Applying an SQP approach

We apply the SQP method (Algorithm 4) to problem (74). We write $\boldsymbol{j} \stackrel{\text{def}}{=} \nabla_{\boldsymbol{\varphi}} b$ the gradient of the nonlinear constraint, and index by $\square_k$ the value of any quantity at iteration k.

Denoting by $f(\boldsymbol{\varphi})$ the objective (74a), the Lagrangian of the problem 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. The Hessian matrix is then:

$$\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \mathcal{L}_k = \mathbf{J}^T\mathbf{J} + \mu^2 \boldsymbol{j}_k \boldsymbol{j}_k^T + b_k^2 \nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \qquad (75)$$

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 that is particularly well-suited to our case: when $b = 0$, *i.e.* when the boundedness condition is satisfied, $\nabla_{\boldsymbol{\varphi}\boldsymbol{\varphi}}^2 \mathcal{L}$ is exactly equal to $\mathbf{J}^T\mathbf{J} + \mu^2 \boldsymbol{j}\boldsymbol{j}^T$. Under this approximation, the problem corresponding to one iteration of the SQP method is:

$$\underset{\boldsymbol{p} \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{J}\boldsymbol{p} + \mathbf{J}\boldsymbol{\varphi}_k\|^2 + \frac{\mu^2}{2}\|\boldsymbol{j}_k^T\boldsymbol{p} + b_k\|^2 \qquad (76a)$$

$$\text{subject to } \boldsymbol{l}_k' \leq \mathbf{C}\boldsymbol{p} \leq \boldsymbol{u}_k' \qquad (76b)$$

with $\boldsymbol{l}_k' \overset{\text{def}}{=} \boldsymbol{l} - \mathbf{C}\boldsymbol{\varphi}_k$ and $\boldsymbol{u}_k' \overset{\text{def}}{=} \boldsymbol{u} - \mathbf{C}\boldsymbol{\varphi}_k$. If $\boldsymbol{\varphi}_k$ satisfies constraint (74b), then $\boldsymbol{p} = 0$ is a feasible point for (76).

C. Solving the least squares sub-problem

The problem (76) solved at each SQP iteration is a linear least squares with inequality constraints (LSI), a particular case of QP, that we can solve using the active-set method (Algorithm 3). 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 Algorithm 3 solves in our case (see problem (108))

$$\underset{\boldsymbol{d} \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{J}\boldsymbol{d} + \mathbf{J}(\boldsymbol{p}_j + \boldsymbol{\varphi}_k)\|^2 + \frac{\mu^2}{2}\|\boldsymbol{j}^T\boldsymbol{d} + \boldsymbol{j}^T\boldsymbol{p}_j + b_k\|^2 \qquad (77a)$$

$$\text{subject to } \mathbf{C}_{\mathcal{W}_j}\boldsymbol{d} = 0 \qquad (77b)$$

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

This can be solved in two steps (nullspace approach). First, compute a matrix $\mathbf{N}_{\mathcal{W}} \in \mathbb{R}^{n-r}$ 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 an 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:

$$\underset{\boldsymbol{z} \in \mathbb{R}^{n-r}}{\text{minimize}} \quad \frac{1}{2}\left\|\begin{bmatrix}\mu\boldsymbol{j}^T \\ \mathbf{J}\end{bmatrix}\mathbf{N}_{\mathcal{W}}\boldsymbol{z} + \begin{bmatrix}\mu(\boldsymbol{j}^T\boldsymbol{p}_j + f) \\ \mathbf{J}(\boldsymbol{p}_j + \boldsymbol{\varphi})\end{bmatrix}\right\|^2 \qquad (78)$$

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 [22, Chapter 10]. Both of these steps can be significantly tailored to the case of capture problems.

D. 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 $O(n^3)$ [23], at least for the

first iteration of each LSI.[4] We show here how to reduce this complexity to at most $O(n^2)$ for capture problems.

Consider the active set $\mathcal{W}$. 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 $n_{\mathcal{W}} \overset{\text{def}}{=} \sum_k a_k$. 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:

$$\mathbf{C}_{\mathcal{W}} = \begin{bmatrix} \mathbf{C}_0 & & & & & \\ & \mathbf{0}_{a_1, j_1-1} & \mathbf{C}_1 & & & \\ & & & \mathbf{0}_{a_2, j_2-1} & \mathbf{C}_2 & \\ & & & & \ddots & \\ & & & & & \mathbf{0}_{a_p, j_p-1} & \mathbf{C}_p \end{bmatrix} \qquad (79)$$

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:

$$\mathbf{C}_0 = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \quad \mathbf{C}_k = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \quad \mathbf{C}_p = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & 1 \end{bmatrix} \qquad (80)$$

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

$$\mathbf{N}_{\mathcal{W}} = \begin{bmatrix} \mathbf{0}_{a_0, i_1-1} & & & & & \\ \mathbf{I}_{i_1-1} & & & & & \\ & \mathbf{1}_{a_1+1} & & & & \\ & & \mathbf{I}_{i_2-1} & & & \\ & & & \mathbf{1}_{a_2+1} & & \\ & & & & \ddots & \\ & & & & & \mathbf{I}_{i_p-1} \\ & & & & & \mathbf{0}_{a_p, i_p-1} \end{bmatrix} \qquad (81)$$

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 $O(n)$.

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

---

[4]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 $O(n^2)$.
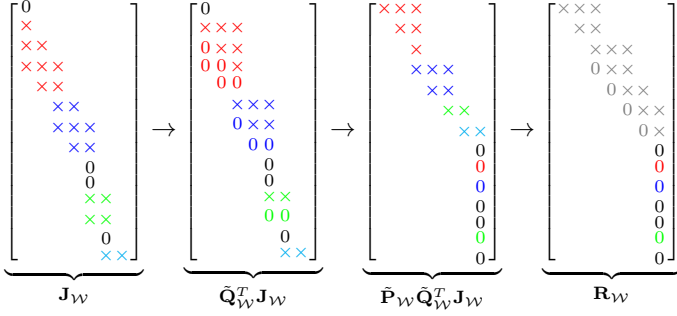
Fig. 3. **QR decomposition for** $n = 15$ **and** $\mathcal{W} = \{1, 2, 6, 9, 10, 11, 13, 14\}$. Cross symbols $\times$ stand for non-zero elements. Left: block structure of $\mathbf{J}_{\mathcal{W}}$, with one color per block. Middle-left: performing QR decomposition for each block. Middle-right: permuting all zero rows to the bottom. Right: completing the QR decomposition.

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

$$\mathbf{C}_{\mathcal{W}}^{\dagger} = \begin{bmatrix} \mathbf{C}_0^{-1} & & & & & & \\ & \mathbf{0}_{i_1-1,a_1} & & & & & \\ & & \mathbf{C}_1^{\dagger} & & & & \\ & & & \mathbf{0}_{i_2-1,a_2} & & & \\ & & & & \mathbf{C}_2^{\dagger} & & \\ & & & & & \ddots & \\ & & & & & & \mathbf{0}_{i_p-1,a_p} \\ & & & & & & \mathbf{C}_p^{-1} \end{bmatrix} \tag{82}$$

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

$$\mathbf{C}_0^{-1} = \begin{bmatrix} 1 & & \\ \vdots & \ddots & \\ 1 & \cdots & 1 \end{bmatrix}, \quad \mathbf{C}_p^{-1} = \begin{bmatrix} -1 & \cdots & -1 & 1 \\ & \ddots & \vdots & 1 \\ & & -1 & 1 \\ & & & 1 \end{bmatrix} \tag{83}$$

$$\mathbf{C}_k^{\dagger} = \frac{1}{a_k+1} \begin{bmatrix} -a_k & -(a_k-1) & -(a_k-2) & \cdots & -1 \\ 1 & -(a_k-1) & -(a_k-2) & \cdots & -1 \\ 1 & 2 & -(a_k-2) & \cdots & -1 \\ 1 & 2 & 3 & \ddots & \vdots \\ \vdots & & & \ddots & -1 \\ 1 & 2 & 3 & \cdots & a_k \end{bmatrix} \tag{84}$$

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

The QR decomposition of $\mathbf{T}$ is performed in two steps: first the QR decomposition $\mathbf{J}_{\mathcal{W}} = \mathbf{Q}_{\mathcal{W}} \mathbf{R}_{\mathcal{W}}$ of $\mathbf{J}_{\mathcal{W}} \stackrel{\text{def}}{=} \mathbf{J} \mathbf{N}_{\mathcal{W}}$, 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 the sought after $\mathbf{T}$ by:

$$\begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{J} \mathbf{N}_{\mathcal{W}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{Q}_{\mathcal{W}} \end{bmatrix} \begin{bmatrix} \mu \boldsymbol{j}^T \mathbf{N}_{\mathcal{W}} \\ \mathbf{R}_{\mathcal{W}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{Q}_{\mathcal{W}} \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1^T \\ \mathbf{Q}_2 \end{bmatrix} \mathbf{R}$$
$$= \begin{bmatrix} \boldsymbol{q}_1^T \\ \mathbf{Q}_{\mathcal{W}} \mathbf{Q}_2 \end{bmatrix} \mathbf{R} \tag{85}$$

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 $O(n^2)$ [23, 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 3. More details can be found in [24]. 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 is aligned with the first column of the second (Figure 3, 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, have a rank equal to their row size minus one, so that the triangular factor of the decomposition has zeros on its last line (Figure 3, 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 3, middle right). The latter can be made triangular with a last tridiagonal QR decomposition in $O(n)$ (Figure 3, right).

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, here we can leverage the geometry of our constraints to get such a pair in $O(n)$. Noting $\boldsymbol{l}_Z$ and $\boldsymbol{u}_Z$ the bounds corresponding to $\mathbf{C}_Z$, the set $\mathcal{Z} \stackrel{\text{def}}{=} \{\boldsymbol{\varphi} \in \mathbb{R}^n, \boldsymbol{l}_Z \leq \mathbf{C}_Z \boldsymbol{\varphi} \leq \boldsymbol{u}_Z\}$ is a zonotope equal to $\mathbf{L}_Z \boldsymbol{l}_Z + \mathbf{L} \text{diag}(\boldsymbol{u}_Z - \boldsymbol{l}_Z) [0,1]^n$ where $\mathbf{L}_Z \stackrel{\text{def}}{=} \mathbf{C}_Z^{-1}$ 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:

$$\boldsymbol{\varphi}(a) \stackrel{\text{def}}{=} \mathbf{L} \boldsymbol{l}_Z + \sum a(u_i - l_i) \mathbf{L}_i \tag{86}$$

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 $\varphi_n(a)$ is an increasing linear function $\varphi_n(a) = s_l + a s_d$ 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 $\varphi_n(a^-) = \omega_{i,\min}^2$ and $\varphi_n(a^+) = \omega_{i,\max}^2$. 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 $\boldsymbol{\varphi}(a)$ for the problem, *e.g.* the middle value $a_m \stackrel{\text{def}}{=} \frac{1}{2}(\max(a^-, 0) + \min(a^+, 1))$. We finally initialize our SQP with $\boldsymbol{\varphi}_0 = \boldsymbol{\varphi}(a_m)$ and $\boldsymbol{\lambda}_0 = 0$.

E. 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 [22]. 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 (74) 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

TABLE I
COMPUTATION TIMES OVER 20000 SAMPLE PROBLEMS FROM A WALKING
SIMULATION. AVERAGES AND STANDARD DEVIATIONS ARE GIVEN IN μS.

Solver	$n = 10$	$n = 15$	$n = 20$	$n = 50$
IPOPT[5]	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.	$\mathbf{18 \pm 10}$	$\mathbf{25 \pm 14}$	$\mathbf{35 \pm 22}$	–

of *all* rows from $\mathbf{C}_Z$ ($e_n = \mathbf{C}_Z^T \mathbf{1}_n$). 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 only arise if a^- or a^+ is equal to 0 or 1, what can be easily detected and avoided by slightly perturbing $\omega_{i,min}$ or $\omega_{i,max}$. It is thus safe to use a basic active-set scheme.

All QR decomposition are performed on matrices with rank deficiency of at most 1. 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 $n \leq 20$ 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 [25, p. 169].

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 for walking. If n is small enough (say $n \leq 20$), we can precompute and store the QR decompositions of *all* possible $\mathbf{J}_\mathcal{W}$ (there are $2^{n+1} - 1$ different sets $\mathcal{W}$: up to n active constraints among $n+1$). This can be done in a reasonable amount of time thanks to the above method, and results in even faster resolution times.

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 ($\alpha = 1$ in Algorithm 4) 98.5% of the time in practice, and converges in very few iterations (4 on average). Since we are starting 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.

Passing the nonlinear constraint as an objective with weight μ is reminiscent of penalty-based methods, where the penalty parameter μ is adapted during successive iterations. In our case, we observed that a fixed parameter (typically $\mu = 10^6$) was enough to get a precise solution in few iterations.

F. Performance comparison

In [11], Problem (70) was solved with the state-of-the-art solver IPOPT [21], which is written in Fortran and C. We compare its performances with our tailored SQP approach, implemented in C++[6]. Taking $\mu = 10^6$, the solutions returned

---

[5] 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.

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

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 by walking simulations 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 $\mathbf{J}_\mathcal{W}$.

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 [26]. This variant is denoted by *SQP + LSSOL*.

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 initialization. The limit rather lies with memory consumption, which follows an exponential law ranging from 2 MB for $n = 10$ to 5 GB for $n = 20$ (a rough upper bound is given by 2.2^{n-9} MB). In practice we use $n = 10$ where memory consumption is low.

When $n \geq 25$, SQP + LSSOL may start to fail (it does so 25% of the time for $n = 50$). This suggests that our least squares solver is more robust, a plausible explanation for this being that we leverage sparsity patterns of $\mathbf{J}$ and $\mathbf{C}$, as well as the knowledge that elements of $\mathbf{C}$ are exactly 1 or -1. We thus end up with exact computations (most notably for the nullspace and pseudoinverse of $\mathbf{C}_\mathcal{W}$) while LSSOL treats those matrices as dense and with floating-point coefficients.

V. ONE-STEP CAPTURABILITY OF THE 3D IPM

While zero-step capturability enables push recovery, the minimum price to pay for bipedal walking is *one-step* capturability, owing to the fact that stepping consists of two distinct phases: acceleration (positive work) by pushing on the takeoff foot, followed by deceleration (negative work) using the landing foot. Fortunately, the significant performance improvement achieved by the tailored solver opens new perspectives to go beyond zero-step solutions. Let

$$\mathcal{C}(s) = \begin{cases} \mathcal{C}_i & \text{for } s_c < s \leq 1 \\ \mathcal{C}_f & \text{for } 0 < s \leq s_c \end{cases} \tag{87}$$

denote a one-step contact sequence with $s_c \in (0, 1)$. We consider the piecewise-constant CoP trajectory defined by:

$$\boldsymbol{r}(s) = \begin{cases} \boldsymbol{r}_i & \text{for } s_c < s \leq 1 \\ \boldsymbol{r}_f & \text{for } 0 < s \leq s_c \end{cases} \tag{88}$$

This choice yields the following boundedness condition (44):

$$\boldsymbol{r}_i \omega_i + (\boldsymbol{r}_f - \boldsymbol{r}_i) s_c \omega(s_c) - \boldsymbol{g} \int_0^1 \frac{\mathrm{d}s}{\omega(s)} = \omega_i \boldsymbol{c}_i + \dot{\boldsymbol{c}}_i \tag{89}$$

That is to say, in terms of optimization variables $\boldsymbol{\varphi}$:

$$g \boldsymbol{e}_z b_\omega(\boldsymbol{\varphi}) = (\boldsymbol{c}_i - \boldsymbol{r}_i)\sqrt{\varphi_n} + (\boldsymbol{r}_i - \boldsymbol{r}_f)\sqrt{\varphi(s_c)} + \dot{\boldsymbol{c}}_i \tag{90}$$

where $b_\omega(\boldsymbol{\varphi})$ is the function defined by Equation (55), and $\varphi(s_c)$ is a linear combination of φ_j and φ_{j+1} when $s_c \in [s_j, s_{j+1}]$ from Equation (50). This equality constraint has a different form than the one we previously encountered in the capture problem (70).

Algorithm 1 Instantiation of $\{0,1\}$-step Capture Problems

Input: $\alpha \in (0,1)$, states $(\boldsymbol{x}_\mathrm{i}, \boldsymbol{x}_\mathrm{f})$, contacts $(\mathcal{C}_\mathrm{i}, \mathcal{C}_\mathrm{f})$
Output: capture input steering $\boldsymbol{x}_\mathrm{i}$ to $\boldsymbol{x}_\mathrm{f}$, if any
    Compute $\bar{z}_\alpha$ and $\dot{\bar{z}}_\mathrm{i}$ from $(\mathcal{C}_\mathrm{i}, \boldsymbol{x}_\mathrm{i}, \boldsymbol{x}_\mathrm{f})$ via (92)
    Compute the halfspace-representation $(\mathbf{F}, \boldsymbol{p})$ of $\mathcal{C}_\mathrm{i}$
    Reduce inequalities (94) into $\omega_{\mathrm{i,min}} \leq \omega_\mathrm{i} \leq \omega_{\mathrm{i,max}}$
    return CAPTUREPROBLEMSOLVER$(\omega_{\mathrm{i,min}}, \omega_{\mathrm{i,max}}, \bar{z}_\alpha, \dot{\bar{z}}_\mathrm{i})$

A. Reformulation to a capture problem

Let us define an external parameter $\alpha \in (0,1)$ such that $\sqrt{\varphi(s_\mathrm{c})} = \alpha\sqrt{\varphi_n}$. The boundedness condition becomes:

$$g\boldsymbol{e}_z b_\omega(\boldsymbol{\varphi}) = (\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\alpha)\sqrt{\varphi_n} + \dot{\boldsymbol{c}}_\mathrm{i} \tag{91}$$

where $\boldsymbol{r}_\alpha \overset{\mathrm{def}}{=} \alpha\boldsymbol{r}_\mathrm{f} + (1-\alpha)\boldsymbol{r}_\mathrm{i}$. Taking the dot product of this equation with the normal $\boldsymbol{n}_\mathrm{i}$ of the initial contact $\mathcal{C}_\mathrm{i}$ yields:

$$b_\omega(\boldsymbol{\varphi}) - \frac{\bar{z}_\alpha\sqrt{\varphi_n} + \dot{\bar{z}}_\mathrm{i}}{g} = 0 \qquad \bar{z}_\alpha \overset{\mathrm{def}}{=} \frac{(\boldsymbol{c}_\mathrm{i} - \boldsymbol{r}_\alpha)\cdot\boldsymbol{n}_\mathrm{i}}{(\boldsymbol{e}_z \cdot \boldsymbol{n}_\mathrm{i})} \tag{92}$$

This time, the constraint is of the form $b(\boldsymbol{\varphi}) = 0$ of the capture problem (72), where the parameter $\bar{z}_\mathrm{i}$ has been replaced by $\bar{z}_\alpha$. Meanwhile, the two horizontal components of Equation (91) are sufficient to characterize $\boldsymbol{r}_\mathrm{i}$:

$$\boldsymbol{r}_\mathrm{i}^{xy} = \frac{1}{1-\alpha}\left[\boldsymbol{c}_\mathrm{i}^{xy} + \frac{\dot{\boldsymbol{c}}_\mathrm{i}^{xy}}{\omega_\mathrm{i}} - \alpha\boldsymbol{r}_\mathrm{f}^{xy}\right] \tag{93}$$

Injecting this equation into the inequalities $\mathbf{F}\boldsymbol{r}_\mathrm{i} \leq \boldsymbol{p}$ of the contact area $\mathcal{C}_\mathrm{i}$ yields:

$$[\alpha\mathbf{F}\boldsymbol{r}_\mathrm{f} + (1-\alpha)\boldsymbol{p} - \mathbf{F}\boldsymbol{c}_\mathrm{i}]\,\omega_\mathrm{i} \geq \mathbf{F}\dot{\boldsymbol{c}}_\mathrm{i} \tag{94}$$

We recognize in these last three equations the conditions found in Section III for zero-step capture: Equation (92) is of the form used in the capture problem, Equations (66) and (93) are identical, and so are inequalities (67) and (94). We can therefore apply our existing solution as summarized in Algorithm 1. Zero-step capture is solved as a special case where this function is called with $\alpha = 0.5$ and $\mathcal{C}_\mathrm{i} = \mathcal{C}_\mathrm{f}$. We have thus reduced one-step capture to a capture problem *parameterized* by $\alpha \in (0,1)$. But how can we interpret this new parameter?

When the minimum of the regularizing cost function is attained and $\omega(s)$ is constant, $\alpha = s_\mathrm{c}$, meaning that the choice of α is equivalent to that of the contact switch s_c. In general, however, $\alpha = s_\mathrm{c}\omega(s_\mathrm{c})/\omega_\mathrm{i}$ couples s_c with variations of ω in a non-intuitive way. Informally, we can interpret α as an "index" of zero-step capturability: when feasible solutions exist for $\alpha \to 1$, $\boldsymbol{r}_\alpha \to \boldsymbol{r}_\mathrm{f}$ and $\varphi(s_\mathrm{c}) \to \varphi_n \Rightarrow s_\mathrm{c} \to 1$, meaning that the initial contact $\mathcal{C}_\mathrm{i}$ is not used. Conversely, when the largest feasible α is lower than one, all capture inputs need to use the first contact and the state is not zero-step capturable.

B. Constraint on the time to contact switch

An important consequence of the parameter α is its effect on the contact switch index s_c, and thus on the time of contact switch $t(s_\mathrm{c})$. Let us denote by $t_\mathrm{c}(\alpha)$ the contact-switch time $t(s_\mathrm{c})$ of the solution $\boldsymbol{\varphi}_\alpha$ to the capture problem parameterized by α. (We can compute $t_\mathrm{c}(\alpha)$ from $\boldsymbol{\varphi}_\alpha$ via Equation (71), by

replacing s_j by s_c in the last interval $s_j \leq s_\mathrm{c} < s_{j+1}$.) 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 $t_\mathrm{c}(\alpha)$ is greater than the remaining duration t_swing of the swing foot motion. This gives us an additional constraint $t_\mathrm{c}(\alpha) \geq t_\mathrm{swing}$, where each evaluation of $t_\mathrm{c}(\alpha)$ costs the resolution of a full capture problem.

C. External optimization over the parameter α

Fortunately, computation times achieved by the custom solver from Section IV allow us to solve several capture problems per control cycle. Rather than designing a new solver, we 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 a solution $\boldsymbol{\varphi}_\alpha$ is computed by Algorithm 1.

A straightforward way to carry out the external optimization is to test for all values in a partition $0 < \alpha_0 < \ldots < \alpha_m < 1$. Unfortunately, this approach is inefficient in practice as it produces a large number of unfeasible problems for the internal optimization.

There are two ways a bad choice of α can yield an unfeasible capture problem (70):

(a) The two bounds in the inequality constraint (70c) are such that $\omega_{\mathrm{i,min}} > \omega_{\mathrm{i,max}}$. Recall that both bounds are computed from the inequalities (94), which involve α.
(b) The intersection between the nonlinear equality constraint (70b) and the polytope (70c)–(70d) is empty. The influence of α on this comes from Equation (92).
(c) The right cylinder given by the linear constraint (70c) does not interesect the zonotope (70d). The influence of α on this comes from Equation (94).

Case (c) can be caught efficiently before solving the capture problem, as discussed in Section IV-D. While anticipating (b) is still an open question for us, case (a) can be avoided altogether thanks to a more careful treatment of CoP inequality constraints.

Let us rewrite the inequality (94) as:

$$(\boldsymbol{u} - \alpha\boldsymbol{v})\omega_\mathrm{i} \geq \boldsymbol{w} \tag{95}$$

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

- $u_j = 1$, $v_j = 0$ and $w_j = \omega_{\mathrm{i,min}}$
- $u_j = -1$, $v_j = 0$ and $w_j = -\omega_{\mathrm{i,max}}$

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

$$\omega_{\mathrm{i,min}}(\alpha) = \max\left\{\frac{w_i}{u_i - \alpha v_i}, i \in \mathcal{A}_{\min}(\alpha)\right\} \tag{96}$$

$$\omega_{\mathrm{i,max}}(\alpha) = \min\left\{\frac{w_i}{u_i - \alpha v_i}, i \in \mathcal{A}_{\max}(\alpha)\right\} \tag{97}$$

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

$$w_i u_j - w_j u_i \leq \alpha(w_i v_j - w_j v_i) \tag{98}$$

Algorithm 2 Computation of α feasibility intervals

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

  $\mathcal{I} \leftarrow \emptyset$
  $\mathcal{R} \leftarrow \{r_j = u_j/v_j | r_j \in (0,1)\}$
  for (r_{2j}, r_{2j+1}) consecutive roots in SORT($\mathcal{R}$) **do**
    $(\alpha_{\min}, \alpha_{\max}, \alpha) \leftarrow (r_{2j}, r_{2j+1}, \frac{1}{2}(r_{2j} + r_{2j+1}))$
    Compute index sets $\mathcal{A}_{\min}(\alpha)$ and $\mathcal{A}_{\max}(\alpha)$
    Reduce $[\alpha_{\min}, \alpha_{\max}]$ using (98) with $\mathcal{A}_{\min}(\alpha)$, $\mathcal{A}_{\max}(\alpha)$
    $\mathcal{I} \leftarrow \mathcal{I} \cup \{[\alpha_{\min}, \alpha_{\max}]\}$
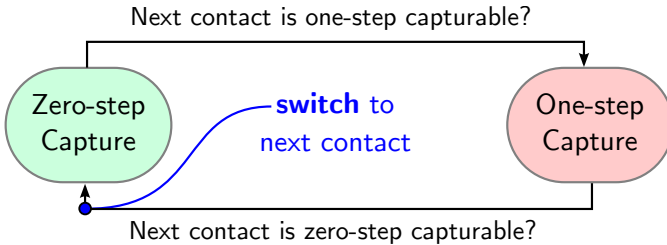 **end for**
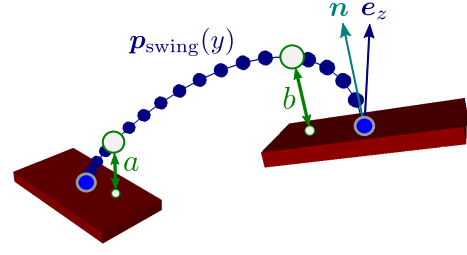 **return** $\mathcal{I}$

Fig. 4. **Interpolation of swing-foot trajectories.** Swing foot trajectories are interpolated as cubic Hermite curves with boundary positions and tangent directions. The remaining two curve parameters are optimized by Quadratic Programming to ensure clearing distances $a > a_{\min}$ and $b > b_{\min}$. Finally, the swing timing is derived using reachability-based time-optimal retiming [20].

These inequalities are of the form $\widetilde{\boldsymbol{u}}\alpha \geq \widetilde{\boldsymbol{v}}$ and can therefore be reduced using Equations (68)–(69) into a single interval $[\alpha_{\min}, \alpha_{\max}]$ on which it is guaranteed that $\omega_{i,\min} \leq \omega_{i,\max}$.

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 2.

In practical walking scenarios, it is common to encounter reunions such as $\alpha \in [0.1, 0.3] \cup [0.5, 0.8]$ consisting of one, two or three disjoint intervals. The outer optimization can choose to explore them in any order. We observed empirically that values of α closer to one usually yield smaller $t_c(\alpha)$, yet via a non-monotonic mapping. In addition, we noted how using only extremal values $\alpha \in \{\alpha_{\min}, \alpha_{\max}\}$ is not a sound strategy for global optimization, as lowest values of the cost function are often attained when α lies well inside its intervals.

VI. WALKING CONTROL

The combination of zero- and one-step capturability enables walking. Consider a sequence of contacts $\mathcal{C}_0, \mathcal{C}_1, \ldots$ given by a contact planner. We will say for short that a contact $\mathcal{C}$ is capturable when there exists a capture trajectory from the current robot state to the equilibrium $\boldsymbol{c}_f = \boldsymbol{o} + \bar{z}_f \boldsymbol{e}_z$ located at the height $\bar{z}_f$ above the center of its area. With this terminology, we can generate walking from capture trajectories via a two-state strategy [12], [27]:

Next contact is one-step capturable?

Zero-step Capture → **switch** to next contact → One-step Capture

Next contact is zero-step capturable?

The behavior realized by this state machine is conservative: when reaching a contact $\mathcal{C}_j$ in the sequence, the robot starts slowing down as if it were to stop and balance above $\mathcal{C}_j$ (*Zero-step Capture* state). This corresponds to the stance phase (negative work) of walking. While in this phase, the robot searches for a one-step capture trajectory stepping to the next contact $\mathcal{C}_{j+1}$. There are two potential outcomes at this stage:

- The next contact is not capturable from the current robot state: in this case, the robot simply stops walking and balances above the current contact.
- A trajectory is found: the robot then drops its balancing behavior and switches to the *One-step Capture* state, corresponding to the swing phase of walking (positive work) where the current contact $\mathcal{C}_j$ is used to accelerate toward $\mathcal{C}_{j+1}$. The process is continued until $\mathcal{C}_{j+1}$ becomes zero-step capturable, whereupon the robot switches back to the zero-step behavior, and the overall process is repeated.

This combination of zero- and one-step capturability is simple to implement, yet we will see that it is sufficient to walk across complex environments.

A. Model predictive control of capture trajectories

We follow model predictive control (MPC) to turn fast trajectory generation into closed-loop control [28], [29]. At each control cycle, we compute the optimum $\boldsymbol{\varphi}^*$ of the capture problem for the current state and convert it into robot inputs, extracting its initial stiffness λ_i and CoP $\boldsymbol{r}_i$ via:

$$\lambda_i(\boldsymbol{\varphi}^*) = \frac{\varphi_n^* - \varphi_{n-1}^*}{\delta_{n-1}} \quad \boldsymbol{r}_i(\boldsymbol{\varphi}^*) = \begin{cases} (58) & \text{for zero-step} \\ (93) & \text{for one-step} \end{cases}$$

These references are then sent to the lower-level controllers of the humanoid (foot force control, CoM position and angular-momentum minimization at the whole-body level) to be applied until the next control cycle. In the standard model-predictive fashion, the rest of the optimal trajectory is discarded. As a matter of fact, we don't compute explicitly the functions $\omega(s)$ or $\lambda(s)$ nor their time counterparts. Only for one-step capture do we compute the time mapping $t(s)$ via Equation (71) to make sure that we select values of the parameter α where contact switches after foot landing.

B. Swing foot trajectory generation

A swing foot trajectory comes under two conflicting imperatives: stay close to the ground while avoiding collisions. We assume here that the terrain is uneven but free from obstacles (see *e.g.* [30] for the converse setting of avoiding obstacles on even terrains). To avoid robot-ground collisions, we enforce two clearance distances: one at toe level for the takeoff contact and the other at heel level for the landing contact, as depicted in Figure 4.

We interpolate swing foot trajectories as cubic Hermite curves with four boundary constraints: initial and final positions corresponding to contact centers, and tangent directions parallel to contact normals. This leaves two free parameters, corresponding to the norms of the initial and final tangents, that can be optimized upon. It can be shown that minimizing these two parameters subject to the clearance conditions $a > a_{\min}$ and $b > b_{\min}$ (notations from Figure 4) is a small constrained least squares problem. We solve it and apply time-optimal retiming to the path thus obtained. Retiming allows us to compute swing timings according to a simplified model of swing foot dynamics. We model the foot as a free-floating body under maximum acceleration of 10 m.s^{-2}, and apply the recent reachability-analysis enhancement of time-optimal path parameterization (TOPP-RA) [20] for which open-source software is readily available.[7]

C. Simulations

We implemented capturability-based 3D walking in simulation with a model of the HRP-4 humanoid robot. Our simulations use *pymanoid*[8], an extension of OpenRAVE [31] for humanoid robotics. Whole-body inverse kinematics is implemented using a standard quadratic-programming formulation (see *e.g.* [32, Section 1] for a 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} to 1
IPM tracking	Center of mass	1×10^{-2}
IPM tracking	Min. CAM 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 consider three scenarios, depicted in Figure 5. The first one is an elliptic staircase with randomly-tilted footsteps (Figure 5a) used to test the ability to adapt to general uneven terrains where the ground may go up, go down or tilt in arbitrary directions. We also consider a regular staircase with 15-cm high steps (Figure 5b), where collision avoidance requires larger swing foot motions, which in turn affect the timings of the CoM trajectory. Finally, we consider 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 (first step 19.5 cm, last step 14.5 cm, all other steps 18 cm high), then across a flat floor area and finally inside the fuselage where the ground consists of temporary wooden slabs.

We use the same parameters across all three scenarios. For capture problems, we used $n = 10$ discretization steps with a partition $s_i = i/n$. The external optimization for one-step problems sampled 5 values of alpha per feasibility interval.

For the IPM, we set $\lambda_{\min} = 0.1g$, $\lambda_{\max} = 2g$ and a target CoM height $\bar{z}_\mathrm{f} = 0.8$ m suitable to HRP-4. For swing foot trajectories, we use a landing clearance of $b_{\min} = 10$ cm. Our only context-dependent parameters are those related to the takeoff phase: in staircases where contacts are close to each other, swing foot trajectories start with a backward-leaning tangent along with a clearance $a_{\min}$ equal to the step height. In all other cases, the contact normal is used as tangent, and $a_{\min} = 5$ cm.

In all scenarios, the robot starts and ends in double support (the first one-step capture state is triggered without swing foot trajectory so that the robot leverages its starting double support phase to gain some initial momentum). At each control cycle, the state machine needs to solve both zero-step and one-step capture problems: once for the current state and once to evaluate the transition condition. Fortunately, solving capture problems (70) can be done several times while staying fast enough for the control loop: on a consumer laptop computer, zero-step and one-step capture problems are solved in respectively 0.38 ± 0.13 ms and 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 external instantiation and optimization over α in Python (Algorithms 1 and 2).

The results of all three simulations are depicted in Figure 5 and in the accompanying video. We release all our source code for reference, review and reproducibility.[9]

D. Feedback from whole-body to reduced model

A well-known problem when using a reduced model to control the higher-dimensional system lies in the feasibility of the reverse map. Notably, the CoM trajectory output by the reduced model may not be trackable due to limited kinematic reachability of the leg. We noticed this problem when walking down the elliptic staircase (Figure 5a) where the prescribed stance CoM height of 80 cm conflicts with the need to crouch for reaching the next footstep. At present, two main strategies have been proposed to address this issue: *a priori* estimation of reachable CoM positions from contact locations [33], [34] or more recently *a posteriori* feedback of the whole-body CoM position to the reduced model [35]. The former is a planning strategy, the latter a control one. Given our assumption that the contact sequence is given by an external (potentially faulty) contact planner, we only investigated the feedback strategy. We applied the idea from Sato and Sugihara [35] at the velocity level, feeding back the whole-body CoM position via a first-order low-pass filter with cutoff frequency 20 Hz. This solved reachability issues in all three scenarios with little implementation cost. Interestingly, we observed as a consequence of this feedback the emergence of butterfly-shaped frontal CoM motions [33, Figure 4].

VII. DISCUSSION AND FUTURE WORK

While the present study is coming to an end, our understanding of the inverted pendulum model is, hopefully, only
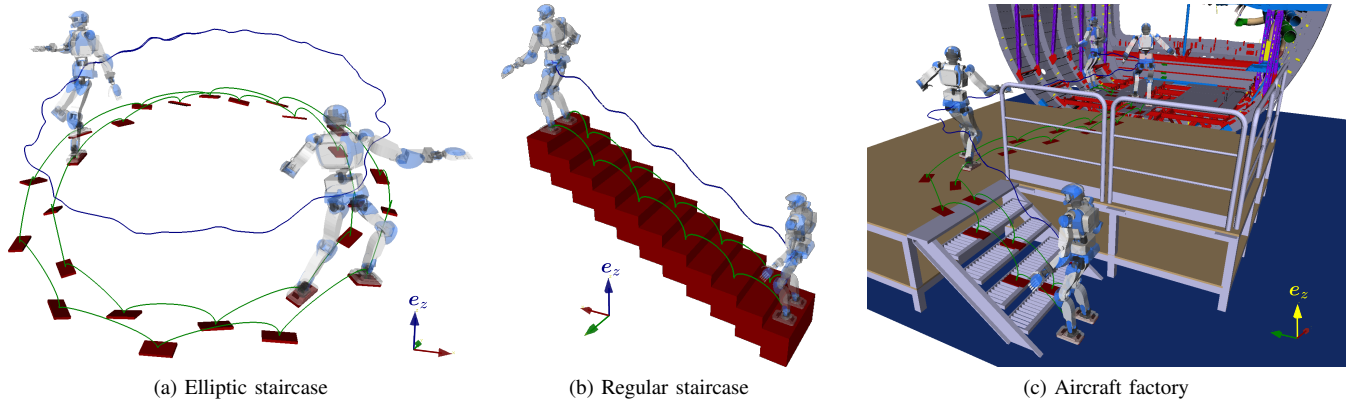

Fig. 5. **Simulation scenarios:** the elliptic staircase with randomly-tilted footsteps (a) tests the ability to walk over rough terrains, *i.e.* to adapt to both 3D translation and 3D orientation variations between contacts. The regular staircase (b) has 15-cm high steps; it assesses the behavior of 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 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.

beginning to unfold. What did we understand so far? First, that capturability of the inverted pendulum 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 can be solved orders of magnitude faster than generic nonlinear problems. This allows us to solve for both zero- and one-step capturability by breaking them down into one or several capture problems, all solved at once without exceeding the time budget of a control loop. With these tools in hand, walking becomes a matter of closing the loop via model predictive control.

Our overall discussion draws numerous connections with the existing literature. To start with, the decomposition of the nonlinear inverted pendulum into its convergent and divergent components was proposed in 2004 by J. Hauser *et al.* [13] 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 LIPM itself has been the focus of a large part of the recent literature, in the wake of major works such as [36], [28], [4], [2]. Solutions allowing for 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 [37], [38], [14], the remainer of the system can be controller in the 2D horizontal plane similarly to the LIPM, yet with a time-variant rather than time-invariant equation of motion. Two successful LIPM solutions have been generalized following this idea: linear model predictive control [28] was extended into [38], and the time-invariant divergent component of motion [6] was extended into a time-variant counterpart [14].[10]

---

[10] Both [38] and [14] use polynomial CoM height functions, which makes it easy to satisfy boundary conditions but yields non-integrable dynamics. Terada and Kuniyoshi [37] proposed a symmetric alternative where the system becomes integrable, yet where enforcing boundary conditions is a nonlinear root finding problem.

Interestingly, in [14] Hopkins *et al.* use the Riccati equation (13) to compute $\omega(t)$ from $c^z(t)$, while in the present study we compute $\boldsymbol{c}(t)$ 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 $\boldsymbol{c}^{xy}$. Another noteworthy example of the latter can be found in the linearized MPC proposed by Brasseur *et al.* [33], 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 LIPM by Lanari *et al.* [15] and applied to model predictive control of the LIPM in [29]. This condition can be more generally applied to different asymptotic behaviors, including but not restricted to stopping. For instance, infinite stepping is another option [39]. 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 [40], [41], [27] and uneven terrains [42], [12]. The latter follow a single line of work leading to the present study: [42] finds rough-terrain (even multi-contact) solutions but tends to produce conservatively slow trajectories; [12] 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 [43], [44] but where closed-loop controllers suffer from frequent switches between local optima [12]. 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 [45]. Both cases, as noted in [45] and in the present study, bear a close connection with time-optimal path parameterization (see *e.g.* [16] 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 leverage this tool for fast evaluation of contact reachability, while controllers can evaluate several contact candidates in parallel, adapting their choice to *e.g.* external pushes or changes in the desired walking direction. These extensions are open to future works.

Acknowledgment

References

[1] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Proceedings of the 6th 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 *Proceedings of the 2009 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 *Proceedings of the 2009 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 *Proceedings of the 12th 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 *Proceedings of the 2017 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 *Proceedings of the 2007 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 *Proceedings of the 15th 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 *Proceedings of the 16th 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," Sep. 2017, accepted to ICRA 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01590509

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

[13] J. Hauser, A. Saccon, and R. Frezza, "Achievable motorcycle trajectories," in *Proceedings of the 2004 IEEE Conference on Decision and Control*, vol. 4, 2004, pp. 3944–3949.

[14] M. A. Hopkins, D. W. Hong, and A. Leonessa, "Humanoid locomotion on uneven terrain using the time-varying divergent component of motion," in *Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 266–272.

[15] L. Lanari, S. Hutchinson, and L. Marchionni, "Boundedness issues in planning of locomotion trajectories for biped robots," in *Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 951–958.

[16] Q.-C. Pham, "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1533–1540, 2014.

[17] B. Stephens, "Humanoid push recovery," in *Proceedings of the 7th IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 589–595.

[18] V. Samy, S. Caron, K. Bouyarmane, and A. Kheddar, "Post-impact adaptive compliance for humanoid falls using predictive control of a reduced model," in *Proceedings of the 17 IEEE-RAS International Conference on Humanoid Robots*, 2017.

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

[20] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," Jul. 2017, pre-print.

[21] 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.

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

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

[24] 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/

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

[26] P. E. E. Gill, S. J. Hammarling, W. Murray, M. A. Saunders, and M. H. Wright, "User's guide for lssol (version 1.0): a fortran package for constrained linear least-squares and convex quadratic programming," Standford University, Standord, California 94305, Tech. Rep. 86-1, January 1986.

[27] 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.

[28] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *Proceedings of the 6th IEEE-RAS International Conference on Humanoid Robots*, 2006, pp. 137–142.

[29] N. Scianca, M. Cognetti, D. De Simone, L. Lanari, and G. Oriolo, "Intrinsically stable MPC for humanoid gait generation," in *Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 101–108.

[30] S. Brossette and P.-B. Wieber, "Collision avoidance based on separating planes for feet trajectory generation," in *Proceedings of the 17th IEEE-RAS International Conference on Humanoid Robots*, 2017.

[31] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, Aug. 2010.

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

[33] 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 *Proceedings of the 15th IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 595–601.

[34] J. Carpentier, R. Budhiraja, and N. Mansard, "Learning Feasibility Constraints for Multi-contact Locomotion of Legged Robots," in *Proceedings of Robotics: Science and Systems*, vol. Proceedings of Robotics Science and Systems, Cambridge, MA, United States, Jul. 2017.

[35] R. K. Sato and T. Sugihara, "Walking control for feasibility at limit of kinematics based on virtual leader-follower," in *Proceedings of the 17th IEEE-RAS International Conference on Humanoid Robots*, 2017.

[36] 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 *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, vol. 2, 2003, pp. 1620–1626.

[37] K. Terada and Y. Kuniyoshi, "Online gait planning with dynamical 3d-symmetrization method," in *Proceedings of the 7th IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 222–227.

[38] A. Herdt, N. Perrin, and P.-B. Wieber, "Lmpc based online generation of more efficient walking motions," in *Proceedings of the 12th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2012, pp. 390–395.

[39] L. Lanari and S. Hutchinson, "Planning desired center of mass and zero moment point trajectories for bipedal locomotion," in *Proceedings of the 15th IEEE-RAS International Conference on Humanoid Robots*, 2015, pp. 637–642.

[40] 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.

[41] M. Khadiv, A. Herzog, S. A. A. Moosavian, and L. Righetti, "Step timing adjustment: A step toward generating robust gaits," in *Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 35–42.

[42] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3d com accelerations," in *Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots*, 2016.

[43] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 295–302.

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

[45] 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 *Proceedings of the 17th IEEE-RAS International Conference on Humanoid Robots*, 2017, pp. 599–605.

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

APPENDIX A
PROOF OF PROPERTY 1

As $\mathcal{I}^c_{\boldsymbol{x}_i, \boldsymbol{x}_f} \subset \mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$, it is enough to prove that $\mathcal{I}^c_{\boldsymbol{x}_i, \boldsymbol{x}_f}$ is non-empty as soon as $\mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$ is non-empty, *i.e.* that given a input function $\lambda(t), \boldsymbol{r}(t) \in \mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$ we can find another input

$\lambda^c(t), \boldsymbol{r}^c(t)$ steering to the same state while converging. To this end, consider the following state-dependent inputs:

$$\sqrt{\bar{\lambda}(\boldsymbol{x})} = 2 \frac{\sqrt{g(5\bar{z} - \bar{z}_f) + \dot{\bar{z}}^2} - \dot{\bar{z}}_i}{5\bar{z}_i - \bar{z}_f} \tag{99}$$

$$\bar{\boldsymbol{r}}(\boldsymbol{x}) = \boldsymbol{c} + \frac{\boldsymbol{c} - \boldsymbol{c}_f}{4} + \frac{\dot{\boldsymbol{c}}}{\sqrt{\bar{\lambda}(\boldsymbol{x})}} + \frac{\boldsymbol{g}}{\bar{\lambda}(\boldsymbol{x})} \tag{100}$$

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

$$\left[\boldsymbol{c} - \boldsymbol{o} + \frac{\boldsymbol{c} - \boldsymbol{c}_f}{4} \right] \cdot \boldsymbol{n} + \frac{(\dot{\boldsymbol{c}} \cdot \boldsymbol{n})}{\sqrt{X}} + \frac{(\boldsymbol{g} \cdot \boldsymbol{n})}{X} = 0 \tag{101}$$

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

Injecting those inputs into (1) yields the nonlinear differential equation:

$$\ddot{\boldsymbol{c}}(t) = -\frac{\bar{\lambda}(\boldsymbol{x}(t))}{4}(\boldsymbol{c}(t) - \boldsymbol{c}_f) - \sqrt{\bar{\lambda}}\dot{\boldsymbol{c}}(t) \tag{102}$$

It is immediate that $\boldsymbol{x}_f$ is an equilibrium for this dynamics. The linearized system around this equilibrium is:

$$\ddot{\boldsymbol{c}}^\ell(t) = -\frac{\lambda_f}{4}(\boldsymbol{c}^\ell(t) - \boldsymbol{c}_f) - \sqrt{\lambda_f}\dot{\boldsymbol{c}}^\ell(t) \tag{103}$$

for which the equilibrium is stable. Therefore the equilibrium $\boldsymbol{x}_f$ of (102) is locally stable: if $\boldsymbol{x}(0)$ is close enough to $\boldsymbol{x}_f$, then $\boldsymbol{x}(t)$ remains close to $\boldsymbol{x}_f$ and converges toward this limit.

We now consider a generic input function $\lambda(t), \boldsymbol{r}(t) \in \mathcal{I}_{\boldsymbol{x}_i, \boldsymbol{x}_f}$. By definition, the solution of (1) converges to $\boldsymbol{x}_f$ as $t \to \infty$. Then, there exists some time T such that $\boldsymbol{x}(T)$ is close enough to $\boldsymbol{x}_f$ so that, starting from this position, the state-dependent control remains feasible and converges to $\boldsymbol{x}_f$. We conclude by noting that the input function that switches at time T from $\lambda, \boldsymbol{r}$ to $\bar{\lambda}, \bar{\boldsymbol{r}}$ belongs to $\mathcal{I}^c_{\boldsymbol{x}_i, \boldsymbol{x}_f}$.

APPENDIX B
BACKGROUND ON NUMERICAL OPTIMIZATION

In this Appendix, we recall terminology and state-of-the-art algorithms for numerical optimization. We essentially rewrite treatment from [22] for double-sided inequality constraints.

A. Definitions and notations

Consider the optimization problem:

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

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

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:

$$\mathcal{E} \overset{\text{def}}{=} \{j \in [1, m], l_j = u_j\} \tag{105}$$

$$\mathcal{A}(\boldsymbol{x})^- \overset{\text{def}}{=} \{j \notin \mathcal{E}, h_j(\boldsymbol{x}) = l_j\} \tag{106}$$

$$\mathcal{A}(\boldsymbol{x})^+ \overset{\text{def}}{=} \{j \notin \mathcal{E}, h_j(\boldsymbol{x}) = u_j\} \tag{107}$$

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 $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}^-, \boldsymbol{\lambda}^+) \overset{\text{def}}{=} f(\boldsymbol{x}) + \boldsymbol{\lambda}^{-T}(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{l}) + \boldsymbol{\lambda}^{+T}(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{u})$ 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 $\boldsymbol{\lambda} \overset{\text{def}}{=} \boldsymbol{\lambda}^- + \boldsymbol{\lambda}^+$. We can work with it instead of $\boldsymbol{\lambda}^-$ and $\boldsymbol{\lambda}^+$ (see [46, §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 [22, chap. 12]). They are often use 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[11] by solving at each iteration k the following Quadratic Program with only Equality constraints (QPE):

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

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

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:

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

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 3. See [22, chap. 16] for more details on this method.

C. Sequential Quadratic Programming

Sequential quadratic programming (SQP) is an iterative optimization technique for solving general constrained problems

---

Algorithm 3 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 $\boldsymbol{x}_{\text{f}} = \boldsymbol{x}_k$
        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

---

such as (104). At each iteration k, a QP approximation of (104) is formed and solved:

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

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

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

Algorithm 4 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 4. There are several criteria for assessing whether a step is acceptable, see [22, chap. 18] for details.

---

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