Multi-contact Walking Pattern Generation based on Model Preview Control of 3D COM Accelerations

Stéphane Caron, Abderrahmane Kheddar

Multi-contact Walking Pattern Generation based on Model Preview Control of 3D COM Accelerations

Stéphane Caron[1] and Abderrahmane Kheddar[1,2]

***Abstract*— We present a multi-contact walking pattern generator based on preview-control of the 3D acceleration of the center of mass (COM). A key point in the design of our algorithm is the calculation of contact-stability constraints. Thanks to a mathematical observation on the algebraic nature of the frictional wrench cone, we show that the 3D volume of feasible COM accelerations is always an upward-pointing cone. We reduce its computation to a convex hull of (dual) 2D points, for which optimal $\mathcal{O}(n \log n)$ algorithms are readily available. This reformulation brings a significant speedup compared to previous methods, which allows us to compute time-varying contact-stability criteria fast enough for the control loop. Next, we propose a conservative *trajectory-wide* contact-stability criterion, which can be derived from COM-acceleration volumes at marginal cost and directly applied in a model-predictive controller. We finally implement this pipeline and exemplify it with the HRP-4 humanoid model in multi-contact dynamically walking scenarios.**

I. Introduction

Years ago, humanoid robots were considered as research platforms with vague perspectives in terms of concrete applications. Without much conviction, they were envisioned for entertainment, as receptionists, or as a high-tech showcase for other businesses. Some projects are challenging humanoids to be a daily companion or an assistant for frail persons[1]. The DARPA robotics challenge boosted the idea that humanoid robots can operate in disaster interventions. The challenge exhibited interesting developments while highlighting the road ahead. Nowadays, Airbus Group seriously envisions humanoids as manufacturing robots to act in large-scale airliner assembly lines. What makes humanoid robots a plausible solution in these applications is their physical ability to move in confined spaces, on non-flat floors, using stairs, etc. In such environments, there are large parts where the robot has to walk robustly.

Walking robustly on uneven floors is still an open problem in humanoid research. Recently, Boston Dynamics released an impressive video showing robust humanoid walks on various terrains[2]. This demonstration proves that the goal can be achieved. One key difficulty in locomotion is that the viability (the ability to avoid falling) of the states traversed while walking depends on future contacts. This problem can be addressed geometrically, as done in [2] using the generalized 3D capture point, or using dynamic programming as done in [3] walking in the phase-space of the center of mass (COM), the latter being constrained on predefined surfaces that conform to terrain shape.

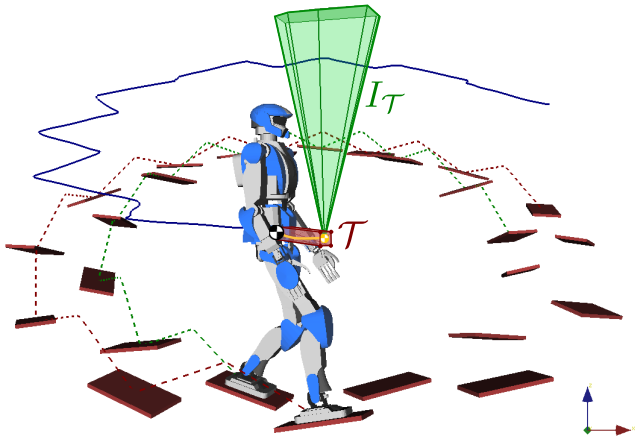

Fig. 1. HRP-4 walking on a circular staircase with tilted stepping stones using a preview controller based on 3D COM accelerations. By bounding future COM positions (preview trajectory in yellow) into a polytope $\mathcal{T}$ (red box), we derive a *trajectory-wide* contact-stability condition, the COM acceleration cone $I_{\mathcal{T}}$ (in green), that is efficiently computed using a 2D convex hull algorithm. (Scale and position of this acceleration cone were chosen arbitrarily for depiction purposes.) See the accompanying video [1] for demonstrations of the controller in various multi-contact scenarios.

Model Predictive Control (MPC) is another widely applied framework that gives controllers the required hindsight to tackle this question. Following the design introduced by Hirukawa et al. [4], one active line of research [5], [6], [7] uses contact forces as control variables, which produces optimization problems with simple inequality constraints but a high number of control variables. Another line of research reduces the problem to the center of mass motion [8], [9], [10], [11]. Optimization problems are then much smaller, but their inequality constraints become quadratic (and non-positive-semidefinite [10]). So far, this problem has only been addressed for walking on parallel horizontal surfaces: in [8], by bounding vertical COM accelerations to keep the formulation linear, and in [9], [10], where Sequential Quadratic Programming was used to cope with quadratic inequalities.

In this paper, we introduce a method that decouples the quadratic inequalities of the general multi-contact problem into pairs of linear constraints, thus opening the way for resolution with classical MPC solvers.

*This work is supported in part by H2020 EU project COMANOID http://www.comanoid.eu/, RIA No 645097.

[1]CNRS-UM2 LIRMM, IDH group, UMR5506, Montpellier, France.
[2]CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/RL.
Corresponding author: stephane.caron@normalesup.org

[1]http://projetromeo.com/

[2]https://www.youtube.com/watch?v=rVlhMGQgDkY

II. Background

A. Screw algebra

Humanoid robots are commonly modeled as a set of rigid bodies and joints whose motion can be described by *screws* [12]. A screw $\boldsymbol{s}_O = (\boldsymbol{r}, \boldsymbol{m}_O)$ is a vector field generated by two vectors: its *resultant* $\boldsymbol{r}$ and its *moment* $\boldsymbol{m}_O$ at a given point O. From $\boldsymbol{m}_O$ and $\boldsymbol{r}$, the moment at any other point P results from the Varignon formula:

$$\boldsymbol{m}_P \;=\; \boldsymbol{m}_O + \overrightarrow{PO} \times \boldsymbol{r}. \tag{1}$$

The generalized velocity of a rigid body, called *twist*, is the screw $\boldsymbol{t}_O = (\boldsymbol{v}_O, \boldsymbol{\omega})$ with resultant $\boldsymbol{\omega}$, the angular velocity, and moment $\boldsymbol{v}_O$, its velocity at given point O. A generalized force acting on the body, called *wrench*, is a screw $\boldsymbol{w}_O = (\boldsymbol{f}, \boldsymbol{\tau}_O)$ with resultant net force $\boldsymbol{f}$, and net moment $\boldsymbol{\tau}_O$, at a reference point O. Although the coordinate vector $\boldsymbol{s}_P$ of a screw depends on the point P where it is taken, the screw itself does not depend on the choice of P as a consequence of the Varignon formula (1). We denote screws with hats ($\hat{\boldsymbol{t}}$, $\hat{\boldsymbol{w}}$) and their coordinates with point subscripts ($\boldsymbol{t}_O$, $\boldsymbol{w}_O$).

Twists and wrenches live in two dual spaces [12]: the motion space M^6 and the force space F^6. The scalar product between a twist $\hat{\boldsymbol{t}} \in \mathsf{M}^6$ and a wrench $\hat{\boldsymbol{w}} \in \mathsf{F}^6$ is given by:

$$\hat{\boldsymbol{t}} \cdot \hat{\boldsymbol{w}} \;\stackrel{\text{def}}{=}\; \boldsymbol{t}_O \cdot \boldsymbol{w}_O \;=\; \boldsymbol{v}_O \cdot \boldsymbol{f} + \boldsymbol{\omega} \cdot \boldsymbol{\tau}_O. \tag{2}$$

From (1), this number does not depend on the point O where it is computed. When $\hat{\boldsymbol{t}}$ and $\hat{\boldsymbol{w}}$ are acting on a single rigid body, their product is the instantaneous power of the motion.

B. Newton-Euler equations

Let m denote the total mass of the robot and G its center of mass (COM). We write $\boldsymbol{p}_A$ the coordinate vector of a point A in the inertial frame and denote by O the origin of this frame (so that $\boldsymbol{p}_O = \boldsymbol{0}$). Suppose that contacts between the robot and its environment are described by K contact points. (This formulation includes surface contacts; see *e.g.* [13].) The Newton-Euler equations of motion of the whole robot are then given by:

$$\begin{bmatrix} m\ddot{\boldsymbol{p}}_G \\ \dot{\mathbf{L}}_G \end{bmatrix} \;=\; \begin{bmatrix} m\boldsymbol{g} \\ \boldsymbol{0} \end{bmatrix} + \sum_{i=1}^{K} \begin{bmatrix} \boldsymbol{f}_i \\ \overrightarrow{GC}_i \times \boldsymbol{f}_i \end{bmatrix} \tag{3}$$

where $\mathbf{L}_G$ is the angular momentum of the robot around G, $\boldsymbol{g} = [0\ 0\ -g]^\top$ is the gravity vector defined from the gravity constant $g \approx 9.81\,\mathrm{m\,s^{-1}}$ and $\boldsymbol{f}_i$ is the force exerted onto the robot at the i^{th} contact point C_i. We say that a contact force $\boldsymbol{f}_i$ is *feasible* when it lies in the friction cone $\mathcal{C}_i$ directed by the contact normal $\boldsymbol{n}_i$, *i.e.*,

$$\|\boldsymbol{f}_i - (\boldsymbol{f}_i \cdot \boldsymbol{n}_i)\boldsymbol{n}_i\|_2 \;\;\leq\;\; \mu_i(\boldsymbol{f}_i \cdot \boldsymbol{n}_i) \tag{4}$$

where μ_i is the static friction coefficient. The problem of *contact stability* (also called *force balance*) is to find whole-body motions for which (3) admits solutions with feasible contact forces $\{\boldsymbol{f}_i\}$.

The Newton-Euler equations describe the components of motion that are independent from the actuation power of the robot, and play a critical role in locomotion. Most of today's trajectory generators [14], [4], [15], [8], [6], [2], [10] focus on solving these equations and rely on whole-body controllers to take actuation limits into account at a later stage of the motion generation process.

C. Wrench cones

Equations (3)-(4) include a large number of force variables. Although some walking pattern generators chose to work directly on this representation [4], another line of research [16], [17] found that these force variables can be eliminated by propagating their inequality constraints (4) into inequalities on the target rate of change $(m\ddot{\boldsymbol{p}}_G, \dot{\mathbf{L}}_G)$ of the whole-body momentum.

Define the net *contact wrench* by:

$$\boldsymbol{w}_O \;=\; \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{\tau}_O \end{bmatrix} \;\stackrel{\text{def}}{=}\; \sum_{i=1}^{K} \begin{bmatrix} \boldsymbol{f}_i \\ \overrightarrow{OC}_i \times \boldsymbol{f}_i \end{bmatrix} \tag{5}$$

In matrix form, $\boldsymbol{w}_O = \mathbf{G}_O \boldsymbol{f}_{\text{all}}$ where $\boldsymbol{f}_{\text{all}}$ is the stacked vector of contact forces and $\mathbf{G}_O$ is the *grasp matrix*. This wrench can be directly computed from whole-body motions, as it only differs from the whole-body momentum by a constant $\hat{\boldsymbol{w}}^g$ due to gravity.

Next, one can linearize regular friction cones $\mathcal{C}_i$ into polyhedral convex cones $\widetilde{\mathcal{C}}_i$, so that (4) becomes in matrix form (see *e.g.* [17] for details):

$$\boldsymbol{f}_i \in \widetilde{\mathcal{C}}_i \;\Leftrightarrow\; \mathbf{F}_i \boldsymbol{f}_i \leq \boldsymbol{0} \tag{6}$$

This form is known as the *halfspace representation* of a polyhedral cone. From the Weyl-Minkowski theorem, any polyhedron thus described can be equivalently written as:

$$\widetilde{\mathcal{C}}_i \;=\; \mathit{conv}(\{\boldsymbol{v}_i\}) + \mathit{rays}(\{\boldsymbol{r}_j\}), \tag{7}$$

where $\mathit{conv}(\{\boldsymbol{v}_i\}) = \{\sum_i \alpha_i \boldsymbol{v}_i, \forall i\, \alpha_i \geq 0 \sum_i \alpha_i = 1\}$ is the *convex hull* of a set of vertices, and similarly $\mathit{rays}(\{\boldsymbol{r}_j\}) = \{\sum_i \lambda_i \boldsymbol{r}_i, \forall i\, \lambda_i \geq 0\}$ denotes positive combinations of a set of rays. This form is known as the *vertex representation* of a polyhedron.

Using suitable conversions between these two representations [16], [17], one can finally compute the Contact Wrench Cone (CWC) described in halfspace representation by:

$$\mathbf{A}_O \boldsymbol{w}_O \leq \boldsymbol{0} \tag{8}$$

By construction, a net contact wrench $\boldsymbol{w}_O$ belongs to the CWC if and only if there exists a set of contact forces $\{\boldsymbol{f}_i\}$ satisfying both (3)-(5) and (6). Hence, the CWC provides a necessary and sufficient condition for the contact stability of whole-body motions.

III. Friction Cones are Dual Twists

Let us consider a row $\boldsymbol{a} = [\boldsymbol{a}_1^\top \boldsymbol{a}_2^\top]^\top$ of the CWC matrix $\mathbf{A}_O$. It defines an inequality constraint of the form

$$\boldsymbol{a}_1 \cdot \boldsymbol{f} + \boldsymbol{a}_2 \cdot \boldsymbol{\tau}_O \;\leq\; 0 \tag{9}$$

Applying the Varignon formula (1), the cone for the same wrench $\boldsymbol{w}_G$ taken at a different point G is subject to:

$$\boldsymbol{a}_1 \cdot \boldsymbol{f} + \boldsymbol{a}_2 \cdot (\boldsymbol{\tau}_G + \overrightarrow{OG} \times \boldsymbol{f}) \;\leq\; 0 \tag{10}$$

Using the invariance of the mixed product under circular shift, we can rewrite the left-hand side as:

$$(\boldsymbol{a}_1 + \overrightarrow{GO} \times \boldsymbol{a}_2) \cdot \boldsymbol{f} + \boldsymbol{a}_2 \cdot \boldsymbol{\tau}_G \ \leq \ 0 \tag{11}$$

Let us now defined the *dual twist* $\hat{\boldsymbol{a}} \in \mathsf{M}^6$ by:

$$\begin{bmatrix} \boldsymbol{a}_O \\ \boldsymbol{a} \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \end{bmatrix} \tag{12}$$

Equations (9) and (11) rewrite to:

$$\boldsymbol{a}_O \cdot \boldsymbol{f} + \boldsymbol{a} \cdot \boldsymbol{\tau}_O \ \leq \ 0 \tag{13}$$

$$\boldsymbol{a}_G \cdot \boldsymbol{f} + \boldsymbol{a} \cdot \boldsymbol{\tau}_G \ \leq \ 0 \tag{14}$$

where $\boldsymbol{a}_G = \boldsymbol{a}_1 + \overrightarrow{GO} \times \boldsymbol{a}_2$. In concise form:

$$\hat{\boldsymbol{a}} \cdot \hat{\boldsymbol{w}} \ \leq \ 0 \tag{15}$$

This inequality is independent from O where $\mathbf{A}_O$ is computed. Therefore, the CWC can be interpreted as a set of dual twists, the coordinates $\mathbf{A}_O$ of which one can compute at a fixed reference point using known techniques [16], [17].

This shift in the way of considering the cone has an important implication: using the Varignon formula, we can now calculate analytically the cone $\mathbf{A}_G$ at a mobile point G using a fixed solution $\mathbf{A}_O$ and the vector coordinates $\boldsymbol{p}_G$. Our following contributions build upon this property.

IV. Contact Stability Areas and Volumes

A. Static-equilibrium COM polygon

Bretl and Lall [18] showed how static equilibrium can be sustained by feasible contact forces if and only if the (horizontal projection of the) center of mass lies inside a specific polygon, henceforth called the *static-equilibrium polygon*.

In fact, the static-equilibrium polygon is embedded in the CWC. Suppose that its matrix $\mathbf{A}_O$ was computed at a given point O, and let $\hat{\boldsymbol{a}}$ denote a twist of the CWC corresponding to the inequality (13). In static equilibrium, the whole-body momentum is zero, so that the net contact wrench $\boldsymbol{w}_G$ at the center of mass G is simply opposed to gravity:

$$\boldsymbol{w}_G \ = \ \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{\tau}_G \end{bmatrix} \ = \ \begin{bmatrix} -m\boldsymbol{g} \\ \mathbf{0} \end{bmatrix} \tag{16}$$

Then, expressing (13) at G, (14) yields:

$$\boldsymbol{a}_G \cdot (-m\boldsymbol{g}) + \boldsymbol{a} \cdot \mathbf{0} \ \leq \ 0 \tag{17}$$

which also writes, since $m > 0$:

$$-\left(\boldsymbol{a}_O + \boldsymbol{a} \times \boldsymbol{p}_G\right) \cdot \boldsymbol{g} \ \leq \ 0 \tag{18}$$

Expanding this scalar product yields:

$$a_{Oz} - a_y x_G + a_x y_G \ \leq \ 0 \tag{19}$$

where $\boldsymbol{a}_O = [a_{Ox} \ a_{Oy} \ a_{Oz}]^\top$ and $\boldsymbol{a} = [a_x \ a_y \ a_z]^\top$. The set of inequalities (19) over all twists $\hat{\boldsymbol{a}}$ of the CWC provides the half-plane representation of the static-equilibrium polygon. Note that the static-equilibrium polygon does not depend on the mass, which was not observed in previous works [18], [19], [11], [20].

In what follows, we will use the following equivalent formulation. Let us define the *slackness* of (19) by:

$$\sigma_{\hat{\boldsymbol{a}}}(x_G, y_G) \ \stackrel{\text{def}}{=} \ -a_{Oz} + a_y x_G - a_x y_G \tag{20}$$

it is the signed distance between (x_G, y_G) and the supporting line $-a_y x + a_x y + a_{Oz} = 0$ of the corresponding static-equilibrium polygon's edge. A point (x_G, y_G) is then inside the polygon if and only if $\sigma_{\hat{\boldsymbol{a}}}(x_G, y_G) \geq 0$ for all the CWC twists $\hat{\boldsymbol{a}}$.

B. Vertex enumeration for polygons

The half-plane representation (19) is best-suited for COM feasibility tests. Meanwhile, the vertex representation is best-suited for planning. Converting from halfspace to vertex representation is known as the *vertex enumeration* problem, for which the *double description method* [21] has been applied in previous works [22], [16], [23], [17].

For general d-dimensional polyhedra, vertex enumeration has polynomial, yet super-linear time complexity. For example, the Avis-Fukuda algorithm [24] runs in $\mathcal{O}(dhv)$, with h and v the numbers of hyperplanes and vertices, while the original double-description method by Motzkin has a worst-case time complexity of $\mathcal{O}(h^2 v^3)$ [21]. Yet, for $d = 2$, the problem boils into computing the *convex hull* of a set of points, for which optimal algorithms (for instance [25]) are known that match the theoretical lower-bound of $\Omega(h \log v)$.

Our formulation (19) allows us to enumerate vertices in 2D. Let us assume for now that the origin $(x_G, y_G) = (0, 0)$ lies in the interior of the polygon, and divide each inequality (19) by a_{Oz} to put the overall inequality system in polar form:

$$\mathbf{B} \begin{bmatrix} x_G \\ y_G \end{bmatrix} \ \leq \ \mathbf{1} \tag{21}$$

We run a convex hull algorithm on the rows of the matrix $\mathbf{B}$. By duality, the cyclic order of extreme points thus computed corresponds to a cyclic order of adjacent edges for the primal problem. Intersecting pairs of adjacent lines in this order yields the vertices of the initial polygon. The conversion of inequalities (19) to (21) being $\mathcal{O}(n)$, computing the output polygon is done overall in $\mathcal{O}(h \log v)$. See the Appendix for a comparison with existing approaches.

To construct (21), we assumed that the origin lies inside the polygon. When this is not the case, one can simply compute the Chebyshev center (x_C, y_C) by solving a single Linear Program (LP) as detailed *e.g.* in [26] p. 148. From there, a translation $(x'_G, y'_G) = (x_G - x_C, y_G - y_C)$ brings the origin inside the polygon.

C. Pendular ZMP support areas

Let us revisit the derivation of the pendular ZMP support area [11] using our new approach. To achieve linear-pendulum mode of the Newton-Euler equations of the system, the following four equality constraints are applied to the contact wrench:

$$\boldsymbol{n} \cdot \boldsymbol{f} \ = \ m(\boldsymbol{n} \cdot \boldsymbol{g}) \tag{22}$$

$$\boldsymbol{\tau}_G \ = \ \mathbf{0} \tag{23}$$

where $\boldsymbol{n}$ denotes the unit vector normal to the plane in which the ZMP is taken. In what follows, we suppose that $\boldsymbol{n}$ is opposite to gravity, so that $\boldsymbol{n} \cdot \boldsymbol{g} = -g$. Equation (22) is used to linearize the pendulum dynamics, the ZMP being defined in general by the non-linear formula $\boldsymbol{p}_Z \stackrel{\text{def}}{=} \frac{\boldsymbol{n} \times \boldsymbol{\tau}_O}{\boldsymbol{n} \cdot \boldsymbol{f}} + \boldsymbol{p}_O$. Under Equations (22)-(23), the resultant force can be computed from COM and ZMP positions by [11]:

$$\boldsymbol{f} \;=\; \frac{mg}{h} \begin{bmatrix} x_Z - x_G \\ y_Z - y_G \\ 1 \end{bmatrix} \tag{24}$$

where $h = z_Z - z_G$ is the constant difference between ZMP and COM altitudes. This value can be positive or negative: for the sake of exposition, we will take $h > 0$. Injecting Equations (22)-(23) into an inequality constraint (14) of the CWC yields:

$$(\boldsymbol{a}_O + \boldsymbol{a} \times \boldsymbol{p}_G) \cdot \boldsymbol{f} + \boldsymbol{a} \cdot \boldsymbol{0} \;\leq\; 0 \tag{25}$$

Substituting (24) into (25) yields:

$$a_i(x_Z - x_G) + b_i(y_Z - y_G) \;\leq\; h\sigma_{\hat{\boldsymbol{a}}}(\boldsymbol{p}_G) \tag{26}$$

where

$$\begin{bmatrix} a_i \\ b_i \end{bmatrix} \;=\; \begin{bmatrix} a_{Ox} \\ a_{Oy} \end{bmatrix} + \begin{bmatrix} a_y z_G - a_z y_G \\ -a_x z_G + a_z x_G \end{bmatrix} \tag{27}$$

Assuming that the COM lies inside the static-equilibrium polygon,[3] the right-hand side of this expression is positive from (19). The inequality is then expressed in polar form as

$$\mathbf{B}_{\text{ZMP}}(\boldsymbol{p}_G) \begin{bmatrix} x_Z - x_G \\ y_Z - y_G \end{bmatrix} \;\leq\; \mathbf{1} \tag{28}$$

where the origin $(x_Z, y_Z) = (x_G, y_G)$ lies inside the polygon by construction. As in (21), a convex hull algorithm can finally be applied to compute the vertices of the pendular ZMP support area.

D. 3D Volume of COM accelerations

Equation (22) is a limitation of the linear-pendulum mode in that the COM trajectory needs to lie in a plane predefined by the vector $\boldsymbol{n}$. This limitation is all the less grounded that, from Equation (24), controlling the ZMP in this mode is equivalent to controlling the resultant contact force $\boldsymbol{f}$. We therefore propose to directly control this force, or equivalently, to directly control the three-dimensional COM acceleration $\ddot{\boldsymbol{p}}_G = \frac{1}{m}\boldsymbol{f} + \boldsymbol{g}$.

Substituting this acceleration into (25), one gets:

$$(\boldsymbol{a}_O + \boldsymbol{a} \times \boldsymbol{p}_G) \cdot \ddot{\boldsymbol{p}}_G \;\leq\; (\boldsymbol{a}_O + \boldsymbol{a} \times \boldsymbol{p}_G) \cdot \boldsymbol{g} \tag{29}$$

Expanding scalar products, this inequality rewrites to:

$$a_i \ddot{x}_G + b_i \ddot{y}_G - \sigma_{\hat{\boldsymbol{a}}} \ddot{z}_G \;\leq\; g\sigma_{\hat{\boldsymbol{a}}} \tag{30}$$

(We dropped the argument $\boldsymbol{p}_G$ of $\sigma_{\hat{\boldsymbol{a}}}$ to alleviate notations.) Assuming that the COM lies in the interior of the polygon[3] ($\sigma_{\hat{\boldsymbol{a}}} > 0$) and that $\ddot{z}_G > -g$,

$$\left(\frac{a_i}{\sigma_{\hat{\boldsymbol{a}}}}\right) \cdot \frac{\ddot{x}_G}{g + \ddot{z}_G} + \left(\frac{b_i}{\sigma_{\hat{\boldsymbol{a}}}}\right) \cdot \frac{\ddot{y}_G}{g + \ddot{z}_G} \;\leq\; 1 \tag{31}$$

[3]Otherwise, center the polygon on its Chebyshev center as previously.

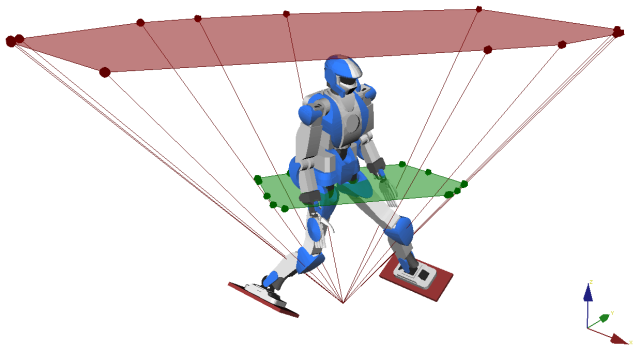

Fig. 2. Static-equilibrium polygon (in green at the altitude of the center of mass) and cone of 3D COM accelerations (in red, with the zero acceleration centered on the COM) in a double-support configuration. For the latter, the scaling from accelerations to positions is 0.08 s^2, and the cone was cut at $\ddot{z}_G = g$ to show its cross-section.

This expression is in polar form $\mathbf{B}_{\text{3D}}(\boldsymbol{p}_G)[\tilde{x}\ \tilde{y}]^\top \leq \mathbf{1}$ for the new coordinates:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} \;=\; \frac{1}{g + \ddot{z}_G} \begin{bmatrix} \ddot{x}_G \\ \ddot{y}_G \end{bmatrix} \tag{32}$$

We can enumerate the vertices $\{(\tilde{x}_i, \tilde{y}_i)\}$ of the corresponding polygon using a convex hull again. For a given vertical acceleration $\ddot{z}_{G,i} > -g$, the COM acceleration coordinates $(\ddot{x}_{G,i}, \ddot{y}_{G,i})$ corresponding to a vertex $(\tilde{x}_i, \tilde{y}_i)$ are:

$$\begin{bmatrix} \ddot{x}_{G,i} \\ \ddot{y}_{G,i} \end{bmatrix} \;=\; (g + \ddot{z}_{G,i}) \begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \end{bmatrix} \tag{33}$$

We recognize the equation of a 3D polyhedral convex cone, pointing upward, with apex located at $(\ddot{x}_G, \ddot{y}_G, \ddot{z}_G) = (0, 0, -g)$ and rays defined by $\boldsymbol{r}_i = [\tilde{x}_i\ \tilde{y}_i\ 1]^\top$. Figure 2 shows the cone in a sample contact configuration.

Overall, we have thus both (1) a geometric characterization and (2) an algorithm to compute the cone of feasible COM accelerations when the angular momentum is regulated to zero.[4] This construction generalizes the pendular support area defined in [11]. Furthermore, as a consequence of Equations (26)-(30), we have the following property:

Proposition 1: The COM is in the interior of the static-equilibrium polygon if and only if the set of feasible COM accelerations (equivalently, whole-body ZMPs) under zero angular momentum contains a neighborhood of the origin.

In other words, the static-equilibrium polygon is not only related to static equilibrium: it is also the set of positions from which the robot can accelerate its center of mass in any direction (with zero angular momentum). When the COM reaches the edge of this polygon, the zero acceleration touches a facet of the acceleration cone. Denoting by $\boldsymbol{d}$ the facet normal, all feasible accelerations $\ddot{\boldsymbol{p}}_G$ are then such that $\boldsymbol{d} \cdot \ddot{\boldsymbol{p}}_G \geq 0$, *i.e.*, $\boldsymbol{d}$ is an "irresistible" direction of motion.

[4]The same derivation can be applied with non-zero angular-momentum references; however, the question of finding such references is still open.

V. Preview control of COM accelerations

Let the control variable $\boldsymbol{u}$ be the COM acceleration $\boldsymbol{u} := \ddot{\boldsymbol{p}}_G$. The discretized COM dynamics with sampling ΔT are:

$$\boldsymbol{x}(k+1) \quad = \quad \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}\boldsymbol{u}(k) \tag{34}$$

where $\boldsymbol{x}(k) = [\boldsymbol{p}_G(k\,\Delta T)^\top \ \dot{\boldsymbol{p}}_G(k\,\Delta T)^\top]^\top$ and, denoting by $\mathbf{E}_3$ the 3×3 identity matrix,

$$\mathbf{A} \ = \ \begin{bmatrix} \mathbf{E}_3 & \Delta T\mathbf{E}_3 \\ \mathbf{0}_{3\times 3} & \mathbf{E}_3 \end{bmatrix} \quad \mathbf{B} \ = \ \begin{bmatrix} \frac{1}{2}\Delta T^2\mathbf{E}_3 \\ \Delta T\mathbf{E}_3 \end{bmatrix} \tag{35}$$

At each control step, a preview controller receives the current state $\boldsymbol{x}_0 = (\boldsymbol{p}_0, \dot{\boldsymbol{p}}_0)$ and computes a sequence of controls $\boldsymbol{u}(0), \ldots, \boldsymbol{u}(N)$ driving the system from $\boldsymbol{x}_0$ to $\boldsymbol{x}(N)$ at the end of the time horizon $T = N\Delta T$ of the preview window. By recursively applying (34), $\boldsymbol{x}(k)$ can be written as a function of $\boldsymbol{x}_0$ and of $\boldsymbol{u}(0), \ldots, \boldsymbol{u}(k-1)$ (see *e.g.*, [5]):

$$\boldsymbol{x}(k) \quad = \quad \boldsymbol{\Phi}_k\boldsymbol{x}_0 + \boldsymbol{\Psi}_k\mathbf{U}(k-1) \tag{36}$$

$$\mathbf{U}(k) \quad = \quad [\boldsymbol{u}(0)^\top \ \cdots \ \boldsymbol{u}(k)^\top]^\top \tag{37}$$

A necessary condition for contact-stability throughout the trajectory is that all accelerations $\boldsymbol{u}(k)$ lie in the COM-acceleration cone $\mathcal{C}(\boldsymbol{p}_G(k\Delta T))$. Expanding its inequalities (29) for an arbitrary twist $\hat{\boldsymbol{a}} \in \mathrm{CWC}$ yields:

$$\boldsymbol{p}_G(k\,\Delta T)^\top[-\boldsymbol{a}\times](\boldsymbol{u}(k) - \boldsymbol{g}) + \boldsymbol{a}_O^\top(\boldsymbol{u}(k) - \boldsymbol{g}) \leq 0 \tag{38}$$

Let L denote the number of twists in the CWC. By stacking up the N inequalities (38), we get in more concise form:

$$\boldsymbol{p}_G(k\,\Delta T)^\top\mathbb{A}_\times(\boldsymbol{u}(k) - \boldsymbol{g}) + \mathbf{A}'_O(\boldsymbol{u}(k) - \boldsymbol{g}) \leq \mathbf{0} \tag{39}$$

where $\mathbb{A}_\times$ is a $3 \times L \times 3$ tensor, and $\mathbf{A}'_O$ consists of the first three columns of $\mathbf{A}_O$. Combining (36) and (39), this condition yields a set of quadratic inequality constraints of the form:

$$\forall k < N, \ \mathbf{U}(k)^\top\mathbb{P}_k\mathbf{U}(k) + \mathbf{Q}_k\mathbf{U}(k) + \boldsymbol{l}_k \leq \mathbf{0} \tag{40}$$

where $\mathbb{P}_k$ is a $3(k+1) \times L \times 3(k+1)$ tensor, $\mathbf{Q}_k$ is a $L \times 3(k+1)$ matrix $\boldsymbol{l}_k$ is an L-dimensional vector. One can thus formulate the preview control problem as a Quadratically Constrained Quadratic Program (QCQP). Although a QCQP formulation was successfully applied for walking on even terrains with variable COM height [10], we chose not to do so for the following reasons:

- QCQP is a harder class of problems than Quadratic Programming (QP), especially when inequality constraints are not positive-semidefinite [10] so that the problem non-convex. Real-timeness implies that only a small number of SQP iterations can be run in the control loop (two in [10]), thus with no convergence guarantee.
- Constraints (40) are given without any redundancy elimination. In practice, eliminating redundancy (in our case, by applying a convex hull algorithm) significantly reduces the number of inequality constraints (Table I).

Instead, we propose a trajectory-wide contact-stability criterion that yields linear inequality constraints, and for which we can apply the convex-hull reduction from Section IV.

A. Robust trajectory-wide contact-stability criterion

We want to drive the COM from $\boldsymbol{x}_0 = (\boldsymbol{p}_0, \dot{\boldsymbol{p}}_0)$, its current state, to a goal position $\boldsymbol{x}_T$ through a trajectory $t \in [0, T] \mapsto \boldsymbol{p}_G(t)$. Due to the initial velocity $\dot{\boldsymbol{p}}_G(0)$ and real-world uncertainties, the trajectory will not be exactly a line segment $[\boldsymbol{p}_0, \boldsymbol{p}_T]$. Yet, we assume that it lies within a polyhedral "tube" $\mathcal{T} = \mathit{conv}(\{\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_q\})$ containing $[\boldsymbol{p}_0, \boldsymbol{p}_T]$. A point $\boldsymbol{p}_G \in \mathcal{T}$ can be written as a convex combination $\boldsymbol{p}_G = \sum_{i=1}^q \alpha_i\boldsymbol{\nu}_i$, where the α_i's are positive and sum-up to one, so that the inequality (39) becomes:

$$\sum_{i=1}^{q} \alpha_i(\boldsymbol{\nu}_i^\top\mathbb{A}_\times + \mathbf{A}'_O)(\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) \ \leq \ \mathbf{0} \tag{41}$$

A particular way to enforce the negativity of a convex combination is to ensure that all of its terms are negative. Thus, a sufficient condition for the satisfaction of (41) is

$$\mathbf{C}_\mathcal{T}(\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) \leq \mathbf{0}, \quad \mathbf{C}_\mathcal{T} := \begin{bmatrix} \boldsymbol{\nu}_1^\top\mathbb{A}_\times + \mathbf{A}'_O \\ \vdots \\ \boldsymbol{\nu}_q^\top\mathbb{A}_\times + \mathbf{A}'_O \end{bmatrix} \tag{42}$$

which is the halfspace representation of the intersection

$$I_\mathcal{T} \ := \ \mathcal{C}(\boldsymbol{\nu}_1) \cap \cdots \cap \mathcal{C}(\boldsymbol{\nu}_q) \ \subseteq \ \mathcal{C}(\boldsymbol{p}_G) \tag{43}$$

Proposition 2: $I_\mathcal{T}$ is the set of COM accelerations that are feasible everywhere in $\mathcal{T}$, *i.e.*,

$$I_\mathcal{T} = \bigcap_{\boldsymbol{p}\in\mathcal{T}} \mathcal{C}(\boldsymbol{p}) \tag{44}$$

Proof: We showed that $\ddot{\boldsymbol{p}}_G \in I_\mathcal{T}$ is feasible at any position $\boldsymbol{p}_G \in \mathcal{T}$ (by construction, (42) $\Rightarrow$ (41) $\Rightarrow$ (38)). Conversely, suppose that $\ddot{\boldsymbol{p}}_G$ is feasible for all $\boldsymbol{p}_G \in \mathcal{T} = \mathit{conv}(\{\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_k\})$. In particular, (41) holds when $\boldsymbol{p}_G$ is equal to any vertex $\boldsymbol{\nu}_j$. Thus, $\ddot{\boldsymbol{p}}_G \in \mathcal{C}(\boldsymbol{\nu}_j)$, and since the index j can be chosen arbitrarily, $\ddot{\boldsymbol{p}}_G \in \cap_j\mathcal{C}(\boldsymbol{\nu}_j) = I_\mathcal{T}$. ■

A trajectory $\boldsymbol{p}_G(t)$ is *contact-stable within* $\mathcal{T}$ if it satisfies:

$$\forall t \in [0, T], \ \boldsymbol{p}_G(t) \in \mathcal{T} \text{ and } \ddot{\boldsymbol{p}}_G(t) \in I_\mathcal{T} \tag{45}$$

This condition can be compared with the previous trajectory-wide stability criterion from [17], where trajectories were computed such that

$$\forall t \in [0, T], \ \ddot{\boldsymbol{p}}_G(t) \in \mathcal{C}(\boldsymbol{p}_G(t)) \tag{46}$$

The implicit assumption behind such a strategy is that either trajectory-tracking is perfect, or small deviations $\delta\boldsymbol{p}_G$ from the reference $\boldsymbol{p}_G(t)$ can be coped with if $\ddot{\boldsymbol{p}}_G$ lies "inside enough" of its cone. By taking $\mathcal{T}$ around the trajectory, we explicitly model how far $\ddot{\boldsymbol{p}}_G$ needs to be inside the cone to cope with a set of deviations $\delta\boldsymbol{p}_G$. In this sense, our criterion is a *robust* condition for trajectory-wide contact stability, with explicit modelling of the robustness margin.

Although the tube-wise intersection formalized by Equation (44) seems like a severely restrictive condition, we noticed (to our surprise) that the acceleration cones left after intersection are still sufficient for locomotion in challenging scenarios (see *e.g.*, Section VI).

TABLE I

EFFECT OF THE DUAL CONVEX-HULL REDUCTION ON THE SIZE OF THE INEQUALITY-CONSTRAINT MATRIX $\mathbf{C}_\mathcal{T}$. AVERAGES ARE GIVEN ON 26 MATRICES (ONE FOR EACH STEP OF THE CIRCULAR STAIRCASE).

	$\lvert\mathbf{C}_\mathcal{T}\rvert$	$\lvert\mathbf{C}'_\mathcal{T}\rvert$
Single support	376 ± 226	6 ± 2
Double support	484 ± 221	6 ± 2

B. Integration with preview control

At each new control step, the current state $\boldsymbol{x}_0 = (\boldsymbol{p}_0, \dot{\boldsymbol{p}}_0)$ of the center of mass feeds the preview controller. We define the target state at the end of the time horizon by $\boldsymbol{x}_T = (\boldsymbol{p}_T, \boldsymbol{v}_T)$, where $\boldsymbol{p}_T$ is determined from the current stance in the step sequence, $\boldsymbol{v}_T$ is a reference velocity of 0.4 m.s^{-1} oriented in the direction of motion, and T is calculated from the time remaining in the current gait phase. From there, our method goes as follows:

1) Define $\mathcal{T}$ containing $[\boldsymbol{p}_0, \boldsymbol{p}_T]$ and compute its half-space representation $\mathbf{P}_\mathcal{T}$ using the double-description. In practice, we defined $\mathcal{T}$ by a polyhedral cylinder centered on $[\boldsymbol{p}_0, \boldsymbol{p}_T]$ with square cross-sections and a robustness radius of 5 cm.
2) Compute the halfspace representation $\mathbf{C}_\mathcal{T}$ of $I_\mathcal{T}$ from the contact wrench cone $\mathbf{A}_O$ (which is computed only once per support phase).
3) Reduce $\mathbf{C}_\mathcal{T}$ to polar form $\mathbf{B}_\mathcal{T}[x\ y]^\top \leq \mathbf{1}$ and compute its vertex representation $\boldsymbol{g} + \mathit{rays}(\{\boldsymbol{r}_i\})$ using a convex hull algorithm[5] (Section IV-D).
4) Compute its non-redundant halfspace-representation $\mathbf{C}'_\mathcal{T}$ using again the double-description.

Next, we formulate the preview control problem as a quadratic program:

$$\min_{\mathbf{U}} \quad : \quad \|\boldsymbol{x}(N) - \boldsymbol{x}_T\|^2 + \epsilon\|\mathbf{U}\|^2 \tag{47}$$

$$\text{s.t.} \quad : \quad \forall k,\ \mathbf{C}'_\mathcal{T}\boldsymbol{u}(k) \leq \mathbf{C}'_\mathcal{T}\boldsymbol{g} \tag{48}$$

$$\forall k,\ \mathbf{P}_\mathcal{T}\boldsymbol{x}(k) \leq \mathbf{1} \tag{49}$$

where $\mathbf{U} \stackrel{\text{def}}{=} \mathbf{U}(N-1)$ is the stacked vector of controls from which all $\boldsymbol{x}(k)$ and $\boldsymbol{u}(k)$ derive by Equation (36). The inequality constraints (48) and (49) are a linear decoupling of (40) via polyhedral bounds $\mathcal{T}$.

As a matter of fact, using polyhedral bounds can be thought of as a general linearization technique. In [8], a linear decoupling was also obtained by bounding vertical COM accelerations. Similarly, polytopes of robust COM positions $\boldsymbol{p}_G$ were obtained in [17], [19] by defining polyhedral bounds on disturbances $\boldsymbol{\epsilon}$, thus eliminating the bilinear coupling between $\boldsymbol{p}_G$ and $\boldsymbol{\epsilon}$.

Eliminating redundancy in the pipeline above is a significant computational step. From one of our experiments (Table I), the number of lines $\lvert\mathbf{C}'_\mathcal{T}\rvert$ in $\mathbf{C}'_\mathcal{T}$ is two orders of magnitude smaller than that of $\mathbf{C}_\mathcal{T}$. Given that the number of inequalities (48) in the above QP is $N\lvert\mathbf{C}'_\mathcal{T}\rvert$, this makes the difference in practice between solving a problem of size 100 versus $10,000$ (we use $N = 10$ steps).

[5]We used *Qhull* [27], available from `http://www.qhull.org/`

TABLE II

BREAKDOWN OF COMPUTATION TIMES INSIDE THE PREDICTIVE CONTROLLER, AVERAGED OVER 2000 CALLS IN EXPERIMENT VI.B.

Function	Output	Time (ms)
Double description of $\mathcal{T}$ (SS and DS)	$\mathbf{P}_\mathcal{T}$	0.3 ± 0.1
H-representation of $I_\mathcal{T}$ (SS and DS)	$\mathbf{C}_\mathcal{T}$	0.2 ± 0.1
Convex hull of $I_\mathcal{T}$ (SS and DS)	$\mathbf{C}'_\mathcal{T}$	2.4 ± 1.2
Other matrix operations	–	0.4 ± 0.4
Solving final QP ($N = 10$)	$\mathbf{U}$	0.2 ± 0.1
Total		3.5 ± 2.1

C. Locomotion state machine

Our preview controller is applied to the HRP-4 humanoid in various environments. The inputs to the controller are the current COM position $\boldsymbol{p}_0$ and velocity $\dot{\boldsymbol{p}}_0$, the preview time horizon T, a target COM position $\boldsymbol{p}_T$, $\mathcal{T}$ and its acceleration cone $I_\mathcal{T}$. These computation of these inputs is supervised by a Finite State Machine (FSM) that cycles between four phases $\varphi \in \{\texttt{SS-L}, \texttt{DS-R}, \texttt{SS-R}, \texttt{DS-L}\}$, where SS (resp. DS) stands for single-support (resp. double-support), while L and R indicate that the phase ends on the left and right foot, respectively. Each phase is thus associated with a unique foot contact. The target COM position $\boldsymbol{p}^*_G(\varphi)$ of a phase is then taken 0.8 m above this contact.

Phase durations are set to $T_\texttt{SS} = 1$ s for single-support and $T_\texttt{DS} = 0.5$ s for double-support. At each iteration of the control loop, the input to the preview controller is decided based on the time T_rem remaining until the next phase transition. Let us denote by φ the current phase in the FSM and φ' the phase after φ. We define preview targets by:

1) if φ is single-support and $T_\text{rem} < \frac{1}{2}T_\texttt{SS}$:
   - $T \leftarrow T_\text{rem} + T_\texttt{DS} + \frac{1}{2}T_\texttt{SS}$
   - $\boldsymbol{p}_G \leftarrow \boldsymbol{p}^*_G(\varphi'')$
2) otherwise, if φ is double-support:
   - $T \leftarrow T_\text{rem} + \frac{1}{2}T_\texttt{SS}$
   - $\boldsymbol{p}_G \leftarrow \boldsymbol{p}^*_G(\varphi)$
3) otherwise (φ is single-support and $T_\text{rem} > \frac{1}{2}T_\texttt{SS}$):
   - $T \leftarrow T_\text{rem}$
   - $\boldsymbol{p}_G \leftarrow \boldsymbol{p}^*_G(\varphi)$

Case 1) switches the target of the preview controller to the next staircase step in the middle of single-support phases[6], which forces the robot to start its next step while allowing it to re-use the kinetic momentum in the direction of motion.

Note that cases 1) and 2) imply contact switches in the middle of preview trajectories, which different cones $\mathbf{C}'_\mathcal{T}$ depending on the step k in the preview problem. To take this into account, we compute the switching step $k_\text{rem} = T_\text{rem}/\Delta T$ along with two tubes $\mathcal{T}_\texttt{SS} \subset \mathcal{T}_\texttt{DS}$ and their dual cones (computations between these two overlapping tubes can be factored; see [1] for details). The corresponding

[6]The same behavior is present in [14]: if the control from Figure 5 of this paper was followed to the end, the COM velocity would go to zero between each step, which is not the behavior observed in Figures 7 and 8.

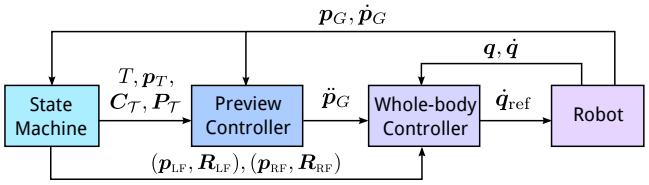

Fig. 3. Overview of our control pipeline. The COM trajectory polyhedron $\mathbf{P}_\mathcal{T}$ and its corresponding contact-stability cone $\mathbf{C}_\mathcal{T}$ are computed using the method described in Section V.

matrices $(\mathbf{C}'_{\mathcal{T}_{\text{SS}}}, \mathbf{P}_{\mathcal{T}_{\text{SS}}})$ and $(\mathbf{C}'_{\mathcal{T}_{\text{DS}}}, \mathbf{P}_{\mathcal{T}_{\text{DS}}})$ are then respectively used in Equations (48)-(49) for $k \leq k_{\text{rem}}$ and $k > k_{\text{rem}}$.

A breakdown of computation times inside the overall predictive controller is reported in Table II. All computations were run on an average laptop computer (Intel(R) Core(TM) i7-6500U CPU @ 2.50 Ghz).

D. Whole-body controller

The last step of the pipeline is to convert task objectives, such as COM or foot positions, into joint commands sent to motor controllers. For this, we used our own solver implemented in the *pymanoid* library.[7] It solves a single quadratic program on five weighted tasks (see [11] for details), by decreasing priority: support foot, swing foot, COM tracking, constant angular-momentum, and posture tracking for regularization. The corresponding task weights were respectively set to $(10^4, 10^2, 10, 1, 10^{-1})$. Each QP solution provides joint-angle velocities $\dot{\boldsymbol{q}}_{\text{ref}}$, which is then sent to the robot. See Figure 3 for a summary of our pipeline.

VI. Experiments

The ground truth for contact stability is the existence of feasible contact forces $\boldsymbol{f}_{\text{all}}$ summing up to the net wrench $\hat{\boldsymbol{w}}$ of the motion. In all experiments, we validate our trajectories by checking the existence of such $\boldsymbol{f}_{\text{all}}$ at each time instant. In both experiments, friction coefficients were set to $\mu = 0.7$.

A. Regular staircase and walking into an aircraft

Our first scenario, provided by Airbus Group, takes place in a 3D realistic model of scale 1:1 for a section of the A350 airplane. This mock-up will be used for experiments with the real robot when the research matures to an integrated software, and gets approval from Airbus Group. In order to access a predefined spot in the A350, the robot needs to climb stairs (size of those available in the factory), walk on a platform (flat ground) to finally reach the mockup floor composed of removable tiles that can be uneven and disposed in various locations, see Figure 4. In this simulation, the footprints were given together with the timing for the DS and SS phases (respectively 0.5 s and 1 s).

B. Slanted circular staircase with tilted steps

The slanted circular staircase depicted in Figure 1 has 26 steps randomly rolled, pitched and yawed by angles $(\theta_r, \theta_p, \theta_y) \in [-0.5, +0.5]^3$ rad. The average radius of the

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

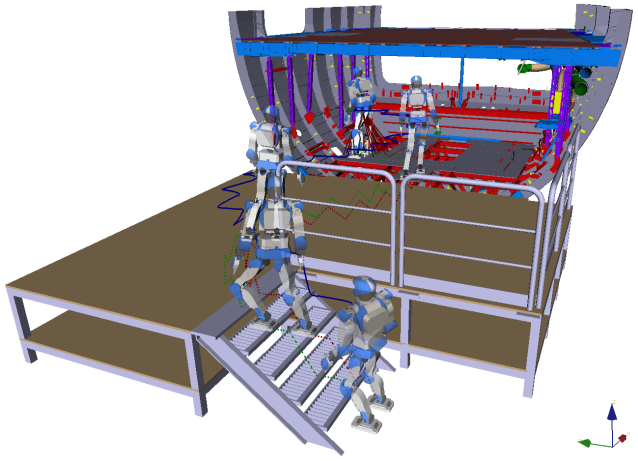

Fig. 4. HRP-4 accessing the floor-shop of an A350 through stairs and reach the working areas by walking on tiles. Foot trajectories (dashed lines) step over two staircase steps at a time, as done in natural stair climbing. The COM trajectory (blue line) illustrates the progression of the robot until its target configuration inside the aircraft. Contact stability of the whole motion was cross-validated by checking the existence of groundtruth contact forces $\boldsymbol{f}_{\text{all}}$ at each timestep, as shown in the accompanying video [1].

staircase is 1.4 m, and the altitude difference between the highest and lower steps is also 1.4 m.

For this scenario, the reference durations of single and double support phases were set to $T_{\text{SS}} = 1$ s and $T_{\text{DS}} = 0.5$ s, respectively. We concur with [5] that the question of finding proper timings becomes crucial in multi-contact. Having constant durations overlooks the fact that some steps are harder to take than others (due to their respective tilting, altitude difference, etc.).[8] Being unable to find a single pair of constants $(T_{\text{SS}}, T_{\text{DS}})$ suited to the whole staircase, we opted for the following workaround condition:

W) At the end of a double-support phase, wait for the COM to be above the static-equilibrium polygon of the next single-support before activating the phase transition.

This choice is motivated by the link (30) between the 3D cone of COM accelerations and the position of the COM in the static-equilibrium polygon. In practice, it allows the use of "optimistic" values of $(T_{\text{SS}}, T_{\text{DS}})$ while only extending T_{DS} when necessary.

VII. Conclusion

We presented a multi-contact walking pattern generator based on preview-control of the 3D acceleration of the center of mass. Our development builds upon algebraic manipulations of friction cones as dual twists, thanks to which we can recompute 3D cones of feasible COM accelerations in real-time. We then showed how to intersect these cones over the preview window of a model-preview controller to construct a conservative trajectory-wide contact-stability criterion. We implemented this pipeline and illustrated it with the HRP-4 humanoid model in multi-contact dynamically walking scenarios. All our source code is released at [1].

[8] This question is less critical for walking on horizontal or well structured floors, where all steps are similar.

ACKNOWLEDGMENT

The authors would like to warmly thank Komei Fukuda for his helpful feedback and Patrick Wensing for pointing out a calculation mistake in a preliminary version of the paper. This paper has also benefitted from rich discussions with Hervé Audren and Adrien Escande.

REFERENCES

[1] [Online]. Available: https://github.com/stephane-caron/3d-mpc
[2] J. Englsberger, C. Ott, and A. Albu-Schaffer, "Three-dimensional bipedal walking control based on divergent component of motion," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.
[3] Y. Zhao, B. Fernandez, and L. Sentis, "Robust phase-space planning for agile legged locomotion over various terrain topologies," in *Robotics: Science and System*, 2016.
[4] H. Hirukawa, S. Hattori, S. Kajita, K. Harada, K. Kaneko, F. Kanehiro, M. Morisawa, and S. Nakaoka, "A pattern generator of humanoid robots walking on a rough terrain," in *IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 2181–2187.
[5] H. Audren, J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida, "Model preview control in multi-contact motion-application to a humanoid robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4030–4035.
[6] A. Herzog, N. Rotella, S. Schaal, and L. Righetti, "Trajectory generation for multi-contact momentum control," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, pp. 874–880.
[7] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3555–3561.
[8] 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 *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, pp. 595–601.
[9] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères, "A reactive walking pattern generator based on nonlinear model predictive control," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 10–17, 2017.
[10] K. Van Heerden, "Real-time variable center of mass height trajectory planning for humanoids robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 135–142, 2017.
[11] S. Caron, Q.-C. Pham, and Y. Nakamura, "ZMP support areas for multi-contact mobility under frictional constraints," *IEEE Transactions on Robotics*, to appear.
[12] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.
[13] S. Caron, Q.-C. Pham, and Y. Nakamura, "Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015.
[14] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 2003, pp. 1620–1626.
[15] I. Mordatch, M. De Lasa, and A. Hertzmann, "Robust physics-based locomotion using low-dimensional planning," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 29, no. 4, p. 71, July 2010.
[16] Z. Qiu, A. Escande, A. Micaelli, and T. Robert, "Human motions analysis and simulation based on a general criterion of stability," in *International Symposium on Digital Human Modeling*, 2011.
[17] S. Caron, Q.-C. Pham, and Y. Nakamura, "Leveraging cone double description for multi-contact stability of humanoids with applications to statics and dynamics," in *Robotics: Science and System*, 2015.
[18] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 794–807, 2008.
[19] A. Del Prete, S. Tonneau, and N. Mansard, "Fast algorithms to test robust static equilibrium for legged robots," in *IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, May 2016.
[20] T. Zhang, S. Caron, and Y. Nakamura, "Humanoid stair climbing based on dedicated plane segment estimation and multi-contact motion generation," *to appear in: International Journal of Humanoid Robotics*, 2016.
[21] K. Fukuda and A. Prodon, "Double description method revisited," in *Combinatorics and computer science*. Springer, 1996, pp. 91–111.
[22] K. Bouyarmane, A. Escande, F. Lamiraux, and A. Kheddar, "Potential field guide for humanoid multicontacts acyclic motion planning," in *IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1165–1170.
[23] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.
[24] D. Avis and K. Fukuda, "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra," *Discrete & Computational Geometry*, vol. 8, no. 3, pp. 295–313, 1992.
[25] D. G. Kirkpatrick and R. Seidel, "The ultimate planar convex hull algorithm?" *SIAM Journal on Computing*, vol. 15, no. 1, pp. 287–299, 1986.
[26] S. Boyd and L. Vandenberghe, *Convex Optimization*, 2004.
[27] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.
[28] Q.-C. Pham and O. Stasse, "Time-optimal path parameterization for redundantly-actuated robots: A numerical integration approach," *IEEE/ASME Transactions on Mechatronics*, 2015.

APPENDIX

We compare the convex hull reduction to the original calculation [18] of the static-equilibrium polygon. Computation times for randomly sampled contact configurations are reported in Table III for four algorithms:

- *cdd + hull*: the method described in Section IV-B, where *cdd* [21] is used to compute the CWC while convex hulls are computed with *Qhull* [27].
- *Parma + hull*: same approach, using the Parma Polyhedra Library[9] rather than *cdd* to compute the CWC.
- *cdd only*: as described in [20], *cdd* can also be used to compute the static-equilibrium polygon directly.
- *Bretl & Lall*: the algorithm from [18], in the implementation from [28] but using GLPK as LP solver.[10]

The *Parma + hull* solution is the slowest but most numerically stable, while *cdd only* is only competitive in single-support. Neck to neck are *Bretl & Lall* and *cdd + hull*, with the latter faster in single- and double-support. But the real benefit of our approach comes with the computation of time-varying criteria: in double- and triple-support, we see that executing *hull only* is more than ten times faster than applying any other algorithm from scratch. We also highlight it as the fastest solution for single-support, as in this case the CWC is known analytically [13] and there is no need for the *cdd* step.

TABLE III

TIME (IN MS) TO COMPUTE THE STATIC-EQUILIBRIUM POLYGON, AVERAGED OVER 100 RANDOM CONTACT CONFIGURATIONS.

Algorithm	Single support	Double support	Triple support
Parma + hull	6.02 ± 0.20	21.0 ± 4.2	42 ± 11
cdd only	0.38 ± 0.01	7.0 ± 2.7	> 500
Brel & Lall [18]	1.00 ± 0.02	3.1 ± 0.8	$\mathbf{5.9 \pm 1.6}$
cdd + hull	0.60 ± 0.01	$\mathbf{2.7 \pm 0.6}$	7.1 ± 1.9
hull only	$\mathbf{0.17 \pm 0.003}$	0.28 ± 0.04	0.38 ± 0.09

[9] https://github.com/haudren/pyparma

[10] Using an efficient LP solver is crucial here: in a preliminary version of this paper, we used the more general CVXOPT, which resulted in computations around $10\times$ slower than those we now report using GLPK.