Fast Algorithms to Test Robust Static Equilibrium for Legged Robots

Andrea del Prete, Steve Tonneau, Nicolas Mansard

Fast Algorithms to Test Robust Static Equilibrium for Legged Robots

Andrea Del Prete, Steve Tonneau and Nicolas Mansard

***Abstract*— Maintaining equilibrium is of primary importance for legged systems. It is not surprising then that static equilibrium is at the core of most control/planning algorithms for legged robots. Being able to check whether a system is in static equilibrium is thus important, and doing it *efficiently* is crucial. While this is straightforward for a system in contact with a flat ground only, it is not the case for arbitrary contact geometries. In this paper we propose two new techniques to test static equilibrium and we show that they are computationally faster than all other existing methods. Moreover, we address the issue of robustness to errors in the contact-force tracking, which could lead to slippage or rotation at the contacts. We extend all the discussed techniques to test for robust static equilibrium, that is the ability to maintain equilibrium while avoiding to lose contacts despite bounded force-tracking errors. Accounting for robustness does not affect the computation time of the equilibrium tests, while it qualitatively improves the contact postures generated by our reachability-based multi-contact planner.**

I. INTRODUCTION

The problem of equilibrium is central for planning, control and analysis of legged robots [1]. Control algorithms for biped and quadruped robots exploit static equilibrium criteria to avoid falls [2], [3]. Motion planning algorithms make large use of static equilibrium tests to verify that the system is able to maintain balance at each generated configuration [4], [5]. This quasi-static approach to motion planning can be the first step for the generation of dynamic movements through time-scaling techniques [6], [7].

For robots making contact on a flat ground only, static equilibrium can be achieved if and only if the vertical projection of the center of mass (CoM) lies inside the convex hull of the contact points [8] (neglecting the joint-torque limits). However, for arbitrary contact geometries this condition is neither sufficient nor necessary and more complex and computationally-expensive techniques are needed to check static equilibrium [9], [7]. Regardless of the application, the computational efficiency of the equilibrium tests is critical, as computation time is often the bottleneck of planning and control algorithms.

Depending on the number of tests that need to be performed at each contact configuration, different test techniques have been identified as being the most efficient [10]. In particular, control algorithms typically need to perform thousands of tests since robots usually maintain the same contact configuration for several seconds, and the controller tests equilibrium 100-1000 times per second. At the other end of the scale we have *motion-before-contact* planners, which compute first the trajectory of the robot trunk and then try to find the contacts that stabilize that trajectory [11]. Because of their working principle these algorithms typically perform very few tests (< 5) for each contact configuration. In between we have *contact-before-motion* planners, which first find the contact points and then generate the motion of the robot along these contacts [5]. These planners typically perform a higher number of equilibrium tests for each contact configuration, in the order of 10-100.

For application on real systems the problem of equilibrium needs to be coupled with the problem of robustness: the robot needs to maintain equilibrium despite some bounded uncertainties. Two robustness measures have been proposed: the capacity of the system to accelerate its CoM [7] or to generate wrench at its CoM [12], [13]. While being interesting, these robustness measures do not seem easy to apply on a real system: how much CoM acceleration/wrench must the system be able to generate in order not to fall?

This paper starts by recalling in Section II the problem of equilibrium[1] and the classic methods to test it. We then propose three contributions. In Section III, we propose a more efficient version of the algorithm proposed by Bretl and Lall [10] (up to 3 times faster). Then in Section IV we show that the classic "support-polygon" technique to test equilibrium on flat ground can also be applied to the more general class of *quasi-flat* contacts. Finally, in Section V we extend all the discussed techniques for testing equilibrium to account for a user-specified degree of robustness to contact-force tracking errors. Contrary to previous works [7], [12], [13], our robustness measure is easy to relate to the tracking capability of the force controller of the robot. Section VI collects all the simulation results, which validate the three contributions through three different tests. To demonstrate the importance of robustness we show in the accompanying video the qualitative difference between robust and non-robust motions generated with a reachability-based planner [11].

II. STATIC EQUILIBRIUM

This section recalls the fundamental equations to model the equilibrium of a floating system in contact and the main algorithms to test it. This long summary is necessary as it introduces the notations and the methods that we will test in the experimental section.

*This work was supported by Euroc (FP7 Grant Agreement 608849), Entracte (ANR Grant Agreement 13-CORD-002-01) and Koroibot (FP7 Grant Agreement 611909).

The authors are with CNRS, LAAS, 7 avenue du colonel Roche, and Univ. de Toulouse, F-31400 Toulouse, France {adelpret,stonneau,nmansard}@laas.fr

[1] In the rest of the paper, unless mentioned otherwise, equilibrium will always refer to static equilibrium.

A. Centroidal Dynamics

Considering a robot in contact with the environment at k contact points, its Newton-Euler equations are:

$$m\ddot{c} = \sum_{i=1}^{k} f_i + mg \tag{1a}$$

$$\dot{L} = \sum_{i=1}^{k} (p_i - c) \times f_i \tag{1b}$$

where m is the robot mass, $c \in \mathbb{R}^3$ is the CoM position, $f_i \in \mathbb{R}^3$ is the i-th contact force, $g = \begin{bmatrix} 0 & 0 & -9.81 \end{bmatrix}^\top$ is the gravity acceleration, $L \in \mathbb{R}^3$ is the angular momentum (expressed at the CoM) and $p_i \in \mathbb{R}^3$ is the i-th contact point. All quantities are expressed in an arbitrary inertial frame having z aligned with the gravity. Each contact force is constrained to lie inside a friction cone:

$$\sqrt{(t_{i1}^\top f_i)^2 + (t_{i2}^\top f_i)^2} \leq \mu_i n_i^\top f_i \qquad \forall i,$$

where μ_i is the friction coefficient, $t_{i1}, t_{i2} \in \mathbb{R}^3$ are the tangent directions and $n_i \in \mathbb{R}^3$ is the normal direction at the i-th contact. In the following we will denote this condition as $f \in \mathcal{K}$. By replacing $c \times \sum_{i=1}^{k} f_i$ with $mc \times (\ddot{c} - g)$ in (1b) we can reformulate (1) in matrix form:

$$\underbrace{\begin{bmatrix} m(\ddot{c} - g) \\ mc \times (\ddot{c} - g) + \dot{L} \end{bmatrix}}_{w} = \underbrace{\begin{bmatrix} I_3 & \dots & I_3 \\ \hat{p}_1 & \dots & \hat{p}_k \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} f_1 \\ \vdots \\ f_k \end{bmatrix}}_{f}, \tag{2}$$

where $w \in \mathbb{R}^6$ is the so-called gravito-inertial wrench (GIW) [7] and $\hat{p} \in \mathbb{R}^{3\times3}$ is the cross-product matrix related to p.

B. Conditions of Static Equilibrium

From (2), the GIW needed for equilibrium is:

$$w_0 = \underbrace{\begin{bmatrix} 0_{3\times3} \\ m\hat{g} \end{bmatrix}}_{D} c + \underbrace{\begin{bmatrix} -mg \\ 0 \end{bmatrix}}_{d} \tag{3}$$

The wrench w_0 is independent of c^z because of the cross product matrix $\hat{g}$ inside D, so we can rewrite (3) as:

$$w_0 = D_{xy} c^{xy} + d$$

In this paper we are interested in characterizing the region of $\mathbb{R}^3$ where the CoM can stand while allowing for equilibrium, that is:

$$\mathcal{S} = \{c \in \mathbb{R}^3 : \exists f \in \mathbb{R}^{3k} \text{ s.t. } w_0(c) = Af, f \in \mathcal{K}\}$$

Since equilibrium is independent of c^z, $\mathcal{S}$ is a generalized cylinder whose axis is parallel to gravity. We call its section the *support section* and denote it by $\mathcal{Y}$ (it is often named support polygon, while in general it is a rounded and possibly-unbounded shape). To test equilibrium, it is sufficient to check whether $c^{xy} \in \mathcal{Y}$.

C. SOCP and Linear Approximation

In the general case testing equilibrium requires the resolution of a Second-Order Cone Program [9]. A very common and computationally-faster alternative—which we adopt as well—is to approximate the friction cones with polytopes [14], [13], [7], [10], [12]. For safety reasons inner approximations are often preferred. We can express the linearized friction-cone constraints as a set of linear inequalities:

$$Bf \leq 0 \tag{4}$$

D. Linear Programming (LP)

The simplest way to test equilibrium for a given CoM position c is to look for contact forces that are inside the (linearized) friction cones and compensate for gravity. This amounts to solving this feasibility Linear Program (LP):

$$\begin{aligned} \text{find} \quad & f \in \mathbb{R}^{3k} \\ \text{subject to} \quad & Bf \leq 0 \\ & Af = D_{xy} c^{xy} + d \end{aligned} \tag{5}$$

The CoM position c allows for equilibrium if and only if (5) has a solution.

E. Polytope Projection (PP)

Another approach to test equilibrium is to compute $\mathcal{S}$ and check whether the interested CoM positions belong to it. Considering linear approximations of the friction cones, $\mathcal{S}$ becomes a polytope, which we can describe by linear inequalities on c. This solution is introduced in [13] and further described in [7]. The proposed algorithm relies on polytope-projection techniques [15] to compute the cone of feasible GIW, expressed as a matrix H such that:

$$Hw \leq 0 \iff \exists f : Bf \leq 0, w = Af$$

The method is built on the observation that a linear cone $\mathcal{P} \subseteq \mathbb{R}^n$ can be equally represented as:

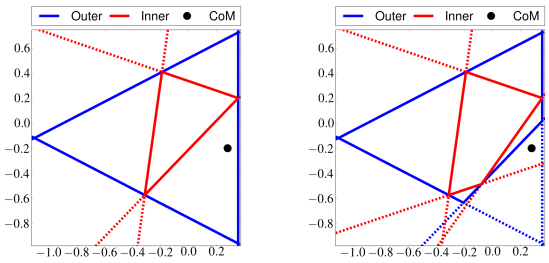
$$\mathcal{P} = \{x \in \mathbb{R}^n : P_f x \leq 0\} = \{P_s \lambda, \lambda \geq 0\}$$

Where the matrices P_f and P_s are respectively named the face and the span descriptions of $\mathcal{P}$. Conversion between face and span form is possible using dedicated numerical algorithms [15] with a significant computational cost. The standard approach for computing H is to [7], [13]:

- convert the force friction cones from face form to span form: $B \to V_f$;
- map the force generators to the GIW space: $V_w = AV_f$;
- convert the GIW cone from span form to face form $V_w \to H$.

Note that the first step—which was performed in previous works [7], [13]—might be avoided by expressing directly the friction cone under span form V_f instead of face form B, as the columns of V_f are simply the generators of the linearized friction cones. Once we have computed H, which is computationally expensive, testing equilibrium boils down to checking a set of linear inequalities, which has negligible computational cost:

$$HD_{xy} c^{xy} + Hd \leq 0 \tag{6}$$



(a) The CoM c is inside $\mathcal{Y}_{out}$, but outside $\mathcal{Y}_{in}$: we can not determine whether $c \in \mathcal{Y}$.

(b) After one iteration c is outside $\mathcal{Y}_{out}$, so we can say that $c \notin \mathcal{Y}$.

Fig. 1: One iteration of the IP algorithm.

F. Incremental Projection (IP)

The key idea of the Incremental Projection (IP) algorithm [10] is to maintain inner and outer polyhedral approximations ($\mathcal{Y}_{in}$ and $\mathcal{Y}_{out}$) of the support section $\mathcal{Y}$ defined by (6). $\mathcal{Y}_{out}$ is stored as a list of inequalities, while $\mathcal{Y}_{in}$ as a list of vertices. To test equilibrium for a CoM position $\bar{c}$ the algorithm iteratively refines both approximations, until either $\bar{c} \in \mathcal{Y}_{in}$ (positive answer) or $\bar{c} \notin \mathcal{Y}_{out}$ (negative answer). At each iteration the algorithm shrinks $\mathcal{Y}_{out}$ and grows $\mathcal{Y}_{in}$ by solving an LP of this form:

$$\begin{aligned} \underset{c\in\mathbb{R}^2, f\in\mathbb{R}^{3k}}{\text{maximize}} \quad & a^\top c \\ \text{subject to} \quad & Bf \leq 0 \\ & Af = D_{xy}c + d, \end{aligned} \tag{7}$$

where $a \in \mathbb{R}^2$ is the normal to a line separating $\bar{c}$ and $\mathcal{Y}_{in}$. The solution (c^*, f^*) of (7) bounds $\mathcal{Y}$ in the direction a, so we can shrink $\mathcal{Y}_{out}$ by adding the following inequalities:

$$a^\top c \leq a^\top c^*$$

Moreover, given that c^* is inside the support polygon (actually it is on its borders), we can grow $\mathcal{Y}_{in}$ by adding c^* to its list of vertices. Fig. 1 shows an example of one iteration of the algorithm. A detailed explanation with a proof of the speed of convergence can be found in the original paper [10].

G. Conclusions

In conclusion, we have recalled three algorithms to test equilibrium of a CoM position given a set of contacts. LP does not require any preparatory computation, but each new test of a configuration has a significant cost. PP requires a costly preparatory computation, but each new test is nearly cost free. IP is a trade off, as it requires some few iterations for the first tests, but its cost decreases after each new test.

In the following sections we propose two original algorithms to test equilibrium, both more efficient than LP, PP and IP in most situations. The first algorithm relies on the same hypothesis, but it is up to 3 times faster; the second one uses more restrictive hypothesis, but its cost is nearly negligible with respect to the others. We then propose a generic solution to extend all these algorithms to test for robust equilibrium.

III. Incremental Projection Revisited (IPR)

We propose a new IP algorithm that at each step solves an LP with less variables than (7). Being the main computational effort in IP the resolution of the LPs (7) in $3k+2$ dimensions, our new algorithm results to be much faster than the original one, especially for large values of k. We replace the original LP with another LP of *constant* dimension 4 that gives us the same solution $a^\top c^*$, so that we can build the same $\mathcal{Y}_{out}$. The 4d LP does not give us c^*, so we can not build $\mathcal{Y}_{in}$ as in the original algorithm. However, we show that a different $\mathcal{Y}_{in}$ can be directly deduced from $\mathcal{Y}_{out}$, at almost no cost.

A. Computing the Outer Approximation

We start by trying to bound the GIW cone in an arbitrary direction $h \in \mathbb{R}^6$ using the following LP:

$$\begin{aligned} \underset{w\in\mathbb{R}^6, \beta\in\mathbb{R}^{mk}}{\text{maximize}} \quad & h^\top w \\ \text{subject to} \quad & \beta \geq 0 \\ & w = V_w \beta, \end{aligned} \tag{8}$$

where m is the number of generators per contact, and the columns of $V_w = AV_f$ are the GIW generators (using the same notation of Section II-E). By eliminating the equality constraint $w = V_w\beta$, the LP (8) becomes:

$$\underset{\beta\geq 0}{\text{maximize}} \quad h^\top V_w \beta$$

It is easy to see that the solution of this LP is zero if $V_w^\top h \leq 0$ and infinity otherwise. If the solution is zero we can then bound the GIW cone in direction h:

$$h^\top w \leq 0,$$

which in turns allows us to bound $\mathcal{Y}$ in direction $D_{xy}^\top h$:

$$h^\top D_{xy} c^{xy} + h^\top d \leq 0$$

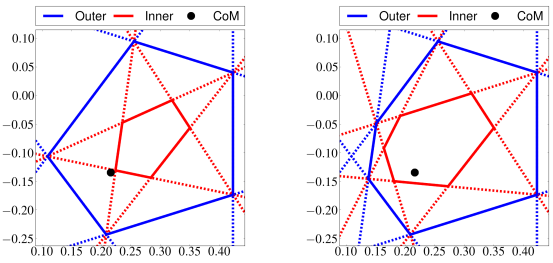
This observation allows us to avoid the explicit computation of the value of β in (8) (or similarly of f in (7)). To get a tight bound on $\mathcal{Y}$ in direction $a \in \mathbb{R}^2$ we now must simply solve this 6d LP (which does not contain the contact forces):

$$\begin{aligned} \underset{h\in\mathbb{R}^6}{\text{maximize}} \quad & d^\top h \\ \text{subject to} \quad & V_w^\top h \leq 0 \\ & D_{xy}^\top h = a \end{aligned} \tag{9}$$

Finally, given the simple structure of the matrix D_{xy}, it is straightforward to eliminate the equality constraints from (9) to convert it into a 4d LP. Solving this 4d LP gives us the same $\mathcal{Y}_{out}$ of the original IP algorithm, that is a collection of inequalities of the form:

$$a^\top c \leq -d^\top h^*$$

Another way to derive (9) is to replace f with $V_f\beta$ in (7) and then formulate the dual problem [16]. When (9) is unbounded, (7) is unfeasible, meaning that the support section is empty. When (9) is unfeasible, (7) is unbounded, that is the support section is unbounded in direction a.



(a) The CoM c is inside $\mathcal{Y}_{out}$, but outside $\mathcal{Y}_{in}$: we can not determine whether $c \in \mathcal{Y}$.

(b) After one iteration c is inside $\mathcal{Y}_{in}$, so we can say that $c \in \mathcal{Y}$.

Fig. 2: One iteration of the IPR algorithm.

B. Computing the Inner Approximation

The dual formulation (9) does not give us the optimal primal variable c^*, so we can not build the same inner approximation $\mathcal{Y}_{in}$ of IP. A cheap way to build an inner approximation starting from a tight outer approximation is:

- to maintain the ordered list of vertices of $\mathcal{Y}_{out}$: v^{out}
- for each vertex i compute the inequality passing through v_i^{out} and v_{i+2}^{out} (with sign such that $v_{i+1}^{out} \notin \mathcal{Y}_{in}$)

This simple procedure gives us a halfspace representation of $\mathcal{Y}_{in}$. Fig. 2 shows one iteration of the algorithm, while a complete example is illustrated in the accompanying video. Note that for $\mathcal{Y}_{in}$ not to be empty we need $\mathcal{Y}_{out}$ to have at least five vertices—some of which may be coincident. Indeed every time $\mathcal{Y}_{out}$ has three coincident vertices, the same vertex belongs to $\mathcal{Y}_{in}$ as well, which is how $\mathcal{Y}_{in}$ can converge to $\mathcal{Y}_{out}$.

Our inner approximation is rougher than the one of the original algorithm [10]. This is the payback for having a much cheaper computation of $\mathcal{Y}_{out}$. Consequently, we can not hope to have the same convergence speed. However, in practice our algorithm turns out to be much faster.

IV. Quasi-Flat Conditions for Static Equilibrium (QF)

A. Classical Case

It is well-known that when the robot is in contact with a horizontal ground a necessary and sufficient condition for equilibrium is [17]:

$$c^{xy} \in \text{Hull}(\{p_i^{xy}\}), \tag{10}$$

where $\text{Hull}(\{p_i^{xy}\})$ is the convex hull of the contact points on the ground, that is a convex 2d polygon. This condition still holds for flat sloped grounds (as long as friction is large enough to avoid slippage). We show now that this condition remains interesting for a large class of contacts, even if only as an approximation.

B. Quasi-Flat Contacts

Consider an arbitrary set of contact points p_i and their associated friction cones denoted by $\mathcal{K}_i$. We say that this contact configuration is *quasi-flat* if the direction opposite to gravity is in every cone $\mathcal{K}_i$. In that case, the condition (10) is sufficient for equilibrium.

Proof: We rewrite (1) for $\ddot{c} = \dot{L} = 0$:

$$\sum_{i=1}^{k} f_i = -mg, \tag{11a}$$

$$\sum_{i=1}^{k} (p_i^{xy} f_i^z - p_i^z f_i^{xy}) = c^{xy} \sum_{i=1}^{k} f_i^z \tag{11b}$$

$$\sum_{i=1}^{k} (p_i^x f_i^y - p_i^y f_i^x) = 0 \tag{11c}$$

Setting $f_i^x = f_i^y = 0 \quad \forall i$ in (11) leaves us with:

$$\sum_{i=1}^{k} f_i^z = m||g||, \qquad \frac{\sum_{i=1}^{k} p_i^{xy} f_i^z}{\sum_{i=1}^{k} f_i^z} = c^{xy}$$

Since by assumption $f_i^z \geq 0$ is a valid force for all contacts, these equations always admit a solution if (10) is satisfied. ■

The sufficiency of (10) implies that any CoM whose projection is inside the convex hull allows for equilibrium. However, other CoM positions that allow for equilibrium may exist. Looking at (11b) we can easily see that the larger the height difference of the contact points, the larger the error we commit using $\text{Hull}(\{p_i^{xy}\})$ as an approximation of $\mathcal{Y}$. We will show that this condition—which we call the *quasi-flat* test (QF)—is accurate in a large majority of interesting situations.

This sufficient condition had already been observed for the case of three contact points [17]. Here, we generalized it to an arbitrary number of contacts and our tests quantify the computational gains of using QF rather than the general PP technique.

V. Robust Static Equilibrium

A. Existing Criteria

Real systems are affected by countless uncertainties and disturbances, hence conditions (6) may not be enough to ensure equilibrium. Different robustness measures have already been proposed. Caron et al. [7] proposed to check the capacity of the system to generate x-y CoM accelerations inside a polytope (while maintaining $\dot{L} = 0$). This measure would typically corresponds to situations in which the orientation of the robot is not properly estimated. The robustness measure proposed by Barthelemy [12], [13] is the radius of the largest hypersphere centered at w_0 that is fully contained inside the GIW cone. This measure is more abstract and it is hard to relate it to any particular uncertainty.

B. Robustness to Contact-Force Errors

In our previous work [18] we studied the control of legged robots focusing on the robustness to torque-tracking errors, a major source of uncertainty that in turns affect the contact-force tracking. We thus propose to account for robustness to errors in the contact-force tracking: we would like the forces necessary to maintain equilibrium not to be too close to the friction cone boundaries. Our motivation is that we do

not want small errors in the contact-force tracking to cause slippage or rotation at the contacts. This uncertainty model also indirectly covers other errors such as in the orientation of the contacting bodies or friction coefficients.

Denoting the i-th contact-force tracking error $e_i \in \mathbb{R}^3$, we consider a worst-case approach by ensuring robustness to any force error within given bounds e^{max}:

$$e_i \in \mathcal{B}_i(e^{max}) = \{e_i \in \mathbb{R}^3 : |R_i e_i| \leq e^{max} \mathbf{1}_3\},$$

where $R_i = \begin{bmatrix} t_{i1} & t_{i2} & n_i \end{bmatrix}^\top$. A force vector f is then considered robust if:

$$B(f + e) \leq 0 \qquad \forall\, e = (e_1 \ \dots \ e_k) \in \mathcal{B}_1 \times \cdots \times \mathcal{B}_k$$

Given the simple structure of B we can show that this is equivalent to a vertical shift of the origin of the friction cones:

$$B(f - \underbrace{(l_1 \ \dots \ l_k)}_{l}) \leq 0,$$

where $l_i = e^{max}(\sqrt{2} + \mu_i)\mu_i^{-1} n_i$.

C. Testing Robust Static Equilibrium

We can easily adapt the three methods for testing equilibrium presented in the previous sections to account for robustness. In the LP test (5) and in the LP (7) used by IP we just need to replace the constraints $Bf \leq 0$ with the new constraints $Bf \leq Bl$. In the PP approach the shift of the origin of the force friction cones causes a shift of the origin of the GIW cone:

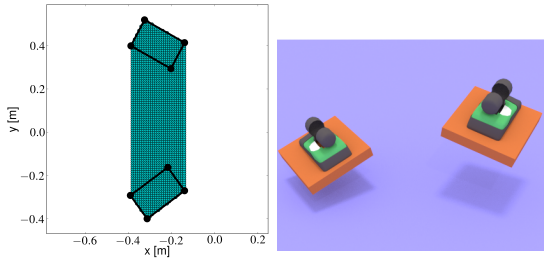
$$H(w - Al) \leq 0 \quad \rightarrow \quad HD_{xy}c^{xy} + H(d - Al) \leq 0$$

Finally, in the LP (9) solved by the IPR algorithm we just have to replace d with $(d - Al)$ in the cost function. These simple modifications allow us to test for robust equilibrium at no additional computational cost.

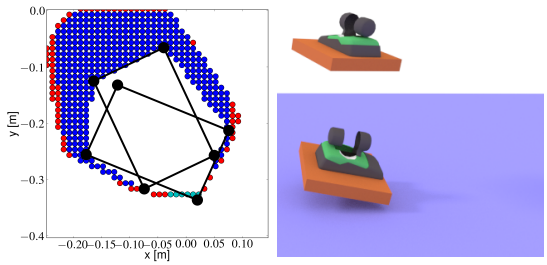
VI. TESTS

We present three tests to validate the three different contributions of this paper. Test 1 focuses on the quasi-flat contacts (Section IV), showing the quality of the QF approximation while quantifying the evident computational gains with respect to PP. It also shows that, on average, QF results in less false negatives than PP with 4-side linearized friction cones. Test 2 compares the performances of four methods for testing equilibrium: LP, PP, IP and IPR. In almost all scenarios our algorithm outperforms the others in terms of computation time. Test 3 shows how using a robust equilibrium criteria (Section V) results in planning qualitatively better contact postures [11].

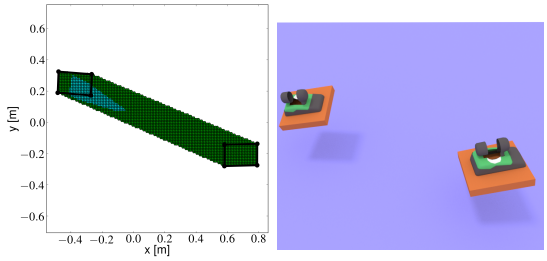
In all our tests, we have used a friction coefficient $\mu = 0.3$ for all contacts. We have solved the Linear Programs with *qpOases* [19], an efficient C++ active-set solver for Quadratic Programs, which can also solve Linear Programs. For the polytope projection we have used the C++ *cdd* library [15]. We wrote all our code in *Python*, using bindings of the aforementioned libraries.



(a) False negatives QF=0%, PP4=100% (cyan), PP8=0%



(b) False negatives QF=49% (blue+red), PP4=8%(red), PP8=0%



(c) False negatives QF=0%, PP4=100% (green+cyan), PP8=88% (green)

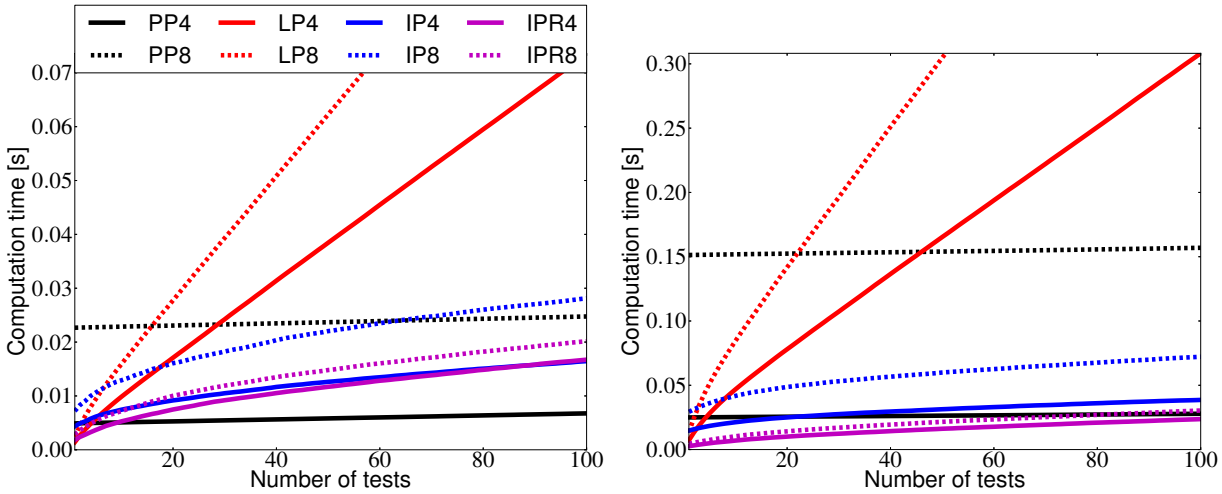
Fig. 3: Three examples of biped contact configurations in which at least one of the three considered methods presents a large number of false negatives. Only the false negatives are depicted in the figures.

TABLE I: Results of Test 1.

Method	QF	PP4	PP8
Mean false negatives	1.2 %	13.7%	1.0%
Max false negatives	49.2%	100%	100%
Mean computation time [ms]	0.14	4.86	22.57

A. Test 1 - Quasi-Flat Contacts

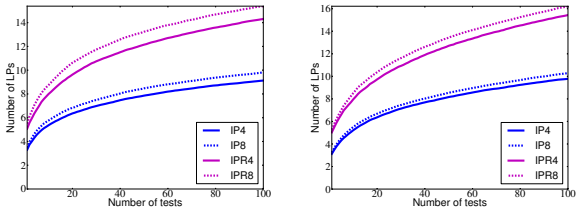
In this test we compare three methods to compute a conservative approximation of $\mathcal{Y}$ in case of quasi-flat contacts (i.e. all the friction cones contain the negative gravity direction). The first method (QF) is to use the 2d convex hull of the vertical projection of the contact points (see Section IV). The second method (PP4) is to approximate the friction cones with 4-side pyramids and to use the polytope-projection technique discussed in Section II-E. The third method (PP8) is the same as the second one, but using 8-side pyramids, so as to be more precise. Each method results in a different polygon, each polygon being included in the real support section $\mathcal{Y}$. For each contact configuration we evaluate the support polygon computed by the three methods on a 2d grid of CoM positions with step 1 cm. To avoid computing the real support section we approximate it by the union of



(a) Two contacts (i.e. 8 contact points).

(b) Four contacts (i.e. 16 contact points).

Fig. 4: Mean computation times to test the equilibrium of 1-100 CoM positions using LP, PP, IP and IPR with 4 or 8 generator per contact point. Using 4 generators rather than 8 we had 19% / 14% of false negatives with two / four contacts.



(a) Two contacts (i.e. 8 contact points).

(b) Four contacts (i.e. 16 contact points).

Fig. 5: Number of LPs solved to test the equilibrium of 1-100 CoM positions using IP and IPR with 4 or 8 generators per contact point.

the three computed polygons. We then count the percentage of false negatives with respect to true positives for each test. We compare the three methods on 10^3 random biped contact configurations (each foot contact being defined by four contact points). We sampled the feet positions in a box of size $1 \times 1 \times 0.5$m (x, y, z) and the feet orientations such that all friction cones contained the negative gravity direction. Table I reports the average/max percentage of false negatives and the mean computation time for each method. As evidenced by Fig. 3, for each method there exist contact configurations resulting in a large percentage of false negatives. This percentage can go as high as to 100% for PP4 and PP8 (i.e. the approximation says that no CoM position allows for equilibrium). This happens when the linearized cones do not contain the negative gravity direction, while the real cones do. On average, PP4 performs much worse than the others, while PP8 performs a bit better than QF. However, the better performances of PP8 comes at a high price in terms of computation time, which is more than 100 times higher with respect to QF.

B. Test 2 - Incremental Projection Revisited

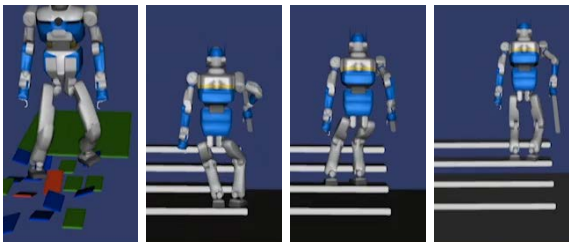
In this test we compare the performances of the algorithms LP, PP, IP and IPR. We compared the algorithms for 2 and 4 rectangular contacts with the environment. We also tested the algorithms using either 4 or 8 generators per contact point to measure the percentage of false negatives when using 4 generators. For each random contact configuration we used the four algorithms to test 100 random CoM positions (sampled in a box slightly bigger than the convex hull of the contact points) and we measured the computation times.

Fig. 4 shows the mean computation times for 10^3 random contact configurations. As expected LP is more efficient than PP for a small number of tests—which varies between 5 and 20 depending on the number of contacts and generators. PP is extremely fast for the case of two contacts and 4 generators, but its cost quickly increases with the number of contacts and generators, making it appealing only for very large numbers of tests (> 100). IP performs better than LP and PP in many situations, especially for an average number of tests (5-20) or for a large number of contacts and generators. IPR is overall the best algorithm because it is always the fastest (except for the case of 2 contacts and 4 generators, which is not very interesting given the large number of false negatives that we measured when using 4 generators). IPR exhibits the best performances in the case of 4 contacts and 8 generators, where it is about 3 times faster than IP.

Fig. 5 shows that the ratio between the mean number of LPs solved by IP and IPR as a function of the number of tests is almost constant (between 1.5 and 1.6). This holds true regardless of the number of contacts and generators. IPR solves then 50-60 % more LPs than IP, but being the LPs of IPR of dimension 4 rather than $3k + 2$ it is not surprising that for $k \geq 8$ IPR performs better than IP.

C. Test 3 - Planning with Robust Static Equilibrium

This last test compares the postures generated by our contact planner using the standard equilibrium test and the robust one. We only provide a qualitative evaluation, as it seems difficult to quantitatively validate the robustness of a



(a) Robust. (b) Non-robust. (c) Non-robust. (d) Robust.

Fig. 6: Examples of robust and non-robust contact postures found by a contact planner.

plan without a statistical number of trials on a real robot.

The contact planner [11] first finds a collision-free trajectory of the base of the robot using a probabilistic planner. It then iteratively computes the contacts that maintain the robot in equilibrium at each key pose. The contacts are obtained by random shooting, rejecting those that do not allow for equilibrium. In the original paper [11] the PP test was used; we replaced it here with a robust PP test and compared the results.

We present two different scenarios. In the first scenario the humanoid robot HRP-2 has to walk over rubble (see Fig. 6a). The ground is composed by three kinds of blocks (green, blue, red) having three different inclinations $(0°, 15°, 20°)$. The robust planner naturally avoids stepping on the red blocks, because this would not result in a robust equilibrium. In the second scenario HRP-2 has to climb some stairs using a handrail for support. The non-robust planner tends to find postures where the CoM of the robot is close to the borders of its support region, which are clearly non-robust (see for instance Fig. 6b and 6c). The robust planner finds instead qualitatively better poses, such as the one depicted in Fig. 6d. The whole sequence of contact poses can be seen in the accompanying video.

VII. CONCLUSIONS

This paper addressed one of the central problem in control and planning of legged robots: testing static equilibrium. The presented algorithms advance the state of the art in two directions. First, from a computational standpoint, we presented two algorithms to test equilibrium that are faster than previous approaches. The first algorithm is actually a well-known technique, which we proved to be applicable to a much larger set of contact geometries (i.e. *quasi-flat* contacts). The second algorithm is a modification of the incremental projection technique [10], which resulted to be up to 3 times faster than the original version. Second, we proposed a method to test robust equilibrium that allows the robot not to lose contact in case of bounded force-tracking errors. We incorporated this feature in all the discussed techniques for testing equilibrium; we also showed the qualitative difference between the robust and non-robust posture sequences generated by a contact planner [11].

An interesting future direction is the extension of our IPR algorithm to handle quadratic cones, similarly to [9]. Another useful extension would be dynamic equilibrium. While on flat grounds the capture point [20] is a good indicator of dynamic equilibrium, no such thing exists for arbitrary contacts, but it is clear that the mathematical tools to deal with it are similar to the one discussed in this paper.

REFERENCES

[1] P.-B. Wieber, R. Tedrake, and S. Kuindersma, "Modeling and Control of Legged Robots," in *Bruno Siciliano and Oussama Khatib, editors, Springer Handbook of Robotics, 2nd Ed, chapter 48*, 2015.

[2] J. R. Rebula, P. D. Neuhaus, B. V. Bonnlander, M. J. Johnson, and J. E. Pratt, "A controller for the LittleDog quadruped walking on rough terrain," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1467–1473, 2007.

[3] E. Yoshida, O. Kanoun, C. Esteves, and J. P. Laumond, "Task-driven support polygon reshaping for humanoids," *Proceedings of the 2006 6th IEEE-RAS International Conference on Humanoid Robots, HUMANOIDS*, pp. 208–213, 2006.

[4] T. Bretl, "Motion Planning of Multi-Limbed Robots Subject to Equilibrium Constraints: The Free-Climbing Robot Problem," *The International Journal of Robotics Research*, vol. 25, no. 4, pp. 317–342, 2006.

[5] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion Planning for Legged Robots on Varied Terrain," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1325–1349, 2008.

[6] Q.-C. Pham and O. Stasse, "Time-Optimal Path Parameterization for Redundantly-Actuated Robots (Numerical Integration Approach)," *IEEE/ASME Transactions on Mechatronics*, 2015.

[7] 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 Systems (RSS)*, 2015.

[8] P.-B. Wieber, "On the stability of walking systems," in *Proceedings of the International Workshop on Humanoid and Human Friendly Robotics*, 2002, pp. 1–7.

[9] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 794–807, 2008.

[10] ——, "A fast and adaptive test of static equilibrium for legged robots," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2006, no. May, pp. 1109–1116, 2006.

[11] S. Tonneau, N. Mansard, C. Park, D. Manocha, F. Multon, and J. Pettr, "A Reachability-based planner for sequences of acyclic contacts in cluttered environments," in *ISRR*, 2015.

[12] S. Barthélemy and P. Bidaud, "Stability measure of postural dynamic equilibrium based on residual radius," *Advances in Robot Kinematics: Analysis and Design*, pp. 399–407, 2006.

[13] 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, pp. 1–8.

[14] ——, "A hierarchical framework for realizing dynamically-stable motions of humanoid robot in obstacle-cluttered environments," in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 867–874.

[15] K. Fukuda and A. Prodon, "Double Description Method Revisited," in *Combinatorics and Computer Science (LNCS 1120)*, 1996, vol. 1, pp. 91–111.

[16] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004, vol. 98.

[17] Y. Or and E. Rimon, "Analytic Characterization of a Class of Three-contact Frictional Equilibrium Postures in Three-dimensional Gravitational Environments," *The International Journal of Robotics Research*, vol. 29, no. 1, pp. 3–22, 2010.

[18] A. Del Prete and N. Mansard, "Addressing Constraint Robustness to Torque Errors in Task-Space Inverse Dynamics," in *Robotics, Science and Systems (RSS)*, Rome, 2015.

[19] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[20] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture Point: A Step toward Humanoid Push Recovery," *2006 6th IEEE-RAS International Conference on Humanoid Robots*, 2006.