C-CROC: Continuous and Convex Resolution of Centroidal dynamic trajectories for legged robots in multi-contact scenarios

Pierre Fernbach, Steve Tonneau, Olivier Stasse, Justin Carpentier, Michel Taïx

C-CROC: Continuous and Convex Resolution of Centroidal dynamic trajectories for legged robots in multi-contact scenarios

Pierre Fernbach, Steve Tonneau, Olivier Stasse, *Senior Member, IEEE,* Justin Carpentier, *Member, IEEE,* and Michel Taïx

***Abstract*—Synthesizing legged locomotion requires planning one or several steps ahead (literally): when and where, and with which effector should the next contact(s) be created between the robot and the environment? Validating a contact candidate implies *a minima* the resolution of a slow, non-linear optimization problem, to demonstrate that a Center Of Mass (CoM) trajectory, compatible with the contact transition constraints, exists.**

We propose a conservative reformulation of this trajectory generation problem as a convex 3D linear program, CROC (Convex Resolution Of Centroidal dynamic trajectories). It results from the observation that if the CoM trajectory is a polynomial with only one free variable coefficient, the non-linearity of the problem disappears. This has two consequences. On the positive side, in terms of computation times CROC outperforms the state of the art by at least one order of magnitude, and allows to consider interactive applications (with a planning time roughly equal to the motion time). On the negative side, in our experiments our approach finds a majority of the feasible trajectories found by a non-linear solver, but not all of them. Still, we demonstrate that the solution space covered by CROC is large enough to achieve the automated planning of a large variety of locomotion tasks for different robots, demonstrated in simulation and on the real HRP-2 robot, several of which were rarely seen before.

Another significant contribution is the introduction of a Bezier curve representation of the problem, which guarantees that the constraints of the CoM trajectory are verified continuously, and not only at discrete points as traditionally done. This formulation is lossless, and results in more robust trajectories. It is not restricted to CROC, but could rather be integrated with any method from the state of the art.

***Index Terms*—Multi contact locomotion, centroidal dynamics, Humanoid robots, legged robots, motion planning**

I. Introduction

THIS paper considers the issue of planning multi-contact motions for legged robots in human environments.

The term "multi-contact motion" has been proposed to distinguish the problem from the gaited locomotion one [**?**], [**?**]. Gaited motions result from the contact interactions created and broken periodically between the end effectors and a flat terrain. The multi-contact problem is more general as it can include non horizontal contacts, and is not restricted to a cyclic strategy. This results in a combinatorial problem in the choice of the contacts being created. It also requires a more complex formulation of the dynamics that govern the motion. This non-linear problem remains open to this date.

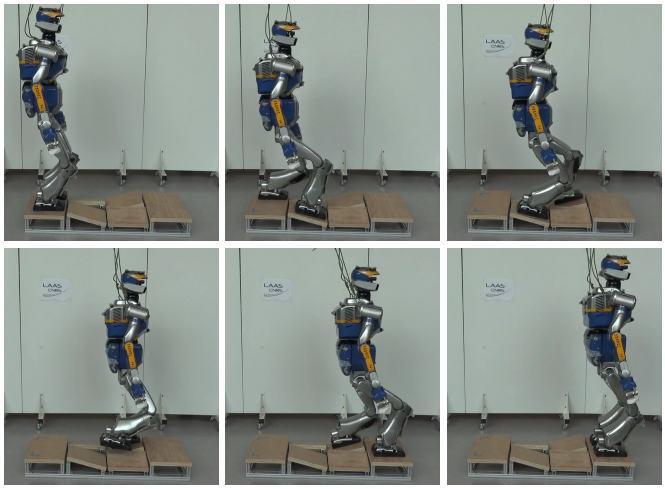

Fig. 1: An instance of the transition feasibility problem: can we guarantee that the contact sequence shown in this picture can be used to produce a feasible motion for the robot? To address this issue in this example we need to account for 9 different contact phases (including phases where the effector is flying, as displayed in the fourth image).

One key issue of multi-contact locomotion consists in choosing contact locations such that the contacts can be broken or created at a given time without violating dynamic or geometric constraints. To tackle this issue one option is to embrace the non-linearity of the problem and simultaneously optimize the contact locations and the motion of the robot [**?**], [**?**], [**?**], [**?**]. While the validity of the generated motions remains to be demonstrated on real robots, the motions generated in simulation are among the most impressive achieved.

Alternatively the problem can be decomposed into a sequence of smaller ones [**?**], [**?**], [**?**], [**?**], [**?**]. In this case, the computation of a contact plan (the discrete list of contact positions along the motion) is achieved prior to the motion generation. This simplifies the problem but introduces the question of the validity (feasibility) of the contact plan.

To further simplify the problem, both families of approaches propose contributions that rely on a model-based approach called the *centroidal model*, which only considers the dynamics of total linear and angular momenta of the system

Steve Tonneau is with the University of Edinburgh, Scotland. e-mail: stonneau@exseed.ed.ac.uk

Justin Carpentier is with INRIA, ENS, CNRS, PSL Research University, Paris, France. e-mail: justin.carpentier@inria.fr

The other authors are with LAAS-CNRS / Université de Toulouse, France. e-mail: {firstname}.{surname}@laas.fr

expressed around the Center of Mass of the robot, rather than considering the entire whole-body dynamics. While being of smaller dimension, this model introduces approximations regarding the geometric constraints that lie on the robot, and also regarding the angular momentum variation induced by the motions of the rigid bodies that compose the robot. The centroidal model is widely adopted because it allows for a reduction in the dimension of the problem. Unfortunately the centroidal dynamics is also non-linear. Computing trajectories that satisfy the centroidal dynamics thus remains challenging and time consuming.

The key idea of our work is to improve the resolution of problems based on the centroidal model, by introducing a conservative but convex reformulation of the centroidal dynamics equations. For this purpose, we only consider a subset of all the feasible center of mass trajectories, but the trade-off is that (i) we are able to compute a trajectory in this subset one order of magnitude faster than any state-of-the-art approaches, (ii) we will always find it, while (iii) not requiring any initial guess.

To derive this formulation, rather than considering the centroidal dynamics in the general context of trajectory optimization, we focus on what we call the **transition feasibility** problem: given two states of the robot, can we guarantee that there exists (or not) a dynamically and kinematically consistent motion that connects these two states (see Figure 1)?

This problem is first relevant for the multi-contact planning problem, which is our main target application. Our strategy is to handle the combinatorial problem by determining as fast as possible whether a set of potential contact candidates allows for a feasible motion. If not, we can then proceed to evaluate the next candidate until we find a relevant contact. The transition feasibility problem also addresses the N-step capturability problem [**?**], [**?**], [**?**]: given the current state of the robot, determine whether it will be able to come to a stop without falling in at most N steps ($N \geq 0$). This issue is very important to guarantee the safety of the robot and its surroundings.

Recent contributions have proposed centroidal trajectory generation methods that could theoretically be used to answer the transition feasibility problem [**?**], [**?**], [**?**]. However, because of the combinatorial aspect of contact planning, the computational time required by these methods is too large to exploit them in a trial-and-error approach to verify the feasibility. Caron et al. recently proposed a computationally efficient method [**?**], but its application range is restricted to single-contact to single-contact transitions.

The works that are the closest to the present paper propose a convex relaxation of the CoM constraints [**?**], [**?**]. The method proposed in this paper is conservative rather than approximate while being more computationally efficient.

A. Contributions

The main contribution is the formulation of a **transition feasibility** criterion called *CROC* (Convex Resolution Of Centroidal dynamic trajectories). Thanks to a conservative and convex reformulation of the problem, the criterion is computed in a fraction of the computational cost required by standard non-linear solvers of state of the art. *CROC* guarantees that the linear dynamics of the Center of Mass (CoM) is always fufilled. It also improves the state of the art by proposing relevant kinematic constraints on the CoM, although they are only approximate.

Considering two states separated by at most one contact creation and one contact break, *CROC* is able to test if there exists a kinematically and dynamically valid motion that connects these two states.

CROC also outputs a CoM trajectory that can be used as a valuable initial guess by a non-conservative non-linear solver to converge towards an optimal solution. Noticeably, this formulation is, along with [**?**], one of the few **able to continuously guarantee that the computed trajectories respect the centroidal dynamics constraints of the problem**, when other approaches require to discretize the trajectory and check punctually the constraints [**?**], [**?**], [**?**].

We demonstrate the interest of *CROC* in the context of the multi-contact planning problem, both in simulation and on the real HRP-2 robot. *CROC* is integrated within an automated contact-planning framework. Our experiments empirically demonstrate that using *CROC* within a multi-contact planning framework [**?**] results in a drastic improvement of the success rates of the planner, effectively increasing it by more than 70% points in the most challenging scenarios.

This paper is organized as follows. In section II, we recall the formal definition of the problem. The main contribution of the paper is presented in section III. We then introduce our contact-planning framework and present our experimental results in section V.

B. Situation of the contribution with respect to the authors previous work

The present paper is an extension of an IROS conference paper [**?**], where we have introduced a convex optimization method to solve the transition feasibility problem. Our previous formulation, as others in the community, is limited by the necessity to use of the double description method [**?**], an unstable mean to compute the linear constraints that apply to the problem [**?**], which allows for fast computations. As for all existing methods, it also requires a discretization of the solution trajectory, such that the constraints of the problem are only checked at specific instants. This behavior is unsafe as the trajectory between each discretization point is unchecked and may not respect the constraints.

In this paper, we propose a new formulation of the problem that removes the need for discretization of the CoM trajectory, thus guaranteeing that the constraints are respected continuously along the whole trajectory. Contrary to our previous work, this formulation applies even when contacts are created or broken along the trajectory. We advocate for the adoption of this formulation for any centroidal generation method, convex or not.

C. Outline of the paper

Sections II and III present important similarities with respect to [**?**]. The novelty appears from section III-D, where we

present a continuous formulation able to deal with contact switching during the trajectory.

The other sections of the paper are also novel. These novelties include the completion of our experimental framework, which enables us to validate our method on several experiments on the real robot. We also provide an empirical analysis of the performances of our method with respect to a state-of-the-art nonlinear solver, in terms of success rate and computation times.

II. Problem definition

We define the transition feasibility problem as follows. Given two configurations of a robot; given the contact locations associated to these two configurations; given the position, velocity and acceleration of the Center Of Mass (CoM) of the robot at these two configurations; can we guarantee that there exists a **feasible** motion that connects the two configurations? A feasible motion should respect the kinematic constraints of the robot, as well as the dynamics expressed at its CoM. Depending on the use case, some constraints may be removed (for instance if the end configuration is unknown, or if the problem is simply to put the robot to a stop).

Thus, in this work we define the transition feasibility problem with respect to the centroidal dynamics of a robot, as now commonly done in the legged robotics community [**?**], [**?**], [**?**]. In this section we provide some formal definitions that are used in the rest of the paper.

A. Contact sequence and state

A legged motion can be discretized into a sequence of contact phases. Each contact phase defines a number of active contacts, and their locations remain constant during the phase. Thus, each contact phase constrains kinematically and dynamically the motion of the robot. Within a contact sequence, each adjacent contact phase differs by exactly one contact creation or removal (for instance when walking, the contact sequence is gaited and alternates simple and double support phases). The considered contact surfaces are assumed to be rectangular (4 extreme points on each foot) for humanoid robots, and punctual for quadrupedal robots.

We define a state $\mathbf{x} = (\mathbf{c}, \dot{\mathbf{c}}, \ddot{\mathbf{c}}) \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3$ as the triplet describing the CoM position, velocity and acceleration. To indicate that a state is compatible with the dynamic and kinematic constraints associated with a contact phase $p \in \mathbb{N}$, we use the superscript notation $\mathbf{x}^{\{p\}} = (\mathbf{c}^{\{p\}}, \dot{\mathbf{c}}^{\{p\}}, \ddot{\mathbf{c}}^{\{p\}})$.

Given two states $\mathbf{x}_s^{\{p\}}$ and $\mathbf{x}_g^{\{q\}}$ with $q \geq p$, the transition feasibility problem consists in determining whether there exists a feasible trajectory $\mathbf{c}(t), t \in \mathbb{R}^+$ of duration $T \in \mathbb{R}^+$, which connects exactly $\mathbf{x}_s^{\{p\}}$ and $\mathbf{x}_g^{\{q\}}$.

B. Centroidal dynamic constraints on $\mathbf{c}(t)$

For a contact phase $\{p\}$ of duration T, for any $t \in [0, T]$ the centroidal dynamic constraints are given by the Newton-Euler equations. These constraints form a convex cone (or polytope), which can be expressed under two different formulations, theoretically equivalent [**?**], [**?**], [**?**], but really different in practice. In this paper we present and discuss both formulations.

1) Equality constraint representation (or force formulation): The Newton-Euler equations are:

$$\begin{bmatrix} m(\ddot{\mathbf{c}} - \mathbf{g}) \\ m\mathbf{c} \times (\ddot{\mathbf{c}} - \mathbf{g}) + \dot{\mathbf{L}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & \dots & \mathbf{I}_3 \\ \hat{\mathbf{p}}_1 & \dots & \hat{\mathbf{p}}_{nc} \end{bmatrix} \mathbf{f} \tag{1}$$

Where :

- m is the total mass of the robot;
- nc is the number of contact points;
- $\mathbf{p}_i \in \mathbb{R}^3, 1 \leq i \leq nc$ is the location of the i-th contact point; [1]
- $\mathbf{f} = \begin{bmatrix} \mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_{nc} \end{bmatrix}^T \in \mathbb{R}^{3nc}$ is the stacked vector of contact forces applied at each contact point;
- $\mathbf{g} = \begin{bmatrix} 0 & 0 & -9.81 \end{bmatrix}^T$ is the gravity vector;
- $\dot{\mathbf{L}} \in \mathbb{R}^3$ is the time derivative of the angular momentum (expressed at $\mathbf{c}$).
- $\hat{\mathbf{p}}_i$ denotes the skew-symmetric matrix of $\mathbf{p}_i$.

The contact forces are further constrained to lie in their so-called friction cone, which we conservatively linearize with four generating rays. Thus $\mathbf{f}$ has the form $\mathbf{f} = \mathbf{V}\boldsymbol{\beta}$, where $\mathbf{V} \in \mathbb{R}^{3nc \times 4nc}$ is the matrix containing the diagonally stacked generating rays of the friction cone of each contact point and $\boldsymbol{\beta} \in \mathbb{R}^{4nc+}$ is a variable.

This formulation has the disadvantage of introducing a large number of variables associated to the contact forces (one vector $\boldsymbol{\beta}$ for each instant where the constraints are verified).

2) Inequality constraint representation (or Double Description formulation): Because the set of admissible contact forces is a polytope, it is possible to use an equivalent "face representation" of the constraints that applies both to the center of mass and the angular momentum quantities. With this formulation, the force variables disappear:

$$\mathbf{H} \underbrace{\begin{bmatrix} m(\ddot{\mathbf{c}} - \mathbf{g}) \\ m\mathbf{c} \times (\ddot{\mathbf{c}} - \mathbf{g}) + \dot{\mathbf{L}} \end{bmatrix}}_{\mathbf{w}} \leq \mathbf{h} \tag{2}$$

where $\mathbf{H}$ and $\mathbf{h}$ are respectively a matrix and a vector defined by the position of the contact points, their normal and their friction coefficients. As this matrix and vector are uniquely defined for a contact phase, we note them with the superscript $\{p\}$ for a contact phase p.

With this formulation, the dimension of the problem is greatly reduced. However, the computation of the matrices $\mathbf{H}^{\{p\}}$ and $\mathbf{h}^{\{p\}}$ is a non-trivial operation called the double description method [**?**]. It is computationally expensive, and subject to occasional failures.

In the following theoretical sections, we will use the inequality formulation because we believe our contribution is more intuitive with this representation. In terms of implementation the equality formulation is more reliable but slower. However we show that under our formulation the computation times remain in the same order of magnitude with both formulations.

[1] As commonly done, in the case of rectangular contacts (like most robot's feet) we define a contact point at each vertex of the rectangle.

3) The dynamic constraints are not convex: Because of the cross product between $\mathbf{c}$ and $\ddot{\mathbf{c}}$ in the equations (1) and (2), the constraints are bi-linear, leading to a non-convex problem to solve.

C. Centroidal kinematic constraints on $\mathbf{c}(t)$

Each active contact also introduces a kinematic constraint on $\mathbf{c}(t)$, depending of the placement of the end-effectors of the robot. We use a linear constraint formulation to represent this constraint depending on the 6D positions of each active contact frames. They give us a necessary but not sufficient condition for the kinematic feasibility. We refer the reader to [**?**] for the computation of these constraints. For a given contact phase $\{p\}$ this constraint can be expressed as :

$$\mathbf{K}^{\{p\}}\mathbf{c} \leq \mathbf{k}^{\{p\}} \tag{3}$$

III. Convex formulation of the transition problem

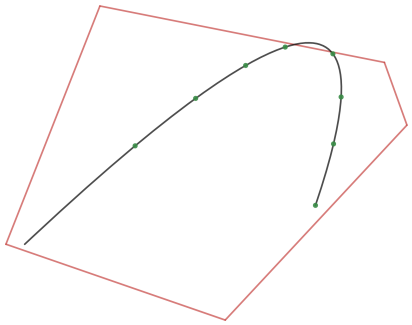

Fig. 2: Example of an invalid solution found by a discretized method. The red lines represent the constraints, wile the black curve is the solution and the green dots are the discretization points. All the discretization points satisfy the constraints while the curve clearly violates them.

As previously proposed [**?**], in order to determine the existence of a valid centroidal trajectory $\mathbf{c}(t)$, we formulate the problem as a convex one by getting rid of the non-linear constraints induced by the cross product operation $\mathbf{c} \times \ddot{\mathbf{c}}$. To achieve this we impose a conservative condition on $\mathbf{c}(t)$.

However, a significant contribution with respect to [**?**] and other contributions is a continuous reformulation of the problem, which guarantees that the resulting trajectory is always valid. Indeed, traditionally the constraints are only verified at specific points of the trajectory, using a discretization step that must be carefully calibrated to avoid an explosion in the number of variables and constraints, while guaranteeing that the constraints will not be violated in between. Figure 2 illustrates the violation of the constraints.

A. Reformulation of $\mathbf{c}(t)$ as a Bezier curve

Let us assume that $\mathbf{c}(t)$ is described by an arbitrary polynomial of degree n of unknown duration T. In such case, without loss of generality, $\mathbf{c}(t)$ is equivalently defined as a constrained Bezier curve of the same degree n:

$$\mathbf{c}(t) = \sum_{i=0}^{n} B_i^n(t/T)\mathbf{P}_i \tag{4}$$

where the B_i^n are the Bernstein polynomials and the $\mathbf{P}_i$ are the control points.

With this formulation we can easily constrain the initial or final positions, velocity or any other derivatives by setting the value of the control points. To exactly connect two states $\mathbf{x}_s = (\mathbf{c}_s, \dot{\mathbf{c}}_s, \ddot{\mathbf{c}}_s)$ and $\mathbf{x}_g = (\mathbf{c}_g, \dot{\mathbf{c}}_g, \ddot{\mathbf{c}}_g)$, we thus need at least 6 control points to ensure that the following constraints are verified:

- $\mathbf{P}_0 = \mathbf{c}_s$ and $\mathbf{P}_n = \mathbf{c}_g$ guarantee that the trajectory starts and ends at the desired locations;
- $\mathbf{P}_1 = \frac{\dot{\mathbf{c}}_s T}{n} + \mathbf{P}_0$ and $\mathbf{P}_{n-1} = \mathbf{P}_n - \frac{\dot{\mathbf{c}}_g T}{n}$ guarantee that the trajectory initial and final velocities are respected;
- $\mathbf{P}_2 = \frac{\ddot{\mathbf{c}}_s T^2}{n(n-1)} + 2\mathbf{P}_1 - \mathbf{P}_0$ and $\mathbf{P}_{n-2} = \frac{\ddot{\mathbf{c}}_g T^2}{n(n-1)} + 2\mathbf{P}_{n-1} - \mathbf{P}_n$ guarantee that the initial and final accelerations are respected.

Depending on the considered problem, some constraints on the boundary positions, velocities or accelerations can be removed, without changing the validity of our approach. For instance, if the objective is simply to put the robot to a stop, the end velocities and accelerations can be set to zero, while the end position is left unconstrained. We can also extend this to any degree and add constraints on initial or final jerk or higher derivatives and automatically compute the position of the control points with a symbolic calculus script such as the one that we provide at the url [2]. We only need to compute the equation of the control points once and for all so we do not need to compute them at runtime. In the following equations, we use a curve of degree 6 with the constraints on initial and final position, velocity and acceleration as described above, and the same reasoning applies to all cases.

B. Conservative reformulation of the transition problem

We now constrain $\mathbf{c}(t)$ to be a Bezier curve of degree $n = 6$. This is a conservative approximation of the transition problem as it does not cover the whole solution space.

As we already need 6 control points to ensure that we connect exactly the two states, this leaves a free control point $\mathbf{P}_3 = \mathbf{y}$:

$$\mathbf{c}(t, \mathbf{y}) = \sum_{i \in \{0,1,2,4,5,6\}} B_i^6(t/T)\mathbf{P}_i + B_3^6(t/T)\mathbf{y} \tag{5}$$

In this case, $\mathbf{y}$ and T are the only variables of the problem. For the time being, we fix T to a constant value. We derive twice to obtain $\ddot{\mathbf{c}}(t)$, and compute the cross product to get the expression of $\mathbf{w}(t)$:

$$\mathbf{w}(t) = \begin{bmatrix} m(\ddot{\mathbf{c}} - \mathbf{g}) \\ m\mathbf{c} \times (\ddot{\mathbf{c}} - \mathbf{g}) + \dot{\mathbf{L}} \end{bmatrix} \tag{6}$$

[2] http://stevetonneau.fr/files/publications/iros18/derivate.py

The non-convexity of the problem disappears, because the cross product of $\mathbf{y}$ by itself is $\mathbf{0}$, and all other terms are either constant or linear in $\mathbf{y}$. $\mathbf{w}(t, \mathbf{y})$ is thus a six-dimensional Bezier curve of degree $2n-3$ [**?**] (9 in this case) linearly dependent of $\mathbf{y}$:

$$\mathbf{w}(t, \mathbf{y}) = \sum_{i \in \{0..9\}} B_i^9(t/T)\mathbf{P}_{\mathbf{w}i}(\mathbf{y}) + \dot{\mathbf{L}}(t) \tag{7}$$

where $\mathbf{P}_{\mathbf{w}i}(\mathbf{y}) \in \mathbb{R}^6$ are the control points of $\mathbf{w}(t, \mathbf{y})$ expressed as :

$$\mathbf{P}_{\mathbf{w}i}(\mathbf{y}) = \mathbf{P}_{\mathbf{w}i}^y \mathbf{y} + \mathbf{P}_{\mathbf{w}i}^s \tag{8}$$

The $\mathbf{P}_{\mathbf{w}i}^y \in \mathbb{R}^{6\times 3}$ and $\mathbf{P}_{\mathbf{w}i}^s \in \mathbb{R}^6$ are constants that only depend on the control points $\mathbf{P}_i$ of $\mathbf{c}(t, \mathbf{y})$ and of T.

In what follows, for the sake of simplicity, we assume $\dot{\mathbf{L}}(t) = \mathbf{0}$. This is not a limitation: if we express $\dot{\mathbf{L}}(t)$ as a polynomial in the problem the following reasoning stands. One way to include $\dot{\mathbf{L}}(t)$ is to represent it as a Bezier curve with one or more free variables. However guaranteeing that we can generate a whole-body motion that tracks a variable $\dot{\mathbf{L}}(t)$ requires additional information on the whole-body motion, which we leave as future work [**?**], [**?**], [**?**].

The existence of a valid trajectory $\mathbf{c}(t)$ can thus be determined by solving a convex problem.

C. Application for a motion with no contact switch

We first consider the case where $p = q = 1$.

1) Discrete formulation: Using a discretization step Δt, we discretize $\mathbf{c}(t, \mathbf{y})$ and $\mathbf{w}(t, \mathbf{y})$ over T as follows:

$$\begin{aligned} \mathbf{c}(j\Delta t, \mathbf{y}) = &\quad \mathbf{c}_j^y \mathbf{y} + \mathbf{c}_j^s \\ \mathbf{w}(j\Delta t, \mathbf{y}) = &\quad \mathbf{w}_j^y \mathbf{y} + \mathbf{w}_j^s \end{aligned} \tag{9}$$

Where $\mathbf{c}_j^y$, $\mathbf{c}_j^s$, $\mathbf{w}_j^y$ and $\mathbf{w}_j^s$ are constants given by $\mathbf{P}_{\{0,1,2,4,5,6\}}$, the total duration T and the time step $j\Delta t$. j belongs to the phase set $J^{\{p\}} : \{j \in \mathbb{N} : 0 \leq j\Delta t \leq T^{\{p\}}\}$. Given these expressions, we can replace $\mathbf{w}(t)$ in (2) by its value at each discretization point $j\Delta t$:

$$\mathbf{H}^{\{p\}}\mathbf{w}_j^y \mathbf{y} \leq \mathbf{h}^{\{p\}} - \mathbf{H}^{\{p\}}\mathbf{w}_j^s \tag{10}$$

By proceeding similarly for the kinematic constraint (3), we can formulate the following linear feasibility problem (FP) in 3 dimensions:

$$\begin{aligned} &\textbf{find} \quad \mathbf{y} \\ &\textbf{s.t.} \quad \underbrace{\begin{bmatrix} \mathbf{K}^{\{p\}}\mathbf{c}_j^y \\ \mathbf{H}^{\{p\}}\mathbf{w}_j^y \end{bmatrix}}_{\mathbf{E}_j^{\{p\}}} \mathbf{y} \leq \underbrace{\begin{bmatrix} \mathbf{k}^{\{p\}} - \mathbf{K}^{\{p\}}\mathbf{c}_j^s \\ \mathbf{h}^{\{p\}} - \mathbf{H}^{\{p\}}\mathbf{w}_j^s \end{bmatrix}}_{\mathbf{e}_j^{\{p\}}} \quad \forall j \in J^{\{p\}} \end{aligned} \tag{11}$$

With this discrete formulation the number of constraints in the problem is proportional to the number of discretization points. Moreover, the constraints are verified only at the discretization points, which leaves a risk that a part of the solution trajectory between two discretization points does not satisfy the constraints of the problem (Figure 2). Choosing the number of discretization steps is thus a compromise between the computation time (which depends on the number of constraints) and the risk of finding a solution partially invalid. This is a well-known issue when relying on discretization methods.

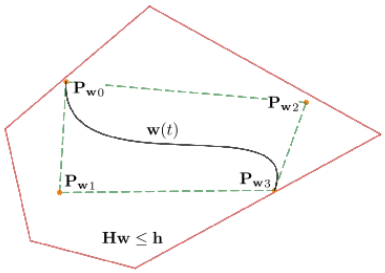

Fig. 3: A bezier curve is comprised in the convex hull of its control points. In this abstract view, the red polygon represents the 6D constraints on $\mathbf{w}(t)$. If the control points $\mathbf{P}_{\mathbf{w}i}$ of $\mathbf{w}(t)$ satisfy the constraints, then the complete curve satisfies the constraints.

2) Continuous formulation: Alternatively, in [**?**] we proposed a continuous formulation of this problem, only valid for the case where no contact transition occurs. We recall this formulation below as it is fundamental for the following section.

Using the fact that a Bezier curve is comprised in the convex hull of its control points, the main idea of this formulation is to express the kinematic constraints (3) on the control points $\mathbf{P}_i$ of $\mathbf{c}(t, \mathbf{y})$ and the dynamic constraints (2) on the control points $\mathbf{P}_{\mathbf{w}i}(\mathbf{y})$ of $\mathbf{w}(t, \mathbf{y})$ (see Figure 3). Constraining the control points of $\mathbf{w}(t, \mathbf{y})$ to satisfy the constraints of the trajectory is *a priori* a conservative approach that further constrains the solution space (we will see that this limitation can be easily overcome). However, this approach allows for a continuous solution to the problem and guarantees that the trajectory is entirely valid.

Assuming that the start and goal states are feasible (otherwise the problem has no solution), for the kinematic constraints we only need to find a $\mathbf{y}$ that satisfies the constraints. For the dynamic constraints all the control points $\mathbf{P}_{\mathbf{w}i}(\mathbf{y})$ must satisfy the equation (2), given the expression (8) we can express the dynamic constraints as follow:

$$\mathbf{H}^{\{p\}}\mathbf{P}_{\mathbf{w}i}^y \mathbf{y} \leq \mathbf{h}^{\{p\}} - \mathbf{H}^{\{p\}}\mathbf{P}_{\mathbf{w}i}^s, \forall i \in [0, 2n-3] \tag{12}$$

Finally, we can reformulate the discretized Linear Feasibility Problem (11) in a continuous fashion:

$$\begin{aligned} &\textbf{find} \quad \mathbf{y} \\ &\textbf{s.t.} \quad \mathbf{K}^{\{p\}}\mathbf{y} \leq \mathbf{k}^{\{p\}} \\ &\qquad \mathbf{H}^{\{p\}}\mathbf{P}_{\mathbf{w}i}^y \mathbf{y} \leq \mathbf{h}^{\{p\}} - \mathbf{H}^{\{p\}}\mathbf{P}_{\mathbf{w}i}^s \quad , \forall i \end{aligned} \tag{13}$$

In this case, the whole trajectory necessarily satisfies the constraints everywhere, as they form a convex set.

D. Application to a motion with one contact switch

We now consider the case where $q = p+1$. In this case we define $T^{\{p\}}$ and $T^{\{q\}}$ as the time spent in each phase, such that $T = T^{\{p\}} + T^{\{q\}}$.

When a contact switch occurs during a motion, the constraints applied to the CoM trajectory change at the switching time $t = T^{\{p\}}$. When $t < T^{\{p\}}$, the constraints of phase $\{p\}$ must be applied and conversely, the constraints of phase $\{q\}$ must be applied and when $t > T^{\{p\}}$. At $t = T^{\{p\}}$, the constraints of both phases must be applied.

1) Discrete formulation: Adapting the discretized FP (11) to this case is straightforward: the formulation remains the same, with the only difference that the constraints that must be verified at each discretized point change at $t = T^{\{p\}}$ and $t > T^{\{p\}}$. We thus have 3 sets of constraints in this case: one for each of the two phases, plus one for the transition time $t = T^{\{p\}}$ where the constraints of both phases apply. We define $J^{\{q\}} : \{j \in \mathbb{N}, T^{\{q-1\}} \leq j\Delta t \leq T^{\{q\}}\}$ and obtain the following FP:

$$\begin{aligned} &\textbf{find} \quad \mathbf{y} \\ &\textbf{s.t.} \quad \mathbf{E}_j^{\{z\}}\mathbf{y} \leq \mathbf{e}_j^{\{z\}} \quad ,\forall j \in J^{\{z\}}, \forall z \in \{p,q\} \end{aligned} \tag{14}$$

2) Continuous formulation: In this case, since $\mathbf{w}(t)$ spans 2 distinct sets of linear inequalities, the convex hull of its control points is not guaranteed to lie in the constraint set. The key idea, unlike Lengagne et al. [**?**], is to fall back to the case where no contact switch occurs, by considering two curves that continuously connect at the switching time $T^{\{p\}}$. A similar approach has been proposed before, in the context of UAVs [**?**], with the difference that in our case the continuity of the trajectory is guaranteed by the De Casteljau decomposition algorithm. This algorithm divides the original curve into two curves $\mathbf{c}(t,\mathbf{y})$, each curve being subject to the constraints of their respective contact phase (see Figure 4). The result is thus the expression of the control points of two Bezier curves $\mathbf{c}_{\{p\}}(t,\mathbf{y})$ and $\mathbf{c}_{\{q\}}(t,\mathbf{y})$ with the same degree as the original curve, such that :

$$\begin{cases} \mathbf{c}_{\{p\}}(t,\mathbf{y}) = \mathbf{c}(t,\mathbf{y}) & \forall t \in [0; T^{\{p\}}] \\ \mathbf{c}_{\{q\}}(t,\mathbf{y}) = \mathbf{c}(t,\mathbf{y}) & \forall t \in [T^{\{p\}}; T] \end{cases} \tag{15}$$

The De Casteljau decomposition guarantees that $\mathbf{c}_{\{p\}}(T^{\{p\}},\mathbf{y}) = \mathbf{c}_{\{q\}}(T^{\{p\}},\mathbf{y})$, and that the composition of the curves in infinitely differentiable ($\mathcal{C}^\infty$), as it is strictly equivalent to $\mathbf{c}(t,\mathbf{y})$. The control points of the new curves are linearly dependent on the control points of the original un-split curve, and thus have the following form:

$$\mathbf{c}_{\{z\}}(t,\mathbf{y}) = \sum_{i=0}^{n} B_i^n(t/T^{\{z\}})\mathbf{P}_i^{\{z\}}(\mathbf{y}) \quad \forall z \in \{p,q\} \tag{16}$$

where the $\mathbf{P}_i^{\{z\}}(\mathbf{y})$ have the form:

$$\mathbf{P}_i^{\{z\}}(\mathbf{y}) = \mathbf{P}_i^{y\{z\}}\mathbf{y} + \mathbf{P}_i^{s\{z\}} \tag{17}$$

with $\mathbf{P}_i^{y\{z\}}$ and $\mathbf{P}_i^{s\{z\}}$ constants.

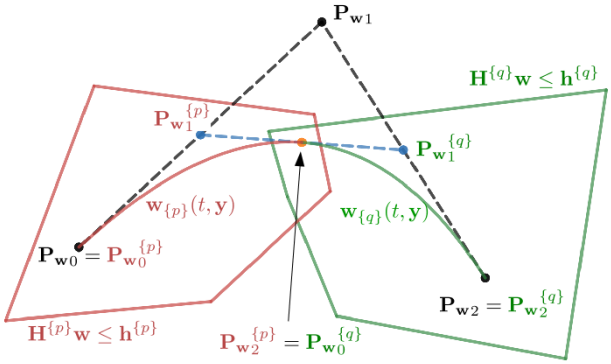

Fig. 4: Example of curve decomposition with the De Casteljau algorithm. The original curve comprises 3 control points (black). It is decomposed into two curves comprising the same number of control points each (3). We can then constrain the control points of the first curve (red) to lie in the first set of constraints, and similarly constrain the control points of the second curve (green) to lie in the second set of constraints. As a result, if the constraints can be satisfied, the connecting control point of both curves satisfies both set of constraints, and we obtain the guarantee that each sub-curve satisfies its respective set of constraints. Interestingly, the control points of the sub-curves are constrained to belong to their respective cones, but those of the original curve can lie outside of the constraints.

It follows that $\mathbf{w}_{\{p\}}(t,\mathbf{y})$ and $\mathbf{w}_{\{q\}}(t,\mathbf{y})$ are also linearly dependent of $\mathbf{y}$:

$$\begin{aligned} \mathbf{w}_{\{z\}}(t,\mathbf{y}) &= \sum_{j=0}^{2n-3} B_j^{2n-3}(t/T^{\{z\}})\mathbf{P}_{\mathbf{w}j}^{\{z\}}(\mathbf{y}) \\ \text{with } \mathbf{P}_{\mathbf{w}j}^{\{z\}}(\mathbf{y}) &= \mathbf{P}_{\mathbf{w}j}^{y\{z\}}\mathbf{y} + \mathbf{P}_{\mathbf{w}j}^{s\{z\}} \quad ,\forall z \in \{p,q\} \end{aligned} \tag{18}$$

Finally the constraints of (13) can be rewritten to deal with the contact switches. The kinematic constraints expressed at each control points are written:

$$\underbrace{\mathbf{K}^{\{z\}}\mathbf{P}_i^{y\{z\}}}_{\mathbf{A}_i^{\{z\}}}\mathbf{y} \leq \underbrace{\mathbf{k}^{\{z\}} + \mathbf{K}^{\{z\}}\mathbf{P}_i^{s\{z\}}}_{\mathbf{a}_i^{\{z\}}}, \forall i, \forall z \in \{p,q\} \tag{19}$$

and the dynamic constraints:

$$\begin{aligned} \underbrace{(\mathbf{H}^{\{z\}}\mathbf{P}_{\mathbf{w}j}^{y\{z\}}}_{\mathbf{D}_j^{\{z\}}}\mathbf{y} \leq \underbrace{\mathbf{h}^{\{z\}} - \mathbf{H}^{\{z\}}\mathbf{P}_{\mathbf{w}j}^{s\{z\}}}_{\mathbf{d}_j^{\{z\}}}, \\ \forall j, \forall z \in \{p,q\} \end{aligned} \tag{20}$$

We can then stack the constraints:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_0^{\{p\}} \\ \vdots \\ \mathbf{A}_n^{\{p\}} \\ \mathbf{A}_0^{\{q\}} \\ \vdots \\ \mathbf{A}_n^{\{q\}} \end{bmatrix} \mathbf{a} = \begin{bmatrix} \mathbf{a}_0^{\{p\}} \\ \vdots \\ \mathbf{a}_n^{\{p\}} \\ \mathbf{a}_0^{\{q\}} \\ \vdots \\ \mathbf{a}_n^{\{q\}} \end{bmatrix} \mathbf{D} = \begin{bmatrix} \mathbf{D}_0^{\{p\}} \\ \vdots \\ \mathbf{D}_{2n-3}^{\{p\}} \\ \mathbf{D}_0^{\{q\}} \\ \vdots \\ \mathbf{D}_{2n-3}^{\{q\}} \end{bmatrix} \mathbf{d} = \begin{bmatrix} \mathbf{d}_0^{\{p\}} \\ \vdots \\ \mathbf{d}_{2n-3}^{\{p\}} \\ \mathbf{d}_0^{\{q\}} \\ \vdots \\ \mathbf{d}_{2n-3}^{\{q\}} \end{bmatrix} \tag{21}$$

We recall that in our case $n = 6$. Finally, we can rewrite FP (13) with a contact switch as:

$$\begin{aligned} \textbf{find} \quad & \mathbf{y} \\ \textbf{s.t.} \quad & \mathbf{A}\mathbf{y} \leq \mathbf{a} \\ & \mathbf{D}\mathbf{y} \leq \mathbf{d} \end{aligned} \tag{22}$$

This boils down to check if each control point of each split curve satisfies the constraints of the current contact phase.

E. General case

In the general case, the same idea will apply. In the continuous case, we use the De Casteljau algorithm to split $\mathbf{c}(t)$ into as many curves as required, thus falling back to a formulation with no contact switches. In the discrete case, we assign the appropriate constraints for each discretized time step. While these decompositions appear mathematically heavy, from a programming point of view, they can be automatically generated, and thus are in fact simple to implement.

In our experiments, we only consider three consecutive phases (which correspond to one step), and solve a new problem for each subsequent set of phases. We call one such convex problem "*CROC*", which stands for *Convex Resolution Of Centroidal dynamic trajectories*.

F. Non-conservative continuous formulation

The presented continuous formulation is more conservative than the discretized one. Constraining the control points to lie inside the constraint set prevents from the generation of curves such as the one illustrated in Figure 5.

However, by relying on the De Casteljau algorithm, it is possible to continuously satisfy the constraints while considering control points outside of the constraint set. Indeed when a curve is split, the constraints no longer apply to the control points of the original curve, but to the control points of the sub-curves. This is illustrated in Figure 4. If the curve is split an infinite number of times, it is straightforward to see that the original curve can span entirely its original definition set as the position of the control points converge to the original curve as the number of split increase.

The price to pay is that the number of constraints increases with the number of curve splittings: a curve of degree s split b times comprises $(s+1)*(b+1)$ constraints. The higher the number of splits is, more the number of constraints to address increases. A parallel can be made with the discretized approach: the lower the discretization step is, the higher the number of constraints is.

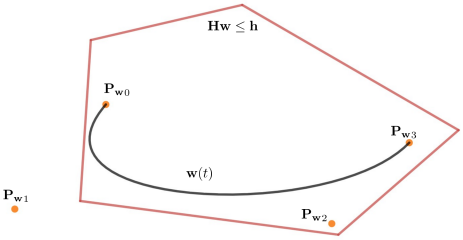

Fig. 5: The curve $\mathbf{w}(t)$ belongs entirely to the convex boundaries (red), while a control point $\mathbf{P}_{\mathbf{w}1}$ lies outside of them.

We believe that a deeper analysis of the pros and cons of using a continuous formulation, not only in the case of *CROC*, but with any other formulation of the problem, requires a significant amount of research, and thus will be discussed in a future work. In this paper, we only divide the curve at the transition points, and we show in our experiments that this is already sufficient to perform similarly to the discretized approaches, while ensuring comparable time performances.

G. Cost function and additional constraints

As the transition feasibility problem is addressed by *CROC*, a feasible CoM trajectory is computed. It is possible to optimize this trajectory to minimize a given cost function $l(\mathbf{y})$, either linear or quadratic. In the latter case the FP problem (22) then becomes a Quadratic Program (QP). One can for instance minimize the integral of the squared acceleration norm or the angular momentum. This cost function is irrelevant to solve the transition feasibility problem, but it can be later used as a reference CoM trajectory for a whole-body motion generator, or as an initial guess for a nonlinear solver as discussed in Section IV-F.

The formulation also allows to add inequality constraints on $\mathbf{c}$ and any of its derivatives by rewriting the expression of the control points of the desired curve as done in equation (17). Here again, these constraints can either be verified continuously on the concerned control points, or in a discretized fashion. In any case, they take the form:

$$\mathbf{O}\mathbf{y} \leq \mathbf{o} \tag{23}$$

We use such constraints to impose bounds on the velocity and acceleration of the center of mass or on the angular momentum variation. The most generic form of our continuous problem is thus the following QP:

$$\begin{aligned} \textbf{find} \quad & \mathbf{y} \\ \textbf{min} \quad & l(\mathbf{y}) \\ \textbf{s.t.} \quad & \mathbf{A}\mathbf{y} \leq \mathbf{a} \\ & \mathbf{D}\mathbf{y} \leq \mathbf{d} \\ & \mathbf{O}\mathbf{y} \leq \mathbf{o} \end{aligned} \tag{24}$$

In our experiments we set constraints on the acceleration and velocity and minimize the squared acceleration norm as a cost l. In the remainder of the paper "*CROC*" refers to this generic QP. If nothing is specified, by default *CROC*

refers to the continuous formulation and with the inequalities representation of the dynamic constraints, as in the QP (24).

H. Time sampling

In the previous sections, in order to remain convex when computing $\mathbf{w}(t)$ (equation (6)) we assumed that the duration of each phase $T^{\{p\}}, T^{\{p+1\}}$ and $T^{\{p+2\}}$ was given.

Time can be reintroduced in the problem using a bi-level optimization approach [**?**]. However, in this work we choose a more pragmatic offline-sampling approach to compute relevant timing candidates, which turns out to be lossless among all of our experiment set.

To achieve this, we consider a large variety of instances of the transition problem. We first consider all the scenarios demonstrated in Section V-C (for HRP-2 and HyQ), from which we extract instances of the transition problem. We secondly generate random scenarios (Figure 7). We randomly allocate initial and end velocities for the center of mass along the direction of motion, between 0 and 1.5 $m.s^{-1}$.

For a total of 10 000 instances of the transition problem, we sample various combinations of times, solve the corresponding QPs and check whether a solution is found. In theory, this would mean that we need to sample an infinity of time combinations in order to be complete. However, we pragmatically reduce this number and give up on the completeness while maintaining a high success rate as follows: we sampled a time for each duration phase $T^{\{z\}}$ by choosing a value between 0.1 and 2 seconds for phases without end-effector motion and between 0.5 and 2 seconds for phases with end-effector motion, with increments of 50ms. For a sequence of three phases with one phase with end-effector motion, this gives a total of 43320 possible combinations. We tested *CROC* with all these combinations on various problems : with HRP-2 or HyQ robots on flat and non-coplanar surfaces, for several thousands of states.

Upon analysis of the results of the convergence of the QPs, we found out that we can use a small list of timings combinations (5 in our case, shown in table I) that covers 100% of the success cases for all the robots and scenarios tested. We thus solve a maximum of 5 QPs for each validation. Figure 6 shows the evolution of the success rate according to the number of timings combinations used. We observe that 3 combinations are enough to reach 99% of success but that two additional combinations are required to reach exactly 100%.

The number 100% may appear large. Intuitively however, it seems to highlight the fact that the accuracy of the transition times are not that important for the considered feasibility problem. Indeed $T^{\{p\}}$ constrains the CoM trajectory to lie in the intersection of two contact phase constraints at this precise time. However this intersection is in general of a significant volume. As a result the CoM trajectory will belong to the intersection for a large time window, which results in a significant slack in the selection of time.

We recall that here, we are only concerned in finding feasible times. For instance, typical double support times when walking on flat ground are closer to 0.2 seconds than 1 second for $T^{\{p\}}$ in dynamical cases. However 0.2 seconds is not feasible when starting from a null velocity. In both cases the interval between 0.8 and 1.2 seconds is almost always feasible in our experiments, which explains why such timings were selected for $T^{\{p\}}$. As such, table I should **not** be considered as a table giving optimal contact time durations, but rather one maximizing feasibility over our set of problems.

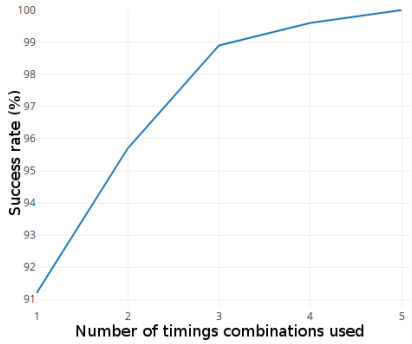

Fig. 6: Evolution of the success rate of *CROC* according to the number of timings combinations used. Tested on various scenarios with coplanar and non-coplanar contacts and with a bipedal and a quadrupedal robots.

timings (s)			Success rate (%)
$T^{\{p\}}$	$T^{\{p+1\}}$	$T^{\{p+2\}}$	
1	0.8	0.8	91.2
1	0.75	0.9	89.2
0.8	0.8	0.9	88.3
0.7	0.5	0.85	77.7
1.2	0.6	1.1	70.8

TABLE I: Success rate with the five used timings combinations.

IV. Performances of *CROC*

A. CROC vs a nonlinear solver

Computing the success rate of our method is a hard task because we do not have any way to determine the "ground truth" feasibility of a transition (ie. there does not exist any method able to determine in finite time whether there exists a valid centroidal trajectory between the two states). We choose to compare the relative success rate of *CROC* with respect to a state-of-the-art non-linear formulation of the same problem [**?**], which is reported to give similar results to the one from Ponton et al. [**?**].

Both approaches share similar formulations in terms of kinematic constraints. Conversely the nonlinear solver does not use the conservative formulation of *CROC* that makes the problem convex, and thus is able to explore a larger part of the solution space, and thus to find a "more optimal" solution of a given locomotion problem.

B. Comparison benchmarks

The scenarios used in our benchmarks consist of randomly generated sequences of 3 contact phases such that:

- both initial and final contact phases are in static equilibrium

- both initial and final contact phases have the same effectors in contact, between two and four
- there is exactly one contact repositioning between both initial and final contact phases and no other contact variation
- the intermediate contact phase is not required to be in static equilibrium.

These benchmarks thus consider the case of a "repositioning" of an end-effector, which encompasses the only two other possible cases, creating a contact or breaking a contact (section III-E).

For this benchmark we considered two kind of scenarios. In the first case, we only sample contact phases with coplanar contacts. In the second case, we sample truly random contacts, which lead to contact phases with non-coplanar contacts and contact sequences that require complex motions. Examples of randomly generated scenarios are shown in Figure 7.

All the benchmarks were run on a single core of an Intel Xeon CPU E5-1630 v3 at 3.7Ghz. The QP problems are solved with QuadProg, and the FP problems with GLPK [**?**].

(a) Unfeasible (b) Feasible (c) Feasible (d) Feasible

Fig. 7: Examples of random contact transitions used for benchmarking. Top row: initial configuration, bottom row: final configuration. (a) and (b) only have both feet in contact, (c) and (d) have both feet and the left hand in contact. All the displayed configurations are in static equilibrium, but the intermediate configuration with one less contact (not shown) is not constrained to be in static equilibrium. None of the methods found a solution for the transition (a), the other transitions were successfully solved.

The first benchmark compares four different methods: both discrete[3] and continuous formulation of *CROC* presented in this paper (using the inequality representation of the constraints), the nonlinear resolution proposed in [**?**] and the same nonlinear method but initialized with the solution found by *CROC* when available. As we compare the relative success rate between the methods, we only consider the scenarios where at least one of the method finds a solution when computing the percentage of success. The results are shown in table II.

[3]with 7 discretization points per contact phases, which corresponds to a time step of approximately 100ms.

Method	**Coplanar** success (%)	**Non-coplanar** success (%)	Total time (ms)
CROC (discrete)	89.7	60.6	3.89
CROC (continuous)	88.4	57.2	3.93
Non-linear	100	94.1	$\simeq$ 150
N-L with init guess	100	100	$\simeq$ 130

TABLE II: Comparison between *CROC* and a non linear solver for randomly generated contact sequences of three contact phases. The two first methods are the ones presented in this paper, with either the discrete[3] or continuous formulation and using the inequality representation of the dynamic constraints. These methods are compared with the non linear solver presented in [**?**], either with their naive initial guess (Non-linear) or with the solution found by *CROC* as an initial guess when available (N-L with init guess). The percentages on the "success" columns only consider the scenario where at least one method found a solution.

C. How conservative is CROC?

Because of its conservative reformulation, *CROC* does not cover the whole solution space. As expected, our method finds less solutions than the nonlinear solver. In the coplanar case, *CROC* almost finds 90% of the solutions. In the non-coplanar case, the centroidal trajectory may be required to present several inflexion points and/or to be really close of the constraints, which cannot be represented using a single variable control point for the trajectory. This explains the difference of success rates between the two cases. However, even in such complex cases *CROC* still finds around 60% of the solutions.

While 60% might appear as a low number, it is important to consider that it corresponds to truly random scenarios. The question of determining how interesting are the remaining 40% of solutions is left for future work. An important take-away message is that in the realistic scenarios considered in this paper (such as stair climbing or uneven terrains), *CROC* covers enough of the solution space to allow to find a solution.

D. Computation time

As claimed in the introduction, *CROC* is about two order of magnitude faster than a state-of-the art nonlinear solver for the centroidal motion generation. For the inequality representation with the double description method, the computation time allocated to solve the QP of equation (24) is extremely fast with $50\mu s$ on average. The computation time of *CROC*, which comprises the time required to solve the QP and the time required to compute all the constraints matrices of equation (21) is around $400\mu s$. The total time in table II also includes the time required by the double description method. However, in some cases the same contact phases may be used several times and the double description method only needs to be computed once per contact phase, thus the time required for the double description may be factorized.

1) Comparison with the equality representation: Table III shows the difference in computation time between the inequality and equality formulation, with a varying number of contacts.

Formulation	Metric	Number of contacts		
		2	3	4
Double-Description	DD time (ms)	3.52	14.88	28.16
	Total time (ms)	3.93	16.18	37.41
Force	Total time (ms)	13.01	25.28	49.65

TABLE III: Comparison between the computation times required to generate and solve the FP[4] defined by *CROC* using either the Double Description (DD) or the Force formulation.

The major difference between the two representations lies in the dimension of the variables and the constraints of the problem, which is greater in the case of the force formulation. As shown in Table III the computation times between the double description and the force formulations remain in the same order of magnitude for 2 to 4 contacts, with an advantage for the double description. However this advantage reduces as the number of contacts increase. Indeed, while the computation time for the force formulation doubles at each additional contact, the time grows cubicly with the Double Description (DD) formulation.

E. Comparing the continuous and discretized formulations

The results of Table II confirm that the continuous formulation presented in section III-C2 is conservative with respect to the discrete formulation. However, these results show only a marginal difference of success rate between the discrete and continuous formulation of *CROC* ($1 - 4\%$). This can first be explained by the fact that the De Casteljau decomposition allows for the control point $\mathbf{y}$ to lie outside of the constraints (Figure 4), thus making the method less restrictive. We propose a second explanation, which is only intuitive (thus not a claim): the remaining missing solutions are necessarily those that will result in the curve lying close to the constraint boundaries. The discretized approach will theoretically find them, but the chances of finding a trajectory partially outside of the constraint sets are much higher in this case (Figure 2).

Moreover, in section III-D2 we proposed to only split the trajectory in one curve for each contact phases but it is possible to split the trajectory in an arbitrary number of curves, as long as each curve is entirely contained in one contact phases, as detailed in section III-F. By increasing the number of split curves, we can further reduce the loss of solutions.

1) Invalid solutions of the discretized methods: Again, the major drawback of a discretized approach is that the portions of the curve in-between two discretization points are never checked and could violate the constraints (Figure 2).

In order to measure this risk four variants of *CROC* were compared with the same randomly generated contact sequence as before: the discretized version with three different values of number of discretization points per phases and the continuous version presented in this paper. The four variants use the inequality representation of the dynamic constraints. Then, for each centroidal trajectory found as a solution, the dynamic constraints were verified with a really small discretization step. If the constraints were not satisfied for at least one point of the trajectory, we count this solution as "invalid".

[4]QP and FP give similar times for the DD formulation, while the FP is much more efficient in the Force formulation. This is only an implementation problem, since GLPK exploits the sparsity of the problem while QuadProg does not.

Method	Invalid solutions (%)		Computation time (ms)
	Coplanar	**Non-coplanar**	
Discrete (3 pts)	10.6	19.7	0.20
Discrete (7 pts)	6.7	9.3	0.37
Discrete (15 pts)	4.2	6.9	0.75
Continuous	0	0	0.41

TABLE IV: Comparison between the method *CROC* with the discrete formulation, with varying number of discretization points, and the continuous formulation presented in this paper.

Table IV shows that the percentage of invalid solutions found by the discrete methods is non negligible. Obviously, as the number of discretization points increase this percentage decreases. As shown in equation (11) the number of constraints in the discretized LP problem is proportional to the number of discretization points. Thus the number of discretization points used is a complex parameter to tune, as it is a compromise between the computation time and the risk of finding invalid solutions. This issue is common to all methods that rely on discretization. It emphasizes the fact that we need a continuous method, able to check exactly whether the whole trajectory is valid with a fixed number of constraints in the problem.

2) Computational advantage of the continuous formulation: Depending on the discretization, the continuous formulation can be slower or faster to compute. However, to reach less than 5 % of false positive trajectories with the discretized approach, table IV shows that the continuous formulation is actually faster.

F. Using CROC to initialize a non linear solver

Choosing an initial guess for the nonlinear solver of a trajectory generation method is essential but may be challenging for multi-contact motions. The quality of this initial guess has a significant influence on the convergence of the nonlinear solver. For the nonlinear method considered in this section [**?**] proposed a naive initial guess of the centroidal trajectory based solely on the position of the contact points.

Interestingly, Table II suggests that the solution set spanned by *CROC* is not strictly included in the one spanned by this nonlinear solver with this naive initial guess. Using the solution of *CROC* to initialize the nonlinear solver can thus help it to converge and increase its success rate. As shown in Table II, this improvement only appears for the non-coplanar case because the naive initial guess used is always close to a valid solution in the coplanar case. We expect that the importance of the initial guess will grow if the contact sequences do not allow static equilibrium configurations at the contact phases, and will check this hypothesis in the future.

Moreover, by using the solution of *CROC* to initialize the nonlinear solver we measured a reduction of the number of iterations required to converge of 20% on average, reducing the total computation time (ie. it is faster to use *CROC* and then the non-linear solver than using the non-linear solver directly).

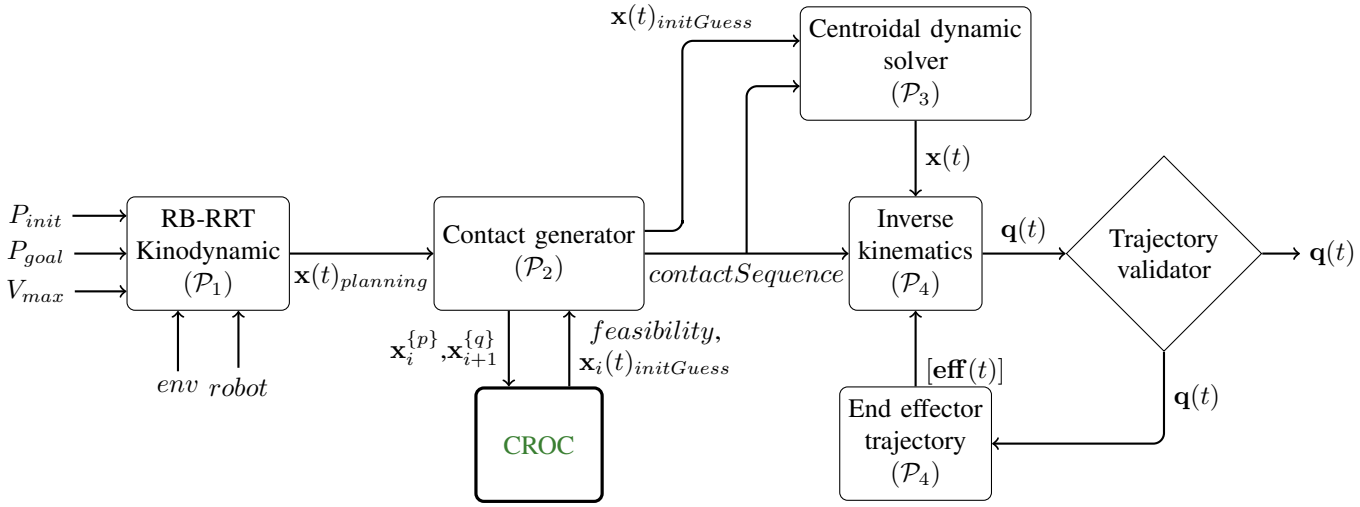

Fig. 8: Complete experimental framework.

V. Experimental results

A. Application to multi-contact planning

We test *CROC* in the context of multi-contact locomotion and evaluate its impact on the success rate of a previously published multi-contact planner. This planner follows the decoupled approach, where the issue of planning the contact locations is decoupled from the generation of a motion [**?**]. The decomposition results into four sub-problems solved sequentially, under a "divide and conquer" strategy:

- $\mathcal{P}_1$ the planning of a trajectory for the root of the robot,
- $\mathcal{P}_2$ the generation of a discrete contact sequence along the root's trajectory,
- $\mathcal{P}_3$ the optimization of the centroidal trajectory of the robot over the whole contact sequence,
- $\mathcal{P}_4$ the generation of a whole-body motion from this contact sequence and centroidal trajectory.

Figure 8 shows the complete architecture used for our experiments, implemented in the Humanoid Path Planner [**?**] framework. The inputs are an initial (respectively goal) position and orientation for the root of the robot, as well as a set of bounds on the velocities and acceleration applying to the CoM and the end-effector, and a complete representation of the 3D environment. The output is a dynamically consistent and collision free whole-body motion which can be executed by a real robot as shown in section V-C. This framework is open-source and we refer the interested reader to its documentation [5] for more details.

The decoupling between each sub-problem allows to break the complexity, but comes with a cost that is the introduction of a feasibility problem: each sub-problem must be solved in the feasibility domain of the next sub-problems: ie. there must exist a sequence of contacts (problem $\mathcal{P}_2$) that can follow the root's trajectory found (solution of $\mathcal{P}_1$), and similarly there must exist a feasible whole-body motion (problem $\mathcal{P}_3$) from the computed contact sequence (solution of $\mathcal{P}_2$). The latter problem is an instance of the transition feasibility problem addressed in this paper (the former was considered in [**?**]).

[5] https://github.com/loco-3d/multicontact-locomotion-planning

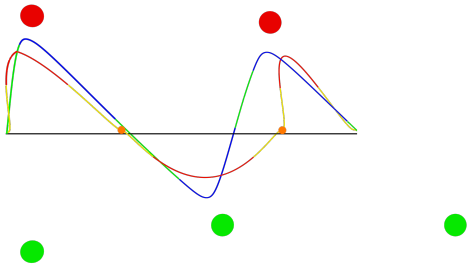

Fig. 9: Example of centroidal trajectories generated with CROC and a nonlinear solver (bird eye view), in a case of bipedal walking. The red and green circles represent the contact positions of the (respectively) left and right feet centers over time. The red and yellow (respectively related to single and double support phases) curve is the curve obtained through the concatenation of curves computed with CROC. The blue and green (respectively related to single and double support phases) curve is obtained through optimization of the latter curve with a nonlinear solver. The orange circles represent the constrained COM positions resulting from the contact planning phase, which are ignored by the nonlinear solver to produce smoother motions.

B. CROC as a feasibility criterion during contact planning

In this paper, we only modify the contact generation method by adding *CROC* as a feasibility criterion, as shown in the green block of Figure 8. It is important to observe that in this context, establishing the transition feasibility as fast as possible is crucial: $\mathcal{P}_2$ is a combinatorial problem, which implies that many contact sequences (thousands) must possibly be tried before finding a feasible contact sequence.

CROC can be efficiently used as a feasibility criterion during the contact planning phase with a trial-and-error approach. More precisely it is used as a filter to determine

which transitions are unfeasible and discard them during the planning in order to produce contact sequence containing only feasible transitions. *CROC* will thus be called for each contact transition considered by the contact generator ($\mathbf{x}_i^{\{p\}}$ and $\mathbf{x}_{i+1}^{\{q\}}$ in Figure 8) and output the feasibility of the given contact transition.

A byproduct of the feasibility test made with *CROC* is a feasible CoM trajectory between each adjacent contact phases ($\mathbf{x}(t)_{initGuess}$). The composition of all these trajectories is given as an initial guess for a non-linear solver tackling the $\mathcal{P}_3$ sub-problem as discussed in section IV-F. The $\mathcal{P}_3$ solver refines the global trajectory by optimizing over the whole contact sequence (Figure 9). As expected, when provided with such an initial guess the non linear solver converges 100% of the time, while we don't have guarantees that the solution to $\mathcal{P}_3$ defines a feasible $\mathcal{P}_4$.

C. Experimental scenarios

The complete experimental framework was tested on several locomotion scenarios in semi structured environments, each scenario showing specific features or difficulties. We insist that the only manual inputs given to our framework were an initial and a goal position for the root of the robot. Most of the obtained motions are demonstrated in the companion video. They were validated either in a dynamics simulator or on the real robot.

1) Inclined platform crossing: This scenario requires the robot to go from one flat platform to the other by taking a step on an inclined platform (Figure 10). The scenario is designed such that no quasi-static solution exists to the problem, and is truly multi-contact for two reasons: firstly part of the motion occurs entirely on non-flat ground; secondly the problem is unfeasible if the right foot is the one selected to go first on the platform. *CROC* then allows to invalidate unfeasible contact sequences that would involve directly taking a step on the final platform, or take a step with the right foot first (Figure 11). It rather allows to find a solution where the left foot is used to step on the inclined platform (Figure 10). A feasible whole-body motion is demonstrated in the companion video.

Additionally, *CROC* also ensures that the left foot is positioned in such a way that the problem becomes feasible, which is not trivial considering the size of the solution space for the chosen step position (Figure 12(a)).

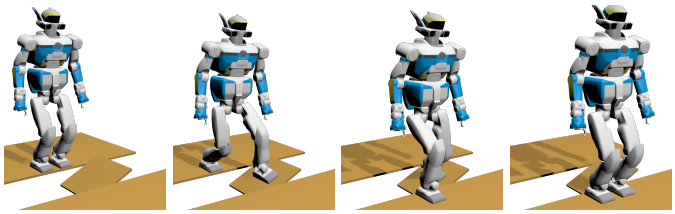

Fig. 10: Platform crossing scenario: no quasi-static solution exists for the flying phase where the left foot is on the inclined platform.

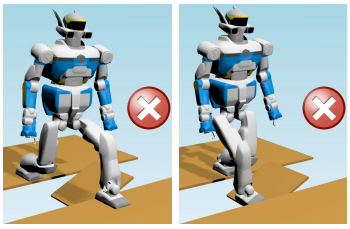

Fig. 11: Unfeasible stepping strategies invalidated by *CROC*.

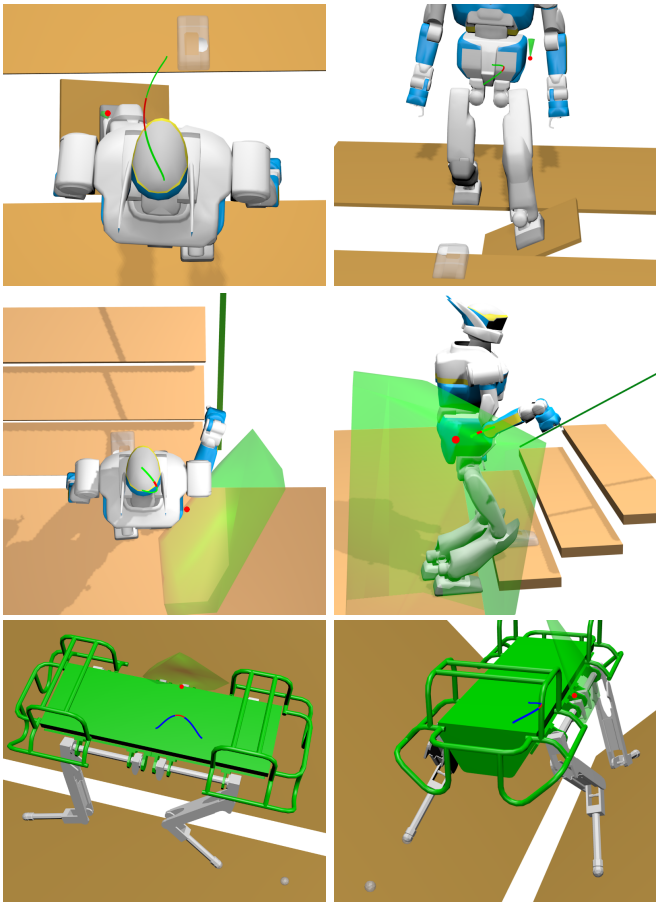

Fig. 12: Examples of centroidal trajectories found by our method. Green polytopes : valid position of $\mathbf{y}$ that verifies the constraints of the problem (24), red sphere : solution found for $\mathbf{y}$ for a given cost function (minimum of the squared acceleration norm). The red part of the trajectory is for the phase with $n_c - 1$ active contacts. The next contact is shown in transparency.

2) 10 cm high steps: This experimental setup is an industrial set of stairs shown in Figure 13 and 16(a). It consists of six 10 cm high and 30 cm long steps. This experiment was done with the HRP-2 robot. All the valid contact sequences produced contain at least 13 contact phases as the robot is kinematically constrained to put both feet on each step.

The complete motion is shown in the companion video. The crouching walk seen is required to avoid singularities in the knee of the extending leg, which are not tolerated by the low-level controller.

An example of unfeasible contact sequence filtered out by

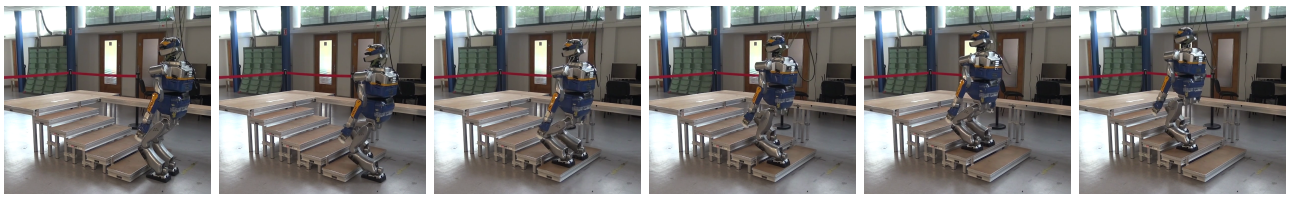

Fig. 13: Snapshots of the motion for the 10cm stairs, the complete motion is shown in the companion video.

our feasibility criterion is depicted on Figure 14. All three configurations in this sequence are valid (ie. respect kinematics and dynamics constraints) but there is no valid centroidal trajectory between the last two configurations. Our feasibility criterion will filter out this kind of contact transitions during contact planning.

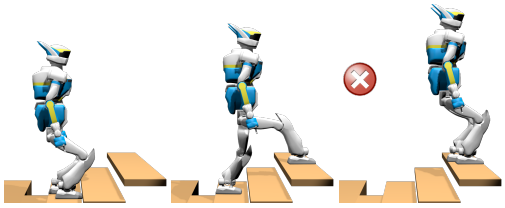

Fig. 14: Exemple of unfeasible contact transition detected by *CROC* and rejected during contact planning

3) 15 cm high steps with handrail: This other set of stairs is composed of four 15 cm high steps and equipped with a handrail. The contact sequence is shown in Figure 16(b) and snapshots of the motion are shown in Figure 15. This is a typical multi-contact problem, showing an acyclic contact sequence with non co-planar contact surfaces. The problem was already solved in a previous work [**?**], but the input contact sequence and effector trajectories had to be manually selected from a large number of trials. In this paper, the only input is a root goal position at the top of the stairs.

A example of centroidal trajectory found by *CROC* for one contact transition in this scenario is shown in Figure 12(b).

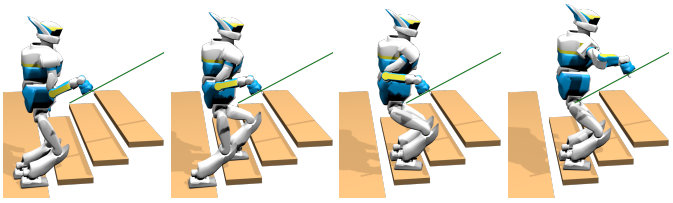

Fig. 15: A feasible multi-contact sequence for a stair climbing with handrail support on the HRP-2 robot automatically computed with our contact planner and *CROC*.

4) Uneven platforms: This setup consists of 30 cm long and 20 cm wide platforms, oriented of 15°around either the x or y axis. This scenario is particularly difficult for the contact planner because of all the possible collisions generated by the feet. We recall that the feet of HRP-2 are 24 cm long for 14 cm wide, which means that the platforms of this setup are only a few centimeters bigger than the feet of the robot. Because of this, there are only few collision free candidates positions for the feet. The probability of finding a contact position which leads to a collision-free configuration while maintaining the equilibrium is extremely small for this setup.

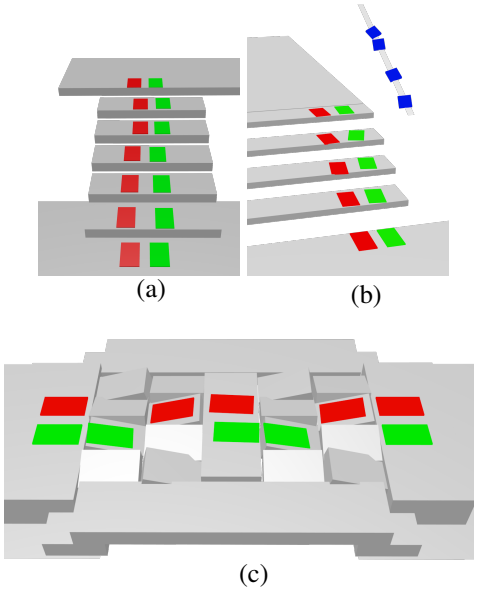

Fig. 16: Examples of contact sequences found with our framework. The color patches represent the planned contact location: green for right foot, red for left foot, blue for right hand.

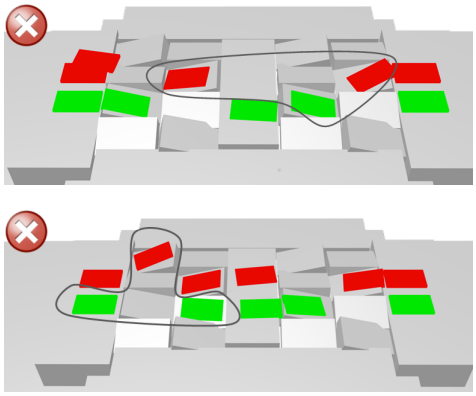

Fig. 17: Examples of unfeasible contact sequences filtered out by *CROC*. There does not exist any valid centroidal trajectory for the contact transitions encircled in black.

The contact sequence found is shown in Figure 16(c), snapshots of the motion are shown in Figure 1 and a motion for this scenario is shown in the companion video. These motions have been validated on the real robot.

The Figure 17 shows two examples of unfeasible contact sequences filtered out by *CROC* in this scenario.

5) Quadrupedal between inclined planes: The quadrupedal robot HyQ navigates between two planes inclined at $45°$. Figure 12(c) shows the the centroidal trajectory found by *CROC* in this scenario for one contact transition. This scenario highlights that *CROC* is suited for any type of legged robot.

D. Benchmarks of the complete framework

In order to quantify the improvement of our contact planner from the use of *CROC* as a feasibility criterion, we used the following test procedure: for each scenario (we also considered a flat ground scenario with no obstacles), we tried to solve the problem using the complete framework presented in section V-A with and without using *CROC* as a feasibility criterion during the contact planning. We recall that this framework takes as input an initial and goal position for the geometric root of the robot and produces as output a whole body motion.

We then measured the success rate and the computation time of the complete framework in both cases, the results are shown in Table V.

Scenario	Method	Motion duration (s)	Total time (s)	Success (%)
Walk (3 steps)	Without *CROC*	7.7	4.48	98
	With *CROC*		4.43	100
Stairs	Without *CROC*	16.23	12.90	47
	With *CROC*		12.56	90.5
Stairs (handrail)	Without *CROC*	23.13	18.38	27.3
	With *CROC*		18.09	88.05
Uneven platforms	Without *CROC*	14.94	15.22	12.5
	With *CROC*		17.83	83.5

TABLE V: Performance analysis of the complete motion planning framework presented in section V-A, with and without using *CROC* as a feasibility criterion during contact planning. *Motion duration* is the average duration of the solution, *total time* is the average computation time required to compute the motion, without the time required to compute the end-effector trajectories. *Success* is the success rate of the complete framework, ie. whether the framework was able to produce a valid motion reaching the goal.

1) Success rate: In the walking on flat ground scenario, *CROC* brings only a marginal improvement to the success rate because the heuristics previously used by the contact planner were sufficient in this case to provide a feasible contact plan most of the time. However, in all the other cases the results empirically prove one claim of this paper: using *CROC* as a feasibility criterion during the contact generation greatly increases the success rate of the multi-contact planning framework.

We observe that when using *CROC* the success rate is close to 100% except for complex scenarios where it is still above 80% in the worst case. From this result, we show that the integration of CROC to our pipeline provides additional guarantees that the computed contact sequence will lead to a valid CoM trajectory and thus that the centroïdal dynamics solver will converge with this contact sequence as input.

The only remaining cause of failure in our framework is the sub-problem $\mathcal{P}_4$. Some cases of failure come from the method used to solve this sub-problem as this method is not complete and may fail to produce a valid solution for a feasible $\mathcal{P}_4$ sub-problem. But other cases of failure comes from the fact that the feasibility of $\mathcal{P}_4$ is not accurately formulated. Indeed, the methods used to solve $\mathcal{P}_3$ and the method *CROC* used in $\mathcal{P}_2$ approximate the whole-body kinematic constraints, which may lead to unfeasible solutions given as input to $\mathcal{P}_4$. Additionally the feasibility of the end-effector motion between two contact locations is not verified, neither in $\mathcal{P}_2$ or $\mathcal{P}_3$. This issue is shared with all the centroidal approaches, and does not penalize *CROC* with respect to them. The kinematic constraints described in the present paper could be made more conservative to always guarantee kinematic feasibility. In the context of our framework however, we prefer to sacrifice accuracy for a larger exploration of the solution space.

2) Computation time: Concerning the computation time, in most of the cases we achieve interactive performances (ie. the computation time is smaller than the motion duration). In the worst case the computation time is greater than the motion duration, but only by a small margin.

When using *CROC* during the contact planning, the computation time required by the contact planning sub-problem increase. This is explained partly by the addition of the time required to run *CROC* for each candidates, but mostly by the fact than we need to evaluate a lot more candidates before we find a valid one (ie. which leads to a feasible transition).

However, thanks to the initial guess of the centroidal trajectory provided by *CROC*, the sub-problem $\mathcal{P}_3$ can be solved faster as explained in the section IV-F.

Depending on the scenario, these two aspects nearly balance themselves. In the end, the results of the Table V shows that we can use *CROC* as a feasibility criterion during contact planning without increasing too much the total computation time required, even though we have to run hundreds of instances of *CROC*.

VI. Conclusion

In this paper we introduce an efficient formulation of the centroidal dynamics of a legged robot, named *CROC*. Our method can compute CoM trajectories that do not require discretization, nor use approximation or relaxation of the dynamic constraints. This formulation is convex yet conservative, but not limited to quasi-static motions. To our knowledge, this is the first method to combine all these properties.

Thanks to the computational efficiency of our method, requiring only a few milliseconds to solve the centroidal dynamic problem with three contact phases, we can use *CROC* as a feasibility criterion during contact planning. The interest of this feasibility criterion has been demonstrated both qualitatively and empirically.

Moreover, the centroidal trajectory produced by *CROC* can be used to provide a relevant initial guess to a non linear solver, resulting in the improvement on the convergence rate and computation time of the non linear solver by comparison to the naive initial guess previously used.

This paper also proposes a continuous formulation of the centroidal dynamics, not restricted to *CROC*. It allows to verify

continuously the constraints of the CoM, by opposition to the discretized methods of the state of the art that only guarantee that the discretized points of the trajectory are valid. We showed that the discretization may lead to a non negligible amount of invalid solutions where the trajectory is invalid between two valid discretization points, which emphasizes the interest of a continuous formulation. We believe that this continuous formulation of the constraints on the centroidal trajectory may be useful for all state-of-the-art methods, convex or non-linear. We leave the study of the feasibility and the interest of this application to a future work.

Finally, the feasibility criterion proposed in this paper permits us to complete our locomotion planning framework [**?**]. In this paper we showed that our framework is able to produce indifferently simple walking motions and multi-contact motions (ie. with non coplanar contacts and acyclic behaviors). These motions were validated in simulation or on the robot HRP-2. We also showed empirically that our framework presents a success rate close to 100% and present interactive computation times (the time required to compute a motion is smaller than the duration of this motion) in the studied scenarios, except for the most complex scenario where the computation time is approximately 20% greater than the duration of the motion, but still remain in the same order of magnitude. We believe that with an optimization of the implementation, interactive performances could be achieved even in the worst cases.

For future work we would like to try more complex motions on the real robotic platform, but we are currently limited by the capabilities of our low level controller.

A. Handling whole-body approximations and uncertainties

The remaining source of approximation is shared with all centroidal-based methods, and comes from the whole-body constraints (joint limits, angular momentum and torques), which are only approximated or ignored in the current formulation. One solution to address the other limitations of the centroidal model could be to alternate centroidal optimization with whole-body optimization as other approaches do [**?**], however for the transition feasibility problem, this approach would result in an increased computational burden that is not compatible with the combinatorial aspect of the search. One way to improve the quality of this approximation is to integrate torque constraints [**?**], [**?**]. Expressing such constraints at the CoM level is considered for future work.

B. Application to 0 and 1 step capturability

The N-Step capturability problem consists in determining the ability of a robot (in a given state) to come to a stop (ie. null velocity and acceleration) without falling by taking at most N steps. It is used to detect and prevent fall.

We can easily change the constraints on $\mathbf{c}(t)$ defined in subsection III-A to remove the constraint on $\mathbf{c}_g$ and constrain $(\dot{\mathbf{c}}_g = \mathbf{0}, \ddot{\mathbf{c}}_g = \mathbf{0})$. With this set of constraints, the feasibility of FP (13) determines the 0-Step capturability. Similarly, FP (22) determines the 1-Step capturability.

For future work we would like to empirically determine the accuracy of our method with respect to this problem, using a framework similar to [**?**].

SOURCE CODE

Code available (C++/python) under a BSD-2 license:
https://github.com/humanoid-path-planner/hpp-bezier-com-traj

ACKNOWLEDGMENT

Supports: ANR LOCO3D ANR-16-CE33-0003, ERC Actanthrope ERC-2013-ADG, H2020 Memmo ICT-780684.

Pierre Fernbach is in a post-doctoral research position in the Gepetto team at LAAS-CNRS, Toulouse, France. He obtained his Ph.D. degrees in robotics from University Toulouse III, France, in 2018. His research interest include motion planning and robotics, with a focus on motion and contact planning for locomotion of legged robots in multi-contact scenario.

Steve Tonneau is a lecturer at the University of Edinburgh. He defended his Phd in 2015 after 3 years in the INRIA/IRISA Mimetic research team, and pursued a post-doc in robotics at LAAS-CNRS in Toulouse, within the Gepetto team. His research focuses on motion planning based on the biomechanical analysis of motion invariants. Applications include computer graphics animation as well as robotics.

Olivier Stasse received in 2000 a Ph.D. in Intelligent Systems from the University of Paris 6, and the French Habilitation to Supervise Research (HDR) in Robotics (2013) from the University of Toulouse III. From 2000 to 2003, he was assistant professor at the Univ. of Paris XIII. From 2003 to 2011, he was at the Joint French-Japanese Robotics Laboratory (JRL) between the CNRS and the AIST in Tsukuba. In 2011 he joined the Gepetto team at LAAS, Toulouse.

Justin Carpentier is a researcher at Inria and Ecole Normale Suprieure de Paris, inside the Willow research team. He graduated from Ecole Normale Suprieure Paris-Saclay in 2014 and received a Ph.D. in Robotics in 2017 from the University of Toulouse, within the Gepetto team at LAAS-CNRS in Toulouse. His research is devoted to the embedding of Optimal Control theory inside the formalism of Machine Learning, with Legged Robotics as a main target application.

Michel Taïx Michel Taïx received the Ph.D. degree in Robotics from Paul Sabatier University at Toulouse in 1991. He is currently an Assistant Professor in the Robotics and Automation Department at the University Paul Sabatier. He is completing his research at the Laboratoire dAutomatique et dAnalyse des Systmes of CNRS in Toulouse. His research interests include motion planning, interactive planning, planning for humanoid robots and planning for compliant and deformable objects.