2PAC: Two Point Attractors for Center of Mass Trajectories in Multi Contact Scenarios

Steve Tonneau, Andrea Del Prete, Julien Pettré, Nicolas Mansard

2PAC: Two Point Attractors for Center of Mass Trajectories in Multi Contact Scenarios

STEVE TONNEAU, LAAS-CNRS
ANDREA DEL PRETE, LAAS-CNRS
JULIEN PETTRÉ, Inria Rennes
NICOLAS MANSARD, LAAS-CNRS

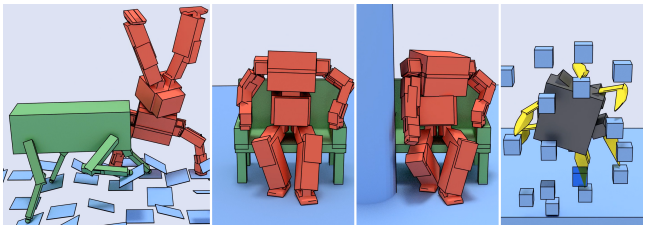

Fig. 1. Multi contact motions synthesized by our method.

Synthesizing motions for legged characters in arbitrary environments is a long-standing problem that has recently received a lot of attention from the computer graphics community.

We tackle this problem with a procedural approach that is generic, fully automatic and independent from motion capture data. The main contribution of this paper is a point-mass-model-based method to synthesize Center Of Mass trajectories. These trajectories are then used to generate the whole-body motion of the character.

The use of a point mass model often results in physically inconsistent motions and joint limit violations. We mitigate these issues through the use of a novel formulation of the kinematic constraints which allows us to generate a quasi-static Center Of Mass trajectory, in a way that is both user-friendly and computationally efficient.

We also show that the quasi-static constraint can be relaxed to generate motions usable for applications of computer graphics (on average 83% of a given trajectory remain physically consistent).

Our method was integrated in our open-source contact planner and tested with different scenarios - some never addressed before- featuring legged characters performing non-gaited motions in cluttered environments. The computational efficiency of our trajectory generation algorithm (under ten ms to compute one second of motion) enables us to synthesize motions in a few seconds, one order of magnitude faster than state-of-the-art methods.

CCS Concepts: • **Computing methodologies** → **Procedural animation**; *Physical simulation*;

Additional Key Words and Phrases: Character animation, multi contact locomotion, motion planning, center of mass

Reference format:
Steve Tonneau, Andrea Del Prete, Julien Pettré, and Nicolas Mansard. 2017. 2PAC: Two Point Attractors for Center of Mass Trajectories in Multi Contact Scenarios. 1, 1, Article 1 (October 2017), 10 pages.
https://doi.org/0000001.0000001_2

1 INTRODUCTION

The automatic synthesis of complex motions for legged characters is an open problem, actively researched by both the computer graphics and robotics communities. One challenge is to provide characters with enough autonomy of motion so as to facilitate character animation tasks, or to animate non-player characters in games. Ideally, a method to synthesize believable motions would only require simple high level inputs (e.g., initial and final positions), and not tedious manual editing from an animator. It would also generalize well to large sets of legged characters (with any number of legs), and not require the gathering of large motion capture datasets for each character. Also, it would not be restricted to flat ground or annotated environments. Finally, it would be computationally efficient, close to real time, and not let animators wait for too long before obtaining a result. It is certainly challenging to fulfill all these requirements. In this paper, we consider the problem of computing a physically plausible motion between two positions for a legged character in an arbitrary environment. We demonstrate significant progress with respect to all the objectives listed above: generalization over various characters, physical accuracy and plausibility, environment variety and complexity, computational performance.

This problem has received a lot of attention over the past decade. As a result the synthesis of walking or running motions in open environments (flat ground, few obstacles) is now commonly and efficiently addressed by existing methods [Kajita et al. 2003; Yin et al. 2007]. In this context where the contact phases are cyclic, the kinematic constraints of the avatar can be easily expressed [Perrin et al. 2012]. Furthermore the dynamics of the avatar can be accurately approximated by assuming that the ground is always flat. *Multi contact locomotion* is the general case that we consider, where all these simplifying assumptions no longer hold. Examples of multi contact motions include climbing, walking on highly uneven terrain or climbing stairs using a handrail. The issue of finding when, where and with which effector to create a contact leads to a high dimensional combinatorial problem. Additionally, the dynamics of the system is more complex, and the kinematic constraints are harder to express. Finally, motions occur in the proximity of obstacles which makes the collision avoidance non trivial. As a result, motion synthesis is hard to generalize because small changes in the environment affect significantly the solution space.

More recent approaches reduce the dimension of the problem by considering a point mass model (3 degrees of freedom) rather

than the complete avatar (60 degrees of freedom). They then focus on planning a trajectory for the center of mass (COM), with significant success. However, this approximation often leads to joint limit violations once the whole-body motion is computed. Moreover, the angular momentum generated by the character bodies is unknown when computing the COM trajectory, so there is no guarantee that the resulting motion will be dynamically consistent. Furthermore, finding a trajectory is usually formulated as a nonconvex optimization problem, computationally expensive, with no guarantee of convergence.

In this paper we propose a different approach to provide guarantees of kinematic and dynamic feasibility. Rather than relying on a simplified model, we first consider a conservative but exact formulation of the dynamics, later relaxed while preserving the kinematic constraints of the motion. Our approach is agnostic to the character morphology, allowing us to consider a large range of complex environments.

Our method is detailed in two steps. In a first step we restrict ourselves to quasi-static locomotion, where the robot moves at low velocity. In this context the point mass model dynamics is accurate because the angular momentum is negligible. Considering a new formulation of the kinematic constraints, we show that finding a trajectory connecting two contact phases boils down to finding a single point that uniquely defines such a solution, and which can be computed efficiently. In a second step we show that we can relax the quasi-static constraint to generate smoother trajectories while maintaining kinematic feasibility, in a flexible way that can be parametrized by the user. Thanks to this property, our method is robust to failure and always proposes a kinematically feasible solution (assuming one exists).

Our main contribution is thus a fully automatic and procedural method to compute a kinematically feasible, quasi-static COM trajectory in multi contact situations, for arbitrary creatures, environments and scenarios. This is achieved thanks to two technical contributions:

(1) a new formulation of the kinematic constraints for the point mass model;
(2) a one-step algorithm to compute a feasible trajectory, the fastest to our knowledge.

Additionally, we demonstrate that our method can be integrated with an open-source contact planner to generate whole-body motions in a large variety of scenarios.

2 STATE OF THE ART

The synthesis of multi contact locomotion is a subject of rising interest for both the robotics and computer graphics communities, which follows many years of research on gaited locomotion. It remains largely open due to the difficulty of handling arbitrary environmental constraints. Both procedural and data-driven approaches currently converge towards simplified models for handling the complex dynamics of legged characters. We believe that improving these models is the key to synthesizing more realistic motions.

2.1 Procedural approaches to multi contact locomotion

After years of research, efficient simplifying models have been proposed to synthesize walking or running motions on the fly, such as the linear inverted pendulum model [Kajita et al. 2003] which led to the introduction of the capture point [Pratt et al. 2006]. Some of them are able to handle terrain deformations [Wang et al. 2010]. More various motions have been obtained through the design of elaborated state machines [Coros et al. 2009; de Lasa et al. 2010; Peng et al. 2016, 2017]. All of them assume a gaited pattern governs the motion, which is not true in constrained environments.

Whole-body dynamic solvers have also been proposed to synthesize non-gaited, highly dynamic motions [Al Borno et al. 2012]. Similarly, reactive multi contact controllers follow local control policies using a small planning horizon that results in myopic behaviors [Hämäläinen et al. 2015; Jain et al. 2009]. Recent work try to generalize these controllers by using them within new sampling based algorithms [Borno et al. 2017]. Recently [Naderi et al. 2017] also proposed to combine these approaches with a global planner, thus reintroducing a combinatorial issue hardly acceptable outside discretized climbing scenarios. Their work lies in the continuity of the seminal work of [Bretl et al. 2004] and [Escande et al. 2008], who presented the first global planning approach to address multi contact locomotion, in complex climbing or table egress scenarios. These works used *contact-before-motion* approaches, where the motion is discretized as a sequence of key contact postures connected through a graph structure, at the cost of large computation times (ranging from minutes to hours).

Later [Mordatch et al. 2012] and [Deits and Tedrake 2014] showed that the problem could be addressed as a nonconvex optimization problem, where the trajectory of the robot and the contact locations are optimized simultaneously. They obtained better computation times thanks to the use of a simplifying point mass model to reduce the dimension of the problem. In a previous work we also used a simplified model to plan a contact sequence [Tonneau et al. 2015], similar to the one later proposed by [Grey et al. 2017]. Then [Herzog et al. 2015] and [Carpentier et al. 2016] computed a dynamically feasible trajectory for the COM along that sequence, to finally obtain interactive computation times.

2.2 Data driven approaches and the arrival of learning methods

In parallel to procedural approaches, data driven methods are able to synthesize complex movements by connecting motion clips within a graph structure [Kovar et al. 2002; Lau and Kuffner 2006], before performing motion deformation to adapt the motion to a new environment [Choi et al. 2003; Witkin and Popovic 1995], even for large deformations [Han et al. 2016; Tonneau et al. 2016]. However, these methods only apply to motions that present the same contact pattern as those used in the provided data, and thus similar environments. The introduction of learning techniques such as [Peng et al. 2016] holds the promise of a breakthrough towards the generality of data driven approaches, one of the most impressive examples being proposed by [Holden et al. 2017]. The first learning paper to directly address multi contact locomotion was by [Kang and Lee 2017], but remains limited to a single type of morphology (a human).

Method	Kinematic model	contact model	Dynamic model	computation times (≈ 10 contacts)	avatars	application
[Tonneau et al. 2015] + Ours	Approximated	6D	Quasi-static	few seconds	Any	Any, including highly constrained
[Mordatch et al. 2012]	Joint limits violation	6D	Point mass	Minutes	Any	Any
[Kang and Lee 2017]	Approximated (Learned)	effector slipping	False	Several seconds to Minutes	Humanoid	Climbing
[Naderi et al. 2017]	Exact	3D	Exact	Minutes	Any	Climbing, discretized contact locations

Table 1. Comparison of the main methods in the literature for the complete issue of both planning and generating a multi contact whole-body motion.

2.3 Simplified models for efficient motion generation

The recent successes in multi contact locomotion share the common idea of simplifying the dynamics of the character, which results in a faster solution of the problem. [Kang and Lee 2017] check whether the COM of the robot lies above the convex hull of the contact points to determine whether the robot is in equilibrium. However, this condition is neither necessary nor sufficient in the multi contact case, and is only valid to verify static equilibrium (i.e. when the acceleration of the COM is null) for coplanar contact points [Del Prete et al. 2016]. But, even with an accurate formulation of the dynamics [Qiu et al. 2011], using the point mass model results in a loss of information on the angular momentum generated by the limbs, which can make the character lose equilibrium.

The other important piece of information that is lost with the point mass model are the kinematic limits: since these methods do not consider the joint positions, it is impossible to guarantee that the resulting motion be collision free or even kinematically feasible. [Kang and Lee 2017] tackle this issue by using learning techniques to approximate the set of kinematically feasible configurations when selecting the contact location. Then they use motion deformation to generate the whole-body trajectory. However, the resulting motion can still result in collisions with the environments.

2.4 Situation of our contribution

As of today the point mass model appears as the pragmatic approach to solve multi contact planning problems with reasonable computation times, but it cannot handle the kinematic constraints. Moreover, the dynamics of the system remains nonconvex, and the most efficient methods proposed by Carpentier et al. and Herzog et al. require several hundreds of milliseconds to generate a feasible trajectory, with no guarantee of convergence or collision avoidance.

Our work proposes contributions regarding both issues by introducing accurate kinematic constraints to the point mass model, and by proposing a fast solution to compute a valid COM trajectory. We choose to integrate our method in an open-source multi contact planning framework to demonstrate its interest. Table 1 situates our approach with respect to the main existing ones. Our method aims at compensating for the limitations of the point mass model with a more accurate approximation of the kinematic model than [Mordatch et al. 2012], and focuses on the quasi-static dynamics to nullify the effect of the angular momentum of the bodies. The use of the point mass model combined with our planner presented in [Tonneau et al. 2015] results in a faster and more versatile solver. Finally, with our approach contact slipping and end effector penetration is prevented through the use of a 6D constraint for the contacts.

3 RATIONALE: TRANSITION POINTS IN MULTI CONTACT LOCOMOTION

In this Section we describe the key idea of our approach, and the different steps of our method. We advise the reader to watch the second video attached to this paper, which provides a complementary overview of the approach.

3.1 Locomotion as the discrete issue of computing transition points

While locomotion is a continuous phenomenon, any legged motion can be described by a discrete set of **contact phases**, for which a constant number of contacts are active. Each contact phase differs from the previous one by exactly one contact addition or removal. Key instants of the motion are transitions, where the motion discretely switches from one phase to the other. The transition is subject to specific constraints allowing to safely remove a contact without falling, or add a new one without violating joint limits.

The key idea of this paper is that **in order to synthesize a plausible motion, we only need to compute a position for the COM of the character at each transition instant**. This allows to compute a trajectory for the COM of the character extremely rapidly. Our problem is thus the following: given a set of contact phases as inputs, how to generate a COM trajectory that will result in a plausible whole-body motion?

We illustrate our approach using Figure 2. In the first row, we can see an example of input to our method. A sequence of 3 contact phases is considered. This sequence describes a motion where the character moves forward using its right foot. In phase 1 both feet and the right arm of the character are in contact; in phase 2 the right foot is lifted; in phase 3 the right foot is in contact at a new location.

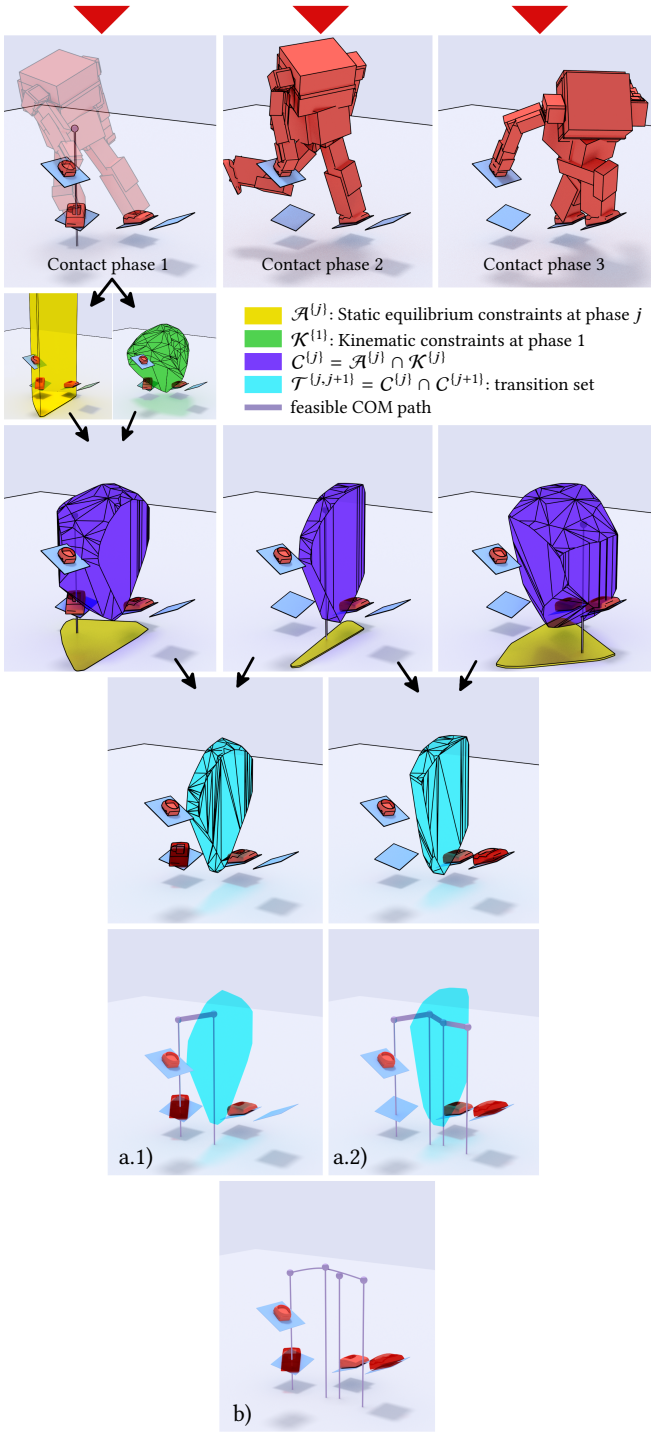

Fig. 2. Method overview. Each colored polytope corresponds to a set constraint on the COM of the character at a specific phase of the motion.

Our method takes as inputs the contact locations at each phase, and the COM position of the character at the beginning (purple sphere), and possibly the end. Given these inputs, our method outputs a trajectory for the COM of the character, later used to generate a whole-body motion.

3.2 Transition constraints for contact phases

To define what a transition point is, we determine how the contact locations uniquely constrain the position of the COM at each phase. These constraints are formulated as 3D objects (polytopes) shown in the second row of Figure 2. Each color refers to a different kind of constraints, yellow for equilibrium (Section 4) and green for kinematic (Section 5) constraints: any COM position included in a polytope respects the constraints associated to it. Since the constraints are independent, the set of COM positions that respects all these constraints is simply the intersection of the green and yellow polytopes, shown in purple in the third row of Figure 2.

The transition instant between two contact phases is further constrained, because at this precise moment the constraints of both contact phases must be respected. The feasible transition set for the COM is thus obtained by intersecting the purple polytopes of both contact phases. This results in the blue polytopes in the fourth row of Figure 2. Any COM position included in the blue polytope is thus a candidate transition point (Section 6).

3.3 Computing a feasible COM trajectory from transition points

From a given COM position in contact phase 1, to reach contact phase 2 we simply need to select a transition point in the associated blue polytope (thus belonging to both phases 1 and 2), and consider the straight line between these points (Figure 2 - a.1)). Because we are dealing with convex objects, in Section 7.1 we show that the straight line corresponds to a feasible trajectory (under the right time parametrization). We can then apply the same procedure to reach phase 3 from the new position, and from there a desired final position (Figure 2 - a.2)).

The resulting trajectory, while feasible, is subject to restrictive equilibrium constraints, which require the character to stop at each transition point. In Section 7.2, we discuss the relaxation of such constraints to generate more dynamic motions (Figure 2 - b)).

4 EQUILIBRIUM CONSTRAINTS

In the remainder of this paper, we say that a COM position or trajectory is **equilibrium feasible** if it respects the laws of dynamics given the active contact constraints. As opposed to us, several approaches such as [Kang and Lee 2017] verify equilibrium feasibility by simply verifying that the COM lies above the convex hull of the contact points projected on $z = 0$. This model is only valid for coplanar, almost flat contacts, and is false in the general case [Del Prete et al. 2016]. An illustration of the limitation of this approach is the case of a vertical or slightly inclined climbing wall; in such a case the convex hull is reduced to a really narrow shape, above which the Center Of Mass cannot stand while avoiding collisions.

The general formulation, for a given configuration with $n_c + 1$ contacts, possibly including grasping contacts, is derived from the Newton-Euler equations, and, unfortunately, takes time to compute [Qiu et al. 2011]:

$$\mathbf{H}(\mu_0, \mu_{n_c}, \mathbf{p}_0, \ldots, \mathbf{p}_{n_c}) \underbrace{\begin{bmatrix} m(\ddot{\mathbf{c}} - \mathbf{g}) \\ m\mathbf{c} \times (\ddot{\mathbf{c}} - \mathbf{g}) + \dot{\mathbf{L}} \end{bmatrix}}_{\mathbf{w}} \leq \mathbf{h}(\mu_0, \mu_{n_c}, \mathbf{p}_0, \ldots, \mathbf{p}_{n_c}) \tag{1}$$

where :

- $\mathbf{c}, \dot{\mathbf{c}}, \ddot{\mathbf{c}} \in \mathbb{R}^3$ are the COM position, velocity and acceleration;
- $m \in \mathbb{R}$ is the mass of the avatar;
- $\mu_i \in \mathbb{R}$ is the friction coefficient at the i-th 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 angular momentum (applied at $\mathbf{c}$).
- $\mathbf{p}_i \in \mathbb{R}^3$ is the position of the i-th contact point.
- $\mathbf{H}$ and $\mathbf{h}$ are respectively a matrix and a vector uniquely defined by the $\mathbf{p}_i$ and μ_i [Del Prete et al. 2016].

All terms are expressed in the world reference frame.

For the moment, we will focus on the quasi-static case, where the acceleration and angular momentum are negligible. Thus $\mathbf{w}$ is simplified to:

$$\mathbf{w} = \begin{bmatrix} -m\mathbf{g} \\ m\mathbf{g} \times -\mathbf{c} \end{bmatrix}$$

This allows us to rewrite (1) as a function of $\mathbf{c}$:

$$\underbrace{m\mathbf{H}_{[3:6,:]}\hat{\mathbf{g}}}_{\mathbf{A}}\mathbf{c} \leq \underbrace{\mathbf{h} + m\mathbf{H}_{[0:3,:]}\mathbf{g}}_{\mathbf{a}} \tag{2}$$

where $\hat{\mathbf{g}}$ is the skew symmetric matrix of $\mathbf{g}$. For each phase j, there exists a convex quasi-static polytope $\mathcal{A}^{\{j\}} : \{\mathbf{c}, \mathbf{A}^{\{j\}}\mathbf{c} \leq \mathbf{a}^{\{j\}}\}$, independent of z, illustrated by the yellow projections in Figure 2.

5 KINEMATIC CONSTRAINTS

Given a set of 6D contact constraints, as well as the collision volumes and joint limits of the avatar, we want to express the space of feasible 3D COM positions **kinematically feasible**. We approximate this space with a polytope formulation, computed offline as follows. For a given effector frame $\mathbf{r}_i$, we sample a large number of whole-body configurations (e.g. 10^6). For those that are collision free, we store the resulting COM position expressed in the frame $\mathbf{r}_i$. We then compute the convex hull $\mathcal{K}_i$ encompassing all these points. This convex hull is a polytope, expressed as a set of linear inequalities:

$${}^{\mathbf{r}_i}\mathbf{c} \in \mathcal{K}_i \Leftrightarrow {}^{\mathbf{r}_i}\mathbf{K}_i {}^{\mathbf{r}_i}\mathbf{c} \leq {}^{\mathbf{r}_i}\mathbf{k}_i \tag{3}$$

with all variables expressed in the effector frame.

Finally, the constraints applying to a set of n_c active contacts are computed under the hypothesis of variable independence (discussed in Section 8.5). This allows us to write

$${}^{W}\mathbf{c} \text{ feasible} \Leftrightarrow \forall i = 0 \ldots n_c, {}^{W}\mathbf{K}_i {}^{W}\mathbf{c} \leq {}^{W}\mathbf{k}_i$$

with all variables being expressed in the world frame W. This is equivalent to stacking all the constraints that define a polytope $\mathcal{K} = \bigcap_{i=0}^{n_c} \mathcal{K}_i$:

$${}^{W}\mathbf{c} \text{ feasible} \Leftrightarrow {}^{W}\mathbf{K} {}^{W}\mathbf{c} \leq {}^{W}\mathbf{k} \tag{4}$$

In the remainder of this paper we omit the ^{W} superscript and assume all variables are expressed in world coordinates. For each phase j, we can thus define a convex kinematic feasibility polytope $\mathcal{K}^{\{j\}} : \{\mathbf{c}, \mathbf{K}^{\{j\}}\mathbf{c} \leq \mathbf{h}^{\{j\}}\}$, illustrated by the green polytope for phase 1 in Figure 2.

6 FINDING A TRANSITION POINT

We consider two consecutive contact phases in a motion, which differ either by a contact creation or a contact removal (Figure 2 - first row). We develop the latter case (Figure 2 - phases 1 and 2), while the first one can be straightforwardly mirrored from the following reasoning.

A state $\mathbf{x}^{\{j\}} = (\mathbf{c}^{\{j\}}, \dot{\mathbf{c}}^{\{j\}}, \ddot{\mathbf{c}}^{\{j\}})$ describes a COM position, velocity and acceleration satisfying the constraints at contact phase j. For the moment, because we are in the quasi-static case, we assume $\dot{\mathbf{c}}^{\{j\}} = \ddot{\mathbf{c}}^{\{j\}} = \mathbf{0}$. We consider the issue of finding a kinematically and equilibrium feasible COM trajectory allowing to go from a state in the first phase to a point in the second phase, $\mathbf{x}^{\{1\}}$ and $\mathbf{x}^{\{2\}}$.

For a quasi-static solution to exist, a necessary condition is the existence of a transition state $\mathbf{x}^{\{1,2\}}$ such that the constraints of both phases are respected at $\mathbf{c}^{\{1,2\}}$. In Figure 2 (fourth row), this is equivalent to proving that the blue polytope $\mathcal{T}^{\{j,j+1\}}$ is not empty.

We observe that the kinematic constraints of phase 1 are included in the kinematic constraints of phase 2, while the equilibrium constraints of phase 2 are included in those of phase 1. Thus, we only need $\mathbf{c}^{\{1,2\}}$ to be kinematically feasible regarding the constraints of the first phase, and equilibrium feasible regarding the constraints of the second phase.

Finding a valid $\mathbf{c}^{\{1,2\}}$ can thus be computed by solving the following linear program:

$$\begin{aligned} \text{find} \quad & \mathbf{c}^{\{1,2\}} \in \mathbb{R}^3 \\ \mathbf{s.t.} \quad & \mathbf{A}^{\{2\}}\mathbf{c}^{\{1,2\}} \leq \mathbf{a}^{\{2\}} \\ & \mathbf{K}^{\{1\}}\mathbf{c}^{\{1,2\}} \leq \mathbf{k}^{\{1\}} \end{aligned} \tag{5}$$

Solving (5) is enough to prove whether a solution exists. However, in our experiments we prefer the following Quadratic Program (QP) formulation:

$$\begin{aligned} \text{find} \quad & \begin{bmatrix} \mathbf{c}^{\{1,2\}} \\ s \end{bmatrix} \in \mathbb{R}^4 \\ \text{minimize} \quad & \|\mathbf{c}^{\{1,2\}} - \mathbf{c}^{\{1\}}\| + w\|s\| \\ \mathbf{s.t.} \quad & \mathbf{A}^{\{2\}}\mathbf{c}^{\{1,2\}} - \mathbf{1}s \leq \mathbf{a}^{\{2\}} \\ & \mathbf{K}^{\{1\}}\mathbf{c}^{\{1,2\}} \leq \mathbf{k}^{\{1\}} \end{aligned} \tag{6}$$

where $w \in \mathbb{R}^+$ is a user defined weighting variable. The introduction of the slack variable s guarantees that the method will always return a solution, guaranteed to be kinematically feasible, even if no static solution exists (in which case $s \leq 0$). This is interesting for computer graphics applications, because it makes it possible to always present a result (with a minimal violation of the constraints if required), rather than simply failing. Using a large w (we set it to 1000) guarantees a static solution will be chosen if it exists.

The introduction of a quadratic cost function allows to select a transition point as close as possible to $\mathbf{c}^{\{1\}}$, resulting in shorter COM trajectories, which we subjectively found more plausible.

7 GENERATING A COM TRAJECTORY

7.1 Quasi-static case

Considering two states $\mathbf{x}^{\{1\}} = (\mathbf{c}^{\{1\}}, \mathbf{0}, \mathbf{0})$ and $\mathbf{x}^{\{2\}} = (\mathbf{c}^{\{2\}}, \mathbf{0}, \mathbf{0})$, as well as a feasible transition state $\mathbf{x}^{\{1,2\}} = (\mathbf{c}^{\{1,2\}}, \mathbf{0}, \mathbf{0})$, we now make the critical observation that **the polyline $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}}\mathbf{c}^{\{2\}}$ is both kinematically and equilibrium feasible** (Figure 2 - a). The proof is straightforward.

Both $\mathbf{c}^{\{1\}}$ and $\mathbf{c}^{\{1,2\}}$ respect the constraints of phase 1. The set of constraints of phase 1 is a convex polytope. By convexity, any point in the segment $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}}$ respects the constraints of phase 1. Therefore, $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}}$ is kinematically and equilibrium feasible. □

The same reasoning applies to $\mathbf{c}^{\{1,2\}}$ and $\mathbf{c}^{\{2\}}$, thus the polyline $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}}\mathbf{c}^{\{2\}}$ is both kinematically and equilibrium feasible. This means that, in the quasi-static case, with a simple call to a low dimensional QP, we are able to compute a feasible trajectory.

We stress the fact that this trajectory does not provide a time parametrization, because of its quasi-static nature: as long as the acceleration remains negligible, the motion is feasible. Formally, if $\mathbf{s}(t), t \in [0,1]$ is a constant-velocity time parametrization of the polyline, $\exists \epsilon(t)$ such that the trajectory $\mathbf{c}(t) = \mathbf{s}(\epsilon(t))$ is equilibrium (and kinematically) feasible. For $\mathbf{c}(t)$ to be feasible, $\epsilon(t)$ must be such that each transition point of the polyline is reached with a velocity of 0. We say that $\mathbf{s}(t)$ is ϵ feasible.

7.2 Relaxing the quasi-static case

The quasi-static solution is valid, but constraining in the sense that it imposes the character to move slowly and stop after each contact transition. To generate smoother trajectories, we thus aim at removing these constraints. We propose to use Bezier curves, because they can approximate arbitrarily well any smooth trajectory, and present a relevant property: **any point in a Bezier curve belongs to the convex hull of its control points**. The transition points will be used as control points for the curve.

7.2.1 Trajectory computation as a Bezier curve. We define a COM trajectory connecting two states $\mathbf{x}^{\{1\}}$ and $\mathbf{x}^{\{l\}}$, (where $l > 1$) as a Bezier curve $\mathbf{c}(t)$ of degree $n \geq 5$:

$$\mathbf{c}(t) = \sum_{k=0}^{n} B_k^n(t)\mathbf{P}_k \tag{7}$$

where the B_k^n are the bernstein polynoms and the $\mathbf{P}_k$ are control points. All the curves that we use comprise at least 6 control points to ensure that the following constraints are verified:

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

We could simplify these equations since the velocities and acceleration are $\mathbf{0}$, but we use this general formulation because this constraint will be removed in Section 7.3.1.

7.2.2 One or two transition points as control points. In this first case, we relax the quasi-static constraint to generate smooth trajectories, while we guarantee to preserve kinematic feasibility. We consider an additional control point to our nominal Bezier curve:

$$\mathbf{c}(t) = \sum_{k \in \{0,1,2,4,5,6\}} B_k^6(t)\mathbf{P}_k + B_3^6(t)\mathbf{c}^{\{1,2\}} \tag{8}$$

This curve use a transition point as control point to "attract" the trajectory towards it, and as such, is an approximation of the polyline $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}}\mathbf{c}^{\{2\}}$. In the remainder, we make the distinction between a **transition point** and an **attractor point**: a transition point is a point reached by the COM trajectory, while an attractor point is a transition point that is never reached by the COM trajectory, but "attracts" the trajectory in its neighbourhood.

Let us assume that a contact is removed from phase 1 to phase 2, as in Figure 2 (a mirror reasoning applies for the contact added between phases 2 and 3). In this case, all the control points of $\mathbf{c}(t)$ belong to the kinematic polytope $\mathcal{K}^2$. By definition of a Bezier curve, $\mathbf{c}(t)$ is entirely included in $\mathcal{K}^2$, and thus respects the kinematic constraints of phase 2. Moreover, a portion of $\mathbf{c}(t)$ is kinematically feasible for phase 1 (because at least $\mathbf{c}^{\{1\}}$ respects this constraint). Therefore, there exists a contact transition time that makes $\mathbf{c}(t)$ kinematically feasible, under the assumption that our approximated kinematic constraints are exact.

With the same reasoning, it is possible to generate a smoother Bezier curve with the same guarantees for three consecutive phases, if and only if a contact is removed in phase 1, and another is added in phase 3:

$$\mathbf{c}(t) = \sum_{k \in \{0,1,2,5,6,7\}} B_k^7(t)\mathbf{P}_i + B_3^7(t)\mathbf{c}^{\{1,2\}} + B_4^7(t)\mathbf{c}^{\{2,3\}} \tag{9}$$

7.2.3 General case. To generate even smoother trajectories, we can consider using attractor points for $m+1$ consecutive phases:

$$\mathbf{c}(t) = B_0^m(t)\mathbf{c}^{\{1\}} + \sum_{i=1}^{m-1} B_i^m(t)\mathbf{c}^{\{i,i+1\}} + B_m^m(t)\mathbf{c}^{\{m\}} \tag{10}$$

In this general case we lose all theoretical guarantees of feasibility. However, we can iteratively construct a curve that ultimately converges to a feasible trajectory.

Let $\mathbf{s}(t), t \in [0,1]$ be a ϵ feasible parametrization of the polyline $\mathbf{c}^{\{1\}}\mathbf{c}^{\{1,2\}} \dots \mathbf{c}^{\{m-1,m\}}\mathbf{c}^{\{m\}}$. The curve

$$\mathbf{c}(t) = \sum_{k=0}^{n} s\left(\frac{k}{n}\right) B_k^n(t) \tag{11}$$

converges towards $\mathbf{s}(t)$ as n increases, according to the Weierstrass Approximation Theorem [Weierstrass 1885].

It is thus possible to generate a curve of arbitrary degree n, and increase n until the required constraints are satisfied, with the guarantee to converge towards the original feasible solution. The attractor points thus drive the COM trajectory towards the quasi-static solution, without exactly reaching it.

7.3 The 2 Points Attractors for COM trajectories method (2PAC)

After presenting the theory behind our method, we propose a concrete implementation, which is the one we used to generate our results. In summary, we use a maximum number of 2 attractor points for generating Bezier curves to ensure kinematic feasibility, and we use a sliding window to connect consecutive curves, so as to avoid stopping along the motion.

Generating a COM trajectory over a window of 2 (respectively 3) contact phases results in a Bezier curve comprising 1 (respectively 2) attractor point(s), with the guarantee that the computed trajectory is always kinematically feasible. Considering simultaneously more than 3 contact phases requires to introduce an iterative approach to progressively introduce new attractor points until a feasible trajectory is obtained. However, as more contact phases are considered, the resulting trajectory becomes smoother. Because we ultimately target real time applications, we choose a maximum window size of 3 contact phases for trajectory generation.

7.3.1 Using a sliding window to generate a smooth trajectory over many contact phases. Each trajectory $\mathbf{c}(t)$ is created such that the terminal velocity $\dot{\mathbf{c}}(1)$ is null. This results in the character stopping completely after each contact creation, a limitation shared with other methods such as [Naderi et al. 2017].

However in our case, we can easily remove this constraint thanks to the connectivity property of Bezier curves, using a sliding window. Let us consider again the example from Figure 2, extended with a fourth contact phase where the hand contact is removed. We thus look for a COM trajectory connecting a state $\mathbf{x}^{\{1\}}$ to a state $\mathbf{x}^{\{4\}}$, given $\{\mathbf{x}^{\{j\}}, j = 1 \dots 4\}$. In this case we use the following scheme:

- Compute a trajectory $\mathbf{c}_1(t)$ for phases 1 to 3 that connects $\mathbf{x}^{\{1\}}$ to $\mathbf{x}^{\{3\}}$;
- Choose $u \in [0,1]$ such that $\mathbf{c}_1(u) \in \mathcal{K}^{\{3\}}$ and $\dot{\mathbf{c}}_1(u) \neq \mathbf{0}$, and define $\mathbf{x}_*^{\{3\}} = (\mathbf{c}_1(u), \dot{\mathbf{c}}_1(u), \ddot{\mathbf{c}}_1(u))$;
- Compute a trajectory $\mathbf{c}_2(t)$ for phases 3 to 4 that connects $\mathbf{x}_*^{\{3\}}$ to $\mathbf{x}^{\{4\}}$;
- The concatenation of the curves is a kinematically feasible Bezier curve, by construction of the state $\mathbf{x}_*^{\{3\}}$.

We can iteratively repeat this process for arbitrary long contact sequences: generate a kinematically feasible trajectory for the first 2 or 3 contact phases; select a point in the trajectory that respects the kinematic constraints of the last contact phase reached; generate from this point a kinematically feasible trajectory for the 2 or 3 next contact phases, and concatenate it with the previous one; repeat until the last contact phase is reached.

7.3.2 Time parametrization of the trajectory. The final trajectory that we obtain is a Bezier curve defined on an interval $[0,1]$. In our method, rather than computing a parametrization $\epsilon(t)$ such that the trajectory becomes feasible, we empirically compute $\epsilon(t)$ by scaling the time t by a weighted distance between the start and goal whole-body configuration. Despite this, most of the computed trajectories are also dynamically feasible (see Section 8.5).

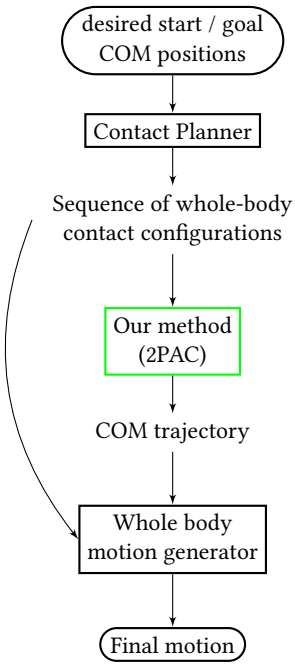

Fig. 3. Workflow of our demonstrator. The rounded rectangles are the inputs and outputs of the framework. The unbounded text refer to internal inputs / outputs used by the three main blocks (sharp rectangles). The whole body motion generator considers as inputs the COM trajectory, but also the initial and goal whole body configurations of the character.

8 APPLICATION: GENERATION OF MULTI CONTACT MOTIONS

To be able to compute the COM trajectories, our method requires as input a list of states $\mathbf{x}^{\{j\}}$. Thus, to demonstrate our method, we integrate it within a framework for multi contact locomotion.

Figure 3 illustrates the workflow.

8.1 Workflow description

To generate the states, we use an open source procedural contact-planner [Tonneau et al. 2015]. Given a desired start and goal COM positions, the contact planner is able to compute a discrete sequence of whole-body contact configurations that describe the motion, such that at most one contact is broken and another contact is added between two consecutive configurations.

We consider two uses cases for the planner:

- **case 1: one contact planning**. In this interactive context we are only interested in computing the next step. The goal configuration is unknown. A random process (or a user) selects a direction of locomotion, and a target contact phase reachable from the current phase in that direction. The planner then generates a collision free, whole-body configuration in static equilibrium that meets the constraints;
- **case 2: contact sequence planning**. The planner computes a sequence of contact phases and configurations allowing to reach a user defined goal configuration.

In the first case, the environment (or another character) geometry can be discretely updated between each step, while in the second case the contact sequence is planned on a static environment. In both cases, our method then generates a COM trajectory, used by a dedicated motion generator to synthesize the final motion.

8.2 Whole body motion generation

To generate a whole-body motion given a user defined time parametrization of the COM trajectory and contact constraints, we use a generalized inverse kinematics solver combined with a sampling based approach to avoid collisions. The method first computes a whole-body motion that follows the given COM trajectory and satisfies the contact constraints [Baerlocher and Boulic 2004], but this motion can result in collisions between the character and the environment. In such case, a sampling based approach is used to compute a collision free motion. We consider two types of collisions.

If a limb is in collision, a constraint-based RRT planner is used on the joints of the limb to look for a collision free motion for the limb, while preserving the contact and COM constraints. This solver is a feature of our motion planner, and its details can be found in [Mirabel et al. 2016].

Otherwise, if the root is in collision at an instant t, with a penetration greater than a user defined value (we use 5cm in our experiments), we assume that the computed COM trajectory is not feasible (If the penetration is lesser than the threshold, we assume that moving the free limbs is sufficient to avoid the collision). In such cases, we replace the transition point $\mathbf{c}_{close}$ that is the closest to the first collision point in the trajectory. To do so we consider the first obstacle in collision with the root, and its surface normal $\mathbf{n}_s$. $\mathbf{n}_s$ gives a repulsion direction. A new transition point $\mathbf{c}_{new}$ is computed as $\mathbf{c}_{new} = \mathbf{c}_{close} + \alpha\mathbf{n}_s$, where α is given by the penetration distance between the root and the obstacle. The transition points locations are updated iteratively until either a collision free solution for the root is found, or a computed $\mathbf{c}_{new}$ no longer respects its phase constraints.

In both cases, we use a user-defined time out value after which the best motion found is returned. It thus appears that when the root is in collision with the environment, our collision avoidance algorithm is not complete, because the approach is local. However, in our experiments this rarely occurred, because the contact planner guarantees that the segment connecting the root poses of two successive key postures is collision free. However, we cannot guarantee that a collision free solution be found if it exists. This is a limitation shared with all existing multi contact approaches, although our experiments demonstrate that our method is able to handle severely cluttered cases.

8.3 Scenarios

The companion video 1 shows the versatility of our method, in the variety of situations it addresses. We use three characters: a humanoid, a quadruped, and a six-limbed insect (Figure 1).

Our main scenario is interactive and matches the first use case. The humanoid and the quadruped play a variant of a game of Twister (http://cpc.cx/k5j): at each turn, an effector is selected to either create a contact or break it (Figure 1 - Left). The contact planner computes a state that satisfies the new constraint, and our method computes the motion to reach the state, by taking into account the bodies of the other character. If no such solution is found, a new effector is randomly selected. This scenario illustrates the ability of our method to compute motions in close interaction while avoiding collisions.

Another scenario shows a humanoid avatar sitting or standing on a chair, in two cluttered environments (Figure 1 - Middle). The scenario demonstrates the strong advantage of combining a contact planner with our method: variations of the same motion can be obtained automatically and almost instantly to adapt to new scenes, in a way different from classical motion editing approaches, because the contact phases differ between the different iterations.

In a third scenario, the humanoid or the insect climb along a set of cubic obstacles (Figure 1 - Right). The same scenario is addressed for both creatures without requiring specific heuristics, tuning, or discretization of the environment.

Still, contrary to other existing procedural methods, our approach leaves room for artistic authoring of the motions, as demonstrated in our fourth scenario. In this scenario we demonstrate 3 different motions obtained between the same two contacts phases. The variation is obtained through the manual displacement of the transition points by a user along the z axis. Since they attract the COM trajectory towards them, the transition points are an intuitive tool to control the trajectory. Furthermore, the 3D representation of the admissible positions for a transition point allows the user to easily determine where to put such points.

8.4 Time performance

In this Section we present the relevant computation times obtained with our framework, which we then discuss.

8.4.1 *Quantitative results.*

Computation time of the COM trajectory. Computing a COM trajectory over 3 contact phases takes (that is roughly 1 second of motion) **less than 10ms in the worst case** with a python implementation. This time is mostly spent to compute the equilibrium constraints (2 ms per contact phase in average) and to find the 2 transition points (0.5ms per point in average).

Computation time for the complete framework. Regarding Figure 3, the other time consuming operations are the computation of an input contact sequence, and the generation of the whole-body motion from the COM trajectory. In all the scenarios presented, the planning time for the contact sequence is less than a second in the worst case. The actual generation of the whole-body motion is extremely dependent on the complexity of the environment, from the standing up / sitting down scenarios (**100ms for one second of animation on average, including planning**) to the twister scenario (**2s for one second on average**). The total time to generate a motion comprising 10 contact creations thus never exceeds a few seconds, with an average time of about one second.

8.4.2 *Comparison with the state of the art.*

Computation time of the COM trajectory. Regarding the time to compute the COM trajectory, our method compares favorably to the fastest existing approaches [Carpentier et al. 2016; Herzog et al.

2015], which take on average several hundreds of ms without guarantee of convergence, which makes them unsuited for a robust real time graphical application, contrary to our approach.

As we said, most of the time of our method is spent in the computation of the equilibrium constraints. In Section 4 we explain that in the general case the convex hull of the contact points is a poor approximation of the equilibrium constraints. However, conducting a user study to determine in which scenarios this approximation remains plausible might be relevant in the future, as it would allow to divide by ten the computation times in such cases.

Computation time for the complete framework. Comparing our results with other approaches is not trivial, because the inputs and outputs differ between each method. For instance [Mordatch et al. 2012] compute a COM trajectory simultaneously with the contact locations, but the inverse kinematics is generated afterwards and does not consider collisions and joint limits; [Naderi et al. 2017] address the complete problem, but consider a discretized set of contact points as inputs. For these reasons and because detailed computation times are not always available, we cannot provide an extensive comparison of the performance of the methods. Still, we can compare the order of magnitudes of the computation times obtained with respect to the literature. Table 1 suggests that our framework performs better, or at least not worse, with respect to the literature, while we address severely cluttered scenarios that have not been demonstrated by the other methods. This comparison shows that 2PAC can be integrated with existing animation frameworks to obtain a complete solution faster than existing approaches.

8.5 Accuracy of the method

Kinematic accuracy. A limitation of our approach lies in the hypothesis of independence (used to compute the kinematic feasibility polytope as the intersection of the kinematic constraints associated with each active contact). As a result the kinematic constraint is not exact, but only approximated. However, in our experiments the kinematic constraint was verified in a large majority of cases. In the most cluttered scenario (i.e. the Twister), we have let the characters perform a random walk over 1000 steps, and counted the number of times the whole-body projection to a transition point failed (that is, the number of false positives induced by our approximation). For the quadruped character, the constraint was always accurate. For the humanoid character, which presents more restricted joint limits, the constraint was accurate 93.7 % of the times. In conclusion, our kinematic feasibility constraint is rather reliable in our context.

Dynamic accuracy. Our equilibrium constraint is exact in the quasi-static case, and a feasible solution will always be found if it exists. However the more dynamic the desired motion is, the less likely the trajectory is dynamically feasible. Still, in the scenarios we tested, for a given relaxed trajectory, on average 83.7 % of that trajectory turns out to be dynamically feasible. This high success rate can be intuitively explained by the fact that the beginning and end of our trajectories are always feasible in the quasi-static sense. Because the computed trajectories rarely lie on the frontier of the equilibrium constraints, a non zero acceleration that is not too important may remain feasible. Then, a part of the trajectory may or may not be unfeasible around the phase transition instant, for instance if the kinematic constraints forces to break a contact before the equilibrium constraint of the next contact phase can be satisfied. This relatively small part of the motion is most likely the part who will not be feasible. This success rate allows us to justify the approach, because most of the resulting trajectory is feasible for the avatar, which makes the resulting motions more plausible.

9 DISCUSSION AND FUTURE WORK

In this paper we presented 2PAC, a method for computing COM trajectories for legged characters in multi contact situations, combined with an existing contact planning approach. This method provides a fully automated framework for synthesizing multi contact movements for arbitrary legged characters and environments, as demonstrated by our results. In this Section we discuss the immediate implications of our method, as well as the perspectives it opens it terms of physics-based computer animation.

A procedural approach that could allow artistic control

As of today, few physics-based methods are used for character animation in the industry, for two main reasons (in our opinion). Firstly, these methods are often complex and hard to integrate within existing frameworks, and sometimes computationally expensive. Secondly, they do not provide flexibility to the artist, who often loses control of the animation. 2PAC is fast and simple to integrate within existing animation systems (upon publication, we plan to deliver the source code of our method). Indeed, although we demonstrate a fully automatic approach, it could straightforwardly be used in coordination with motion capture based or artist edited animations systems. In such a case the whole-body generator would solve a constrained optimization problem to limit the deformation of the reference motion, for instance as in [Tonneau et al. 2016].

More importantly, 2PAC also provides flexibility. Firstly because editing the results is easy through the manipulation of the transition points. The 3D representation of the constraints offers an easy way for the artist to select the transitions points and time parametrization, to obtain a subjectively better motion. Secondly, because we control the COM rather than the whole-body pose, the user can easily specify additional pose constraints on the motion. In such cases where trial and error is required, the rapidity of our method is a strong asset. The efficiency of our method thus makes it interesting for both online and offline motion synthesis applications.

A step towards accurate centroidal approaches

As all existing methods using the point mass model, our method makes approximations on the constraints that govern the whole-body motion of a character. Compared with these approaches our results however demonstrate that 2PAC better approximates these constraints, while obtaining better computation times. Regarding the equilibrium of the character, we have shown that first restricting the motion to a quasi-static case before relaxing it already allows us to address a large variety of problems. One exciting direction of improvement is to use the computed trajectories to warm start existing optimal control solvers [Carpentier et al. 2016; Herzog et al. 2015], with the hope that they would converge much faster to a dynamically accurate solution.

Regarding the improvement of our kinematic constraints formulation, using learning approaches such as [Kang and Lee 2017] is a promising direction for future work, although it would require to learn the dependencies between the different limbs of the character.

A new approach for computing COM trajectories

The computation of a COM (or a whole-body) trajectory is usually treated as a nonconvex optimization problem by procedural approaches. Handling the discontinuities induced by the contact phases in this continuous formulation in particular is one of the bottlenecks of the problem. Our take is completely different from this classic approach, in the sense that our formulation of the problem is discrete, since we only focus on transitions. Our work shows that a quasi-static legged motion can be entirely defined by characterizing these discrete transition instants. We strongly believe that better characterizing the constraints active at the transition phases could result in a generalization of our approach to arbitrary legged locomotion, and hope to inspire more research in this direction.

REFERENCES

Mazen Al Borno, Martin de Lasa, and Aaron Hertzmann. 2012. Trajectory Optimization for Full-Body Movements with Complex Contacts. *IEEE transactions on visualization and computer graphics* (Dec. 2012), 1–11. https://doi.org/3A697C56-7DB9-4DDB-9C30-6DCA693F48F9

Paolo Baerlocher and Ronan Boulic. 2004. An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer* 20, 6 (June 2004). https://doi.org/10.1007/s00371-004-0244-4

Mazen Al Borno, Michiel Van De Panne, and Eugene Fiume. 2017. Domain of Attraction Expansion for Physics-Based Character Control. *ACM Trans. Graph.* 36, 2, Article 17 (March 2017), 11 pages. https://doi.org/10.1145/3009907

Timothy Bretl, Stephen Rock, Jean-Claude Latombe, Brett Kennedy, and Hrand Aghazarian. 2004. Free-Climbing with a Multi-Use Robot.. In *ISER* (2008-09-22) *(Springer Tracts in Advanced Robot.)*, Marcelo H. Ang Jr. and Oussama Khatib (Eds.), Vol. 21. Springer, 449–458.

Justin Carpentier, Steve Tonneau, Maximilien Naveau, Olivier Stasse, and Nicolas Mansard. 2016. A versatile and efficient pattern generator for generalized legged locomotion. In *Proc. of IEEE Int. Conf. on Robot. and Auto (ICRA)*. Stockholm, Sweden.

Min Gyu Choi, Jehee Lee, and Sung Yong Shin. 2003. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Trans. on Graphics* 22, 2 (2003), 182–203. https://doi.org/10.1145/636886.636889

Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. 2009. Robust Task-based Control Policies for Physics-based Characters. *ACM Trans. Graph.* 28, 5, Article 170 (Dec. 2009), 9 pages. https://doi.org/10.1145/1618452.1618516

Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. 2010. Feature-Based Locomotion Controllers. *ACM Trans. on Graphics* 29, 3 (2010).

Robin Deits and Russ Tedrake. 2014. Footstep planning on uneven terrain with mixed-integer convex optimization. In *Humanoid Robots (Humanoids), 14th IEEE-RAS Int. Conf. on*. Madrid, Spain.

Andrea Del Prete, Steve Tonneau, and Nicolas Mansard. 2016. Fast Algorithms to Test Robust Static Equilibrium for Legged Robots. In *Proc. of IEEE Int. Conf. on Robot. and Auto (ICRA)*. Stockholm, Sweden.

Adrien Escande, Abderrahmane Kheddar, Sylvain Miossec, and Sylvain Garsault. 2008. Planning Support Contact-Points for Acyclic Motions and Experiments on HRP-2. In *ISER (Springer Tracts in Advanced Robot.)*, Oussama Khatib, Vijay Kumar, and George J Pappas (Eds.), Vol. 54. Springer, 293–302.

Michael X. Grey, Aaron D. Ames, and C. Karen Liu. 2017. Footstep and Motion Planning in Semi-unstructured Environments Using Randomized Possibility Graphs.

Perttu Hämäläinen, Joose Rajamäki, and C. Karen Liu. 2015. Online Control of Simulated Humanoids Using Particle Belief Propagation. *ACM Trans. Graph.* 34, 4, Article 81 (July 2015), 81:1–81:13 pages.

Daseong Han, Haegwang Eom, Junyong Noh, and Joseph S. Shin formerly Sung Yong Shin. 2016. Data-guided Model Predictive Control Based on Smoothed Contact Dynamics. *Comput. Graph. Forum* 35, 2 (May 2016), 533–543. https://doi.org/10.1111/cgf.12853

Alexander Herzog, Nicholas Rotella, Stefan Schaal, and Ludovic Righetti. 2015. Trajectory generation for multi-contact momentum-control. In *Humanoid Robots (Humanoids), 15h IEEE-RAS Int. Conf. on*.

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned Neural Networks for Character Control. *ACM Trans. Graph.* 36, 4, Article 42 (July 2017), 13 pages. https://doi.org/10.1145/3072959.3073663

Sumit Jain, Yuting Ye, and C. Karen Liu. 2009. Optimization-Based Interactive Motion Synthesis. *ACM Transaction on Graphics* 28, 1 (2009), 1–10. https://doi.org/10.1145/1477926.1477936

S Kajita, F Kanehiro, K Kaneko, K Fujiwara, K Harada, K Yokoi, and H Hirukawa. 2003. Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point. In *Proc. of IEEE Int. Conf. Robot. and Auto (ICRA)*. Taipei, Taiwan.

Changgu Kang and Sung-Hee Lee. 2017. Multi-Contact Locomotion Using a Contact Graph with Feasibility Predictors. *ACM Trans. Graph.* 36, 2, Article 22 (April 2017), 14 pages. https://doi.org/10.1145/2983619

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. 2002. Motion graphs. In *ACM Trans. Graph.*, Vol. 21. 473–482.

Manfred Lau and James J. Kuffner. 2006. Precomputed search trees: planning for interactive goal-driven animation. In *Symposium on Computer Animation*. 299–308.

J. Mirabel, S. Tonneau, P. Fernbach, A. K. Seppälä, M. Campana, N. Mansard, and F. Lamiraux. 2016. HPP: A new software for constrained motion planning. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 383–389. https://doi.org/10.1109/IROS.2016.7759083

Igor Mordatch, Emanuel Todorov, and Zoran Popović. 2012. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. on Graph.* 31, 4 (2012), 43:1–43:8.

Kourosh Naderi, Joose Rajamäki, and Perttu Hämäläinen. 2017. Discovering and Synthesizing Humanoid Climbing Movements. *ACM Trans. Graph.* 36, 4, Article 43 (July 2017), 11 pages. https://doi.org/10.1145/3072959.3073707

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. 2016. Terrain-Adaptive Locomotion Skills Using Deep Reinforcement Learning. *ACM Transactions on Graphics (Proc. SIGGRAPH 2016)* 35, 4 (2016).

Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel van de Panne. 2017. DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. *ACM Transactions on Graphics (Proc. SIGGRAPH 2017)* 36, 4 (2017).

Nicolas Perrin, Olivier Stasse, Leo Baudouin, Florent Lamiraux, and Eiichi Yoshida. 2012. Fast Humanoid Robot Collision-Free Footstep Planning Using Swept Volume Approximations. *IEEE Trans. Robotics* 28, 2 (2012), 427–439. https://doi.org/10.1109/TRO.2011.2172152

Jerry Pratt, J. Carff, S. Drakunov, and Ambarish Goswami. 2006. Capture Point: A Step toward Humanoid Push Recovery. *2006 6th IEEE-RAS International Conference on Humanoid Robots* (2006). https://doi.org/10.1109/ICHR.2006.321385

Zhapeng Qiu, Adrien Escande, Alain Micaelli, and Thomas Robert. 2011. Human motions analysis and simulation based on a general criterion of stability. In *Int. Symposium on Digital Human Modeling*.

Steve Tonneau, Rami Ali Al-Ashqar, Julien Pettré, Taku Komura, and Nicolas Mansard. 2016. Character contact re-positioning under large environment deformation. *Computer Graphics Forum (Proc. Eurographics)* (2016).

Steve Tonneau, Nicolas Mansard, Chonhyon Park, Dinesh Manocha, Franck Multon, and Julien Pettré. 2015. A reachability-based planner for sequences of acyclic contacts in cluttered environments. In *Int. Symp. Robotics Research (ISRR)*. Sestri Levante, Italy.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. 2010. Optimizing Walking Controllers for Uncertain Inputs and Environments. *ACM Trans. Graph.* 29, 4, Article 73 (July 2010), 8 pages. https://doi.org/10.1145/1778765.1778810

Karl Weierstrass. 1885. Über die analytische Darstellbarkeit sogenannter willkürlicher Functionen einer reellen Veränderlichen. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin* 2 (1885), 633–639.

Andrew Witkin and Zoran Popovic. 1995. Motion Warping. In *Proceedings of the 22Nd Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, New York, NY, USA, 105–108. https://doi.org/10.1145/218380.218422

KangKang Yin, Kevin Loken, and Michiel van de Panne. 2007. SIMBICON: Simple Biped Locomotion Control. *ACM Trans. on Graphics* 26, 3 (2007), Article 105.

Received September 2017