A Two-Stage Trajectory Optimization Strategy for Articulated Bodies with Unscheduled Contact Sequences

Tobia Marcucci[1,3], Marco Gabiccini[1,2,3] and Alessio Artoni[2]

***Abstract*— In this paper, we propose a two-stage strategy for optimal control problems of robotic mechanical systems that proves to be more robust, and yet more efficient, than straight-forward solution strategies. Specifically, we focus on a simplified humanoid model, represented as a two-dimensional articulated serial chain of rigid bodies, in the tasks of getting up (sitting down) from (to) the supine and prone postures. Interactions with the environment are integral parts of these motions, and *a priori unscheduled* contact sequences are discovered by the solver itself, opportunistically making or breaking contacts with the ground through feet, knees, hips, elbows, and hands. The present investigation analyzes the effects on the computational performance of: (i) the explicit introduction of contact forces among the optimization variables, (ii) the substitution of undesired contact forces with geometric constraints that prevent interpenetrations, and (iii) the splitting of the planning problem into two consecutive phases of increasing complexity. To the best of our knowledge, these tests represent the only quantitative analysis of the performances achievable with different solution strategies for optimization-based, whole-body dynamic motion planning in the presence of contacts.**

***Index Terms*— Humanoids, whole-body planning, multi-contact planning, optimization.**

I. INTRODUCTION

While successful approaches to dynamic legged locomotion and whole-body motion planning exist, task-oriented strategies that can handle the variety of situations where body parts can come into contact with the environment through *a priori unscheduled* contact sequences have appeared only very recently in robotics [1] [2] [3].

Interestingly enough, most of the approaches that have proved to be successful in this context are *optimization-based* methods, often denoted as *direct trajectory optimization* or *direct transcription methods*. Such methods are well established in the numerical optimal control community [4], [5], with the most efficient variants [6] being the *direct multiple shooting* [7] and the *direct collocation* [8] methods, where parameterized functions with local support are utilized for both states and inputs.

These methods have been successfully applied in the context of locomotion planning for 2D legged robots and 2D grasp synthesis in [1], where a velocity-based time-stepping scheme is adopted along with a Linear Complementarity Problem (LCP) formulation of contact events. A similar approach has been adopted in [9] to synthesize optimal gaits for a planar two-legged robot with series elastic actuators. Here, the introduction of a higher-order integration scheme for states that are continuous through collisions is the main novelty. In the context of autonomous manipulation, in [10] a framework is devised, based on two different contact models, that allows to synthesize both dexterous and environment-exploiting behaviors in a unified way with encouraging performances.

Other contributions [2] have been directed towards rendering such approaches amenable to whole-body motion planning of 3D robot models by considering the full robot kinematics and condensing the dynamics to the robot's centroidal linear and angular dynamics with impressive outcomes. Along these lines, it is worth citing the contact-invariant approach, originally proposed in [3] to discover complex behaviors for humanoid figures and then extended in [11] to the context of manipulation. Here, acceptable simplifying assumptions (e.g., massless limbs) along with a witty relaxation of the complementarity conditions (to allow smooth gradient calculations for contact forces) are paired to a massive use of inverse dynamics, which may, however, lead to inconsistent results for underactuated and/or defective systems. Such trajectory optimization method has been recently employed to gradually train a neural network by following an Alternating Direction Method of Multipliers (ADMM) strategy, with interesting results [12].

To the best of our knowledge, [13] is the only work we are aware of where a quantitative assessment of the performances of direct transcription methods in the context of robotic systems is conducted. The investigation therein includes criteria such as computational time, quality of the

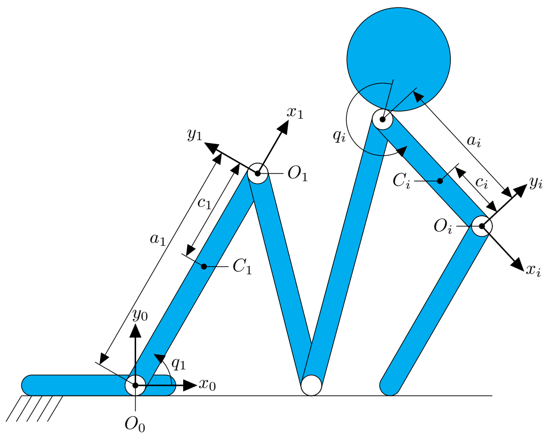

Fig. 1. Humanoid model (all joints actuated).

[1]Research Center "E. Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy
[2] Dipartimento di Ingegneria Civile e Industriale, Largo Lucio Lazzarino 1, 56122 Pisa, Italy
[3] Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy
Corresponding author is: M. Gabiccini, Largo Lucio Lazzarino 1, 56122 Pisa, Italy. Tel. +39-050-221.80.77, Fax: +39-050-221.80.65, Email: `m.gabiccini@ing.unipi.it`.

solutions, and sensitivity to open parameters. The feasibility of applying direct transcription methods for online motion planning of a real ballbot robot is evaluated as a benchmark.

In this paper, we propose a two-stage strategy for optimal control problems of robotic mechanical systems that proves to be more robust, and at the same time more efficient, than straightforward solution strategies. The goal consists of performing various whole-body dynamic tasks (rising from the ground from the supine and prone positions and laying down from the standing position), opportunistically making or breaking contacts with the ground through feet, knees, hips, elbows, and hands. A priori unscheduled contact sequences and the dynamic nature of articulated-body systems make trajectory optimization quite a challenging problem. To the best of our knowledge, these tests represent the only quantitative analysis of the performances achievable with different solution strategies for optimization-based, whole-body dynamic motion planning in the presence of contacts.

II. OPTIMAL CONTROL PROBLEM FORMULATION

A. Description of the Humanoid Robot Model

In order to evaluate the performances of planning with different solution schemes, we employed as a benchmark system the schematic humanoid robot whose model is depicted in Fig. 1. It consists of a 2D serial chain of rigid links connected through revolute joints, whose kinematics is constrained so that the foot link is constantly steady and in contact with the ground (reducing the DoFs of the robot from 8 to 5). This assumption, materialized by fixing the position of the ankle joint center O_0, is rendered consistent by imposing static equilibrium conditions on the foot (in unilateral frictional contact with the ground) through appropriate bound constraints involving the input ankle torque and the ground reactions (see Sec. II-E). This aspect partially restores the difficulty of planning with a floating base since, in the configurations where the vertical load on the foot is very low, the upper bound on (the absolute value of) the ankle torque is practically zero (in this sense, the system can be defined underactuated). Especially in such situations, the fact that the robot can make contact with the ground through its revolute joints O_i becomes a crucial aspect, and (unscheduled) contact sequences emerge as an essential feature of the planned motion.

The lengths, masses, positions of the centers of mass, and centroidal moments of inertia of the humanoid's links are reported in Table I. These are adapted from [14] to an adult man of 73 kg of mass and 1.74 m of height.

TABLE I
HUMANOID PARAMETERS

Link number i	a_i (m)	c_i (m)	m_i (kg)	i_i (kg m^2)
1 (lower leg)	0.473	0.208	6.32	0.0855
2 (upper leg)	0.422	0.173	20.7	0.399
3 (trunk + head)	0.532	0.179	36.8	1.92
4 (upper arm)	0.282	0.119	3.96	0.0227
5 (forearm + hand)	0.355	0.174	3.26	0.0434

B. The CasADi Framework and Expression Graphs

We implemented our problem in the `CasADi` framework [15], which provides building blocks to efficiently formulate and solve large-scale optimization problems, namely sparsity handling, automatic differentiation (AD), and state-of-the-art solvers for nonlinear programming (the interior-point solver IPOPT [16] with the linear solver MA57 [17] from the HSL library [18] was our final choice).

In the `CasADi` framework, symbolic expressions for the objective function and the constraints are formed by applying overloaded mathematical operators to symbolic primitives. These expressions are represented in memory as computational graphs, in contrast to tree representations common to computer algebra systems [19]. Forward and backward source-code transforming AD can be performed at will, such that derivatives of arbitrary order can be computed. The sparsity pattern of the constraint Jacobian is computed using hierarchical seeding [20], and its unidirectional graph coloring is used to obtain the Jacobian with a reduced number of AD sweeps [21].

In order to maximize efficiency of the code, the graph representation of a given expression should be in its minimal form. As an example, if one considers the expression $x + y + \sin(x + y)$, the graph associated with its one-line implementation is composed of 4 nodes: 3 sums and 1 sine. On the other hand, by defining $z = x + y$ one avoids the unnecessary second evaluation of $x + y$, and the nodes of $z + \sin(z)$ are reduced to 3. As expressions grow larger, like those representing the dynamic equations of articulated-body systems, optimizing the associated graph representation can bear huge memory savings and a dramatic reduction in evaluation time.

C. Formulation of the System Dynamics

The equations of motion (EoMs) of the system shown in Fig. 1 are obtained by employing the Euler-Lagrange equations for holonomic systems

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = Q(q,\dot{q},u) \tag{1}$$

where $q \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ collect, respectively, the configuration angles and the input torques at the revolute joints, $B(q) \in \mathbb{R}^{n\times n}$ is the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{n\times n}$ is the matrix of Colioris and centrifugal terms, $G(q) \in \mathbb{R}^n$ and $Q(q,\dot{q},u)$ are the vectors of gravitational and generalized forces, respectively. In this case, the j-th component of Q can be written as

$$Q_j(q,\dot{q},u) = \sum_{i\in\mathscr{F}} J_{v_i}(q)[:,j]^\top f_{c_i}(q,\dot{q}) + u_j \tag{2}$$

where $f_{c_i} = [f_{t_i}\ f_{n_i}]^\top$ are the tangential and normal ground contact forces, J_{v_i} is the linear velocity Jacobian of the i-th contact point O_i, u_j the input torque applied at the j-th joint, and, in order to reduce the computational burden associated with undesirable contacts, we define $\mathscr{F} \subseteq \{1,2,\ldots,n\}$ as the (sub)set of the active contact forces.

The choice of the Euler-Lagrange formulation in (1) is mainly motivated by the ability to find, in this form, patterns that can be exploited to obtain (by hand) a graph-compressing implementation of the dynamic model. As an example, denoting with $B_l[r,s]$ the (r,s)-th element of the inertia matrix $B_l(q)$ due to the l-th link, kinetic energy additivity allows to write $B_l[r,s] = M_l[r,s] + I_l[r,s]$, where

M_l and I_l represent the translational and rotational contributions, respectively, to the global inertia matrix. After some algebraic manipulation, M_l can be profitably put in the form

$$M_l[r,s] = \begin{cases} m_l\left(\sum_{i=\bar{i}}^{l}(a_i^l)^2 + \sum_{\substack{i=r\\ j=s\\ i\neq j}}^{l} a_i^l a_j^l \cos\bar{q}\right) & \text{if } r \le l \wedge s \le l \\ 0 & \text{if } r > l \vee s > l \end{cases}$$

where $\bar{i} = \max(r,s)$, $a_i^l = \{a_i \text{ if } i < l,\ a_i - c_i \text{ if } i = l\}$, and $\bar{q} = \sum_{k=\underline{k}}^{\bar{k}} q_k$, with $\underline{k} = \min(i,j)+1$ and $\bar{k} = \max(i,j)$. Similarly, the contribution of the angular velocity of each link to the global inertia matrix is trivially given by its moment of inertia $I_l[r,s] = \{i_l \text{ if } \max(r,s) \le l,\ 0 \text{ if } \max(r,s) > l\}$.

Pattern exploitation and code optimization enabled us to dramatically increase the efficiency of the computational pipeline.

D. Contact Force Model

A crucial aspect of planning with intermittent contacts is contact force modeling. As recently proposed in [10], two main approaches can be adopted: a penalty-based formulation and a complementarity-based approach. In this paper, we employ a penalty-based contact model due to its intrinsic smoothness that fits in particularly well with the Newton-type algorithms used to solve the resulting numerical optimization problem.

With the aim of approximating a unilateral linear contact model, we adopted the following differentiable constitutive relation (depicted in Fig. 2) between the normal contact force f_n and the normal gap y (it is $y = y(q)$)

$$f_n(y) = f_0 \log_2\left(1 + 2^{-\frac{\kappa}{f_0}y}\right) \tag{3}$$

where f_0 is the force value at $y = 0$, and $-\kappa$ is the contact stiffness as $y \to -\infty$. The rationale behind this choice is evident if one starts from the definition of a contact stiffness function $\partial f_n/\partial y = -\kappa/(1+2^{\kappa y/f_0})$, which constitutes a smooth approximation of a unilateral constant stiffness model $\partial f_n/\partial y = \{-\kappa \text{ if } y < 0,\ 0 \text{ if } y \ge 0\}$.

According to a regularization strategy of the stick-slip behavior [22, p. 80], the relation between tangential contact

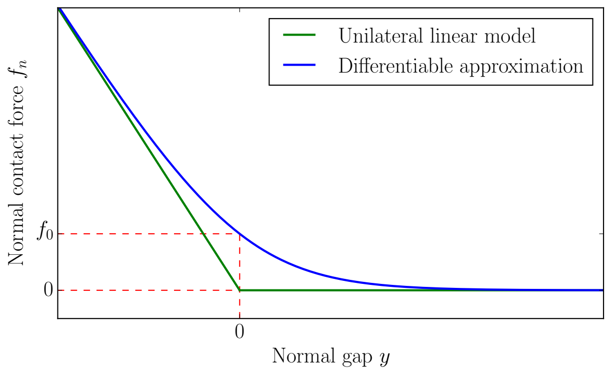

Fig. 2. Constitutive law of the normal contact force (differentiable approximation of a unilateral linear elastic model).

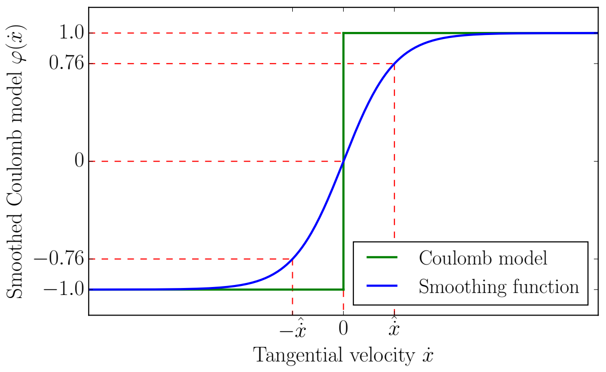

Fig. 3. Differentiable approximation of the Coulomb friction model.

force f_t and tangential velocity $\dot{x}$ for a 2D problem can be modeled by

$$f_t(y,\dot{x}) = -\mu f_n(y)\varphi(\dot{x}), \quad \text{with} \quad \varphi(\dot{x}) = \tanh\left(\frac{\dot{x}}{\hat{\dot{x}}}\right) \tag{4}$$

where μ is the static/dynamic coefficient of friction, φ is a smooth function that approximates the Coulomb model (Fig. 3), and $\hat{\dot{x}}$ is a reference sliding velocity at which the tangential force f_t is 76% of its asymptotic value.

The only form of dissipation introduced in this model comes from sliding actions. The choice of not including damping in the normal contact model is motivated by the following considerations:

- a "standard" normal damping force (proportional to the relative normal velocity) creates an unrealistic pulling force when the robot's joints break contacts rapidly;
- a physically consistent elimination of the previous phenomenon (as suggested in [23], Ch. 9, p. 238, Eq. (11.75)) would require two additional configurations per contact interface in order to model the damped spring back of the boundary surface when the applied contact force is rapidly removed, with a significant increase in the computational burden (also due to the non differentiable *if* function);
- the introduction of a smooth damping force requires at least the setting of two new coefficients, whose values would be quite arbitrary without further hypotheses/investigation;
- for the sake of testing the advantages of a two-stage strategy over a one-stage strategy, a full-fledged contact model would not bear special insight or greater advantages.

As a fair compromise between realistic modeling of contact actions and non-excessive numerical stiffness, we adopted $\kappa = 1 \cdot 10^5$ N/m and $f_0 = 2 \cdot 10^2$ N. With these values, the normal contact force required to support the total weight of the humanoid (696 N) corresponds to $y \approx -7$ mm (penetration), whereas its value at $y = 10$ mm is approximately 9 N and rapidly vanishes as y increases. With similar physical considerations we selected $\hat{\dot{x}} = 2 \cdot 10^{-2}$ m/s. The friction coefficient μ was set to 1.

E. Steady-Foot Requirements

As pointed out in Sec. II-A, the steady-foot assumption involves a set of bounds to the actions applied to the foot.

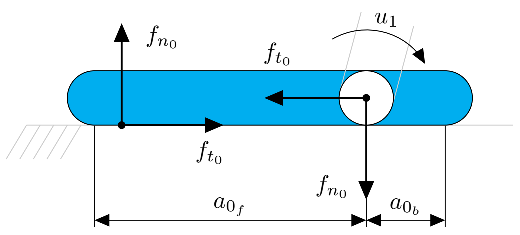

Fig. 4. Free-body diagram of the foot.

Referring to Fig. 4, these constraints can be stated as

$$f_{n_0} \geq 0 \quad \text{(unilateral contact)} \tag{5a}$$

$$-\mu \leq \frac{f_{t_0}}{f_{n_0}} \leq \mu \quad \text{(contact force} \in \text{friction cone)} \tag{5b}$$

$$-a_{0_f} \leq \frac{u_1}{f_{n_0}} \leq a_{0_b} \quad \text{(l.o.a.} \in \text{contact segment)} \tag{5c}$$

where $a_{0_f} = 0.200$ m, $a_{0_b} = 0.058$ m, and the moment arm of the tangential force is neglected. Relation (5c) requires that the line of action (l.o.a.) of the normal contact force f_{n_0} be inside the contact segment of the foot. Note that, since the EoMs were derived from the Virtual Work Principle, the contact reaction $f_{c_0}(q,\dot{q},\ddot{q})$ is not directly available, and its expression has to be obtained from inverse translational dynamics of the entire system. As fairly clear, constraints (5) are computationally very expensive.

F. Discretization

Direct trajectory optimization schemes require a discrete time (DT) version of system dynamics (1), contact force models (3)–(4), and foot constraints (5). With the goal of minimizing the computational burden of the DT EoMs, a straightforward discretization of the implicit form of (1) is performed: deriving a state-space representation, which would call for a repeated use of the computationally expensive (symbolic) inverse of $B(q)$, is therefore not necessary.

Using a direct transcription scheme based on a single collocation point selected as the midpoint of the generic interval $[t_k, t_{k+1}]$ for $k \in \{0,1,\ldots,N-1\}$, and denoting $h = T/N = t_{k+1} - t_k$ (T is the time horizon), $\bar{q}_k = (q_k + q_{k+1})/2$, $\bar{\dot{q}}_k = (\dot{q}_k + \dot{q}_{k+1})/2$, we write $\bar{B}_k = B(\bar{q}_k)$, $\bar{C}_k = C(\bar{q}_k, \bar{\dot{q}}_k)$, $\bar{G}_k = G(\bar{q}_k)$, and $\bar{Q}_k = Q(\bar{q}_k, \bar{\dot{q}}_k, u_k)$. According to this scheme, kinematic reconstruction and EoMs are expressed by

$$q_{k+1} - q_k - h\,\bar{\dot{q}}_k = 0 \tag{6a}$$

$$\bar{B}_k(\dot{q}_{k+1} - \dot{q}_k) + h(\bar{C}_k \bar{\dot{q}}_k + \bar{G}_k - \bar{Q}_k) = 0 \tag{6b}$$

where *states* $(q, \dot{q})$ are linear and *controls* u are constant over each discretization interval. The integration scheme expressed by (6), known as *implicit midpoint rule*, is a symplectic integrator: it can cope with stiff differential equations while ensuring that no artificial numerical damping is introduced.

Two concluding remarks have to be made:

- in order to be consistent with the proposed integration scheme, all differential constraints (such as (5)) must be evaluated at the collocation points;
- the discretization time h must be an appropriate trade-off between the need to keep the number of optimization variables reasonably low and the need to synthesize dynamically plausible trajectories (note that poor discretizations can produce "good-looking" trajectories that are useless in practice, since controllers would not be able to stabilize the robot around them).

G. Optimization

Within the direct transcription framework, equations (6) and the DT version of (5) constitute a set of nonlinear constraints for the optimal control problem we formulate. Additional constraints are: the initial states and input torques of static equilibrium; the goal configuration, slightly relaxed by bound constraints (also, we direct the solver to obtain a terminal configuration of static equilibrium by imposing $\dot{q}_N = \dot{q}_{N-1} = 0$); the upper and lower bounds for the configuration vector. As will be explained in Sec. III-C, two additional sets of constraints may be imposed in some cases: the constitutive models of contact forces (when they enter explicitly as optimization variables), and geometric constraints that prevent undesired contact at selected locations.

The vector of optimization variables $v \in \mathbb{R}^m$ is defined by vertically stacking the sequence

$$\{q_0, \dot{q}_0, u_0, f_{c_0}, \ldots, q_{N-1}, \dot{q}_{N-1}, u_{N-1}, f_{c_{N-1}}, q_N, \dot{q}_N\}$$

where f_{c_k} may be absent (see scheme $\mathcal{E}$ in Sec. III-A). The total number of optimization variables is given by

$$m = \underbrace{2n(N+1)}_{\text{states}} + \underbrace{nN}_{\text{inputs}} + \underbrace{2|\mathscr{F}|N}_{\substack{\text{contact forces}\\ \text{(optional)}}} - \underbrace{3n}_{\substack{\text{initial}\\ \text{conditions}}} - \underbrace{2n}_{\substack{\text{goal}\\ \text{conditions}}}$$

The nonlinear program (NLP) to be solved to obtain optimal trajectories can be cast in the following compact form

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & f(v) = v^\top W v \\ \text{subject to} \quad & g_{\min} \leq g(v) \leq g_{\max} \\ & v_{\min} \leq v \leq v_{\max} \end{aligned} \tag{7}$$

where the quadratic form $f(v): \mathbb{R}^m \to \mathbb{R}$ is the DT objective function and $W \in \mathbb{R}^{m \times m}$ is a block diagonal weight matrix whose elements penalize the sums of the squares of input torques, angular velocities, and their variations between consecutive steps. Eight scalar weights w are defined to populate W: five for the input torques $(w_{u_1}, w_{u_2}, \ldots, w_{u_5})$, and three for velocities $(w_{\dot{q}})$, accelerations $(w_{\Delta\dot{q}})$, and input torque variations $(w_{\Delta u})$. Their numeric values are adjusted from task to task, with the goal of obtaining human-like behaviors and well-balanced contributions by each cost term on the final objective function value.

Concerning initial guesses, we initialize configuration angles using a linear interpolation between initial and goal configurations, and thus velocities are initially set to constant values over the discretization intervals. Input torques and contact forces (if included among the optimization variables) are all initialized to zero.

TABLE II

PERFORMANCES OF DIFFERENT STRATEGIES IN SOLVING THE NLP (7).

		$\mathcal{E}$ $\mathcal{F}$ $1\mathcal{S}$	$\mathcal{A}$ $\mathcal{F}$ $1\mathcal{S}$	$\mathcal{A}$ $\mathcal{F}_{+}C$ $1\mathcal{S}$	$\mathcal{A}$ $\mathcal{F}_{+}C$ $2\mathcal{S}$
$\mathfrak{T}_1$	iterations	failed	417	287	272+31
	time (s)		1239	488	11+40=51
$\mathfrak{T}_2$	iterations	failed	242	193	93+51
	time (s)		735	382	4+81=85
$\mathfrak{T}_3$	iterations	331	208	694	47+129
	time (s)	691	742	1262	2+210=212
$\mathfrak{T}_4$	iterations	311	340	360	175+27
	time (s)	520	1056	696	8+37=45

III. SOLUTION STRATEGIES, BENCHMARK TASKS, AND NUMERICAL RESULTS

A. Solution Strategies

The main objective of this study is to propose a two-stage strategy for optimal trajectory planning of articulated-body systems interacting with the environment via unscheduled contact sequences. In the numerical tests that follow, a fixed time horizon $T = 5$ s and $N = 100$ (hence $h = 50$ ms) were used. Let us first outline the different formulations that eventually led us to such strategy. They will be motivated and assessed in the sections below.

1) Schemes $\mathcal{E}$ and $\mathcal{A}$: Contact forces were treated according to two different schemes, denoted $\mathcal{E}$ (embedded) and $\mathcal{A}$ (augmented). In $\mathcal{E}$, the constitutive relations of contact forces are embedded into the generalized force term of the EoMs (term $Q(q,\dot{q},u)$ in (1)). Conversely, in scheme $\mathcal{A}$, all contact forces appear explicitly as optimization variables, and their constitutive relations become equality constraints: some advantages of using contact forces as primary optimization variables are listed in [11].

2) Schemes $\mathcal{F}$ and $\mathcal{F}_{+}C$: In order to favor/allow interaction with the ground only at specific body parts while avoiding contact at others, two additional schemes, denoted $\mathcal{F}$ and $\mathcal{F}_{+}C$, were considered. In $\mathcal{F}$, the (potential) contact forces between all the joints (and hand) and the ground are modeled, while in $\mathcal{F}_{+}C$ the undesired contact forces at specific joints are replaced with geometric constraints that prevent collision of such joints with the ground (as a result, $\mathscr{F} \subset \{1,2,\ldots,n\}$ and $y_{O_i}(q) > 0$, with $i \in \{1,2,\ldots,n\} \backslash \mathscr{F}$).

3) One- and Two-Stage Schemes: The two-stage strategy that we propose is based on the formulation $(\mathcal{A},\mathcal{F}_{+}C)$. Continuation is applied in two steps, with the first optimization streamlined by the following simplifications:

- EoMs (1) are deprived of dynamic, Coriolis, and centrifugal terms, so that $G(q) = Q(q,\dot{q},u)$ (*pseudo-static model*);
- the number of nodes N is reduced from 100 to 30;
- the contact parameters are relaxed: $\kappa = 5 \cdot 10^4$ N/m, $f_0 = 3 \cdot 10^2$ N, $\hat{\dot{x}} = 1 \cdot 10^{-1}$ m/s.

The results of this first optimization are (interpolated to the finer grid and) used to warm-start the subsequent original, unsimplified optimization problem. This two-stage approach is denoted as $2\mathcal{S}$, while the symbol $1\mathcal{S}$ is used for all other single-run optimizations.

B. Benchmark Tasks

Three reference configurations for the humanoid were defined: standing, prone, and supine. Based on them, four benchmark tasks were devised: trajectories had to be planned for the humanoid to have it transition from prone to standing ($\mathfrak{T}_1$), standing to prone ($\mathfrak{T}_2$), supine to standing ($\mathfrak{T}_3$), and standing to supine ($\mathfrak{T}_4$). Such tasks are certainly not trivial, and they challenge the NLP solver in different ways.

In the following tests, the total number of optimization variables ranges between 1485 and 2485 for the cases $\mathcal{E}$ and $\mathcal{A}$, respectively.

C. Comparative Analysis of Solution Strategies and Numerical Results

Table II shows the performances (in terms of number of IPOPT iterations and CPU time to convergence) recorded for the four benchmark tasks (rows) as a function of the optimization strategy (columns). They were obtained on a notebook computer with a 2.70 GHz Intel(R) Core(TM) i7-4800MQ CPU and 32 GB of RAM.

Our investigation started by comparing the results obtained using strategy $\mathcal{E}$ with those provided by strategy $\mathcal{A}$, first two columns of Table II. In both cases, all contact forces were active (scheme $\mathcal{F}$) and optimizations were single-run (scheme $1\mathcal{S}$). The term 'failed' indicates that the solver was stuck at a point of local infeasibility. Unlike in $\mathcal{E}$, scheme $\mathcal{A}$ allows the solver to manage contact forces (extra optimization variables) and the Lagrange multipliers associated with their constitutive relations (equality constraints) in a more direct and effective way. This results in increased robustness (i.e., capability of obtaining a feasible solution), albeit at the expense of an increased time per iteration. When both $\mathcal{E}$ and $\mathcal{A}$ succeeded in obtaining feasible solutions (tasks $\mathfrak{T}_3$ and $\mathfrak{T}_4$), the resulting trajectories were identical. Due to its intrinsic robustness, we settled upon scheme $\mathcal{A}$ for the subsequent evaluation tests.

The next test was a comparative analysis of formulations $\mathcal{F}$ and $\mathcal{F}_{+}C$. The latter is aimed at minimizing the computational intricacy arising from the presence of multiple contact sources (see Sec. III-D for details). Columns two and three of Table II show a general improvement of $\mathcal{F}_{+}C$ over $\mathcal{F}$ in terms of both number of iterations and CPU time, except for task $\mathfrak{T}_3$. The obtained trajectories are slightly different from those obtained with scheme $\mathcal{F}$.

Finally, we compared the performance of the $(\mathcal{A},\mathcal{F}_{+}C,1\mathcal{S})$ scheme with that of the two-stage strategy $(\mathcal{A},\mathcal{F}_{+}C,2\mathcal{S})$, reported in columns three and four of Table II. As introduced in Sec. III-A, the first optimization problem of the two-stage approach is based on a *pseudo-static* model (in the sense that generalized forces still depend on velocities due to the smoothed Coulomb model we adopt), motivated by the fact that general stand-up and sit-down movements are realized relying more on contact forces with the ground than on inertial forces originated by very dynamic behaviors. Replacing (1) with its pseudo-static version $G(q) = Q(q,\dot{q},u)$ has the direct consequence of reducing the graph nodes associated with the EoMs by approximately 90%, with significant savings in terms of both memory usage and computational effort. In addition, the reduction of collocation nodes carries a proportional decrease in the number of optimization variables

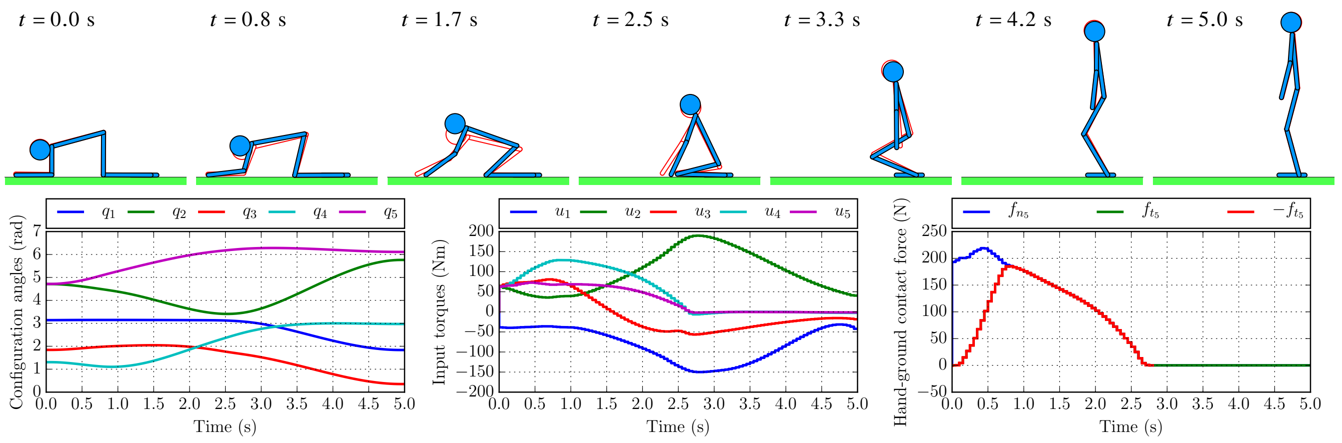

Fig. 5. Solutions of the prone-to-standing task ($\mathfrak{T}_1$). Top: frames of the final trajectory (light blue) and of the warm-start trajectory (red). Bottom, left to right: configuration angles, input torques, and absolute value of the hand-ground contact force (green or red depending on its sign) as functions of time. Set of active contacts (knee, elbow, hand): $\mathscr{F} = \{1, 4, 5\}$.

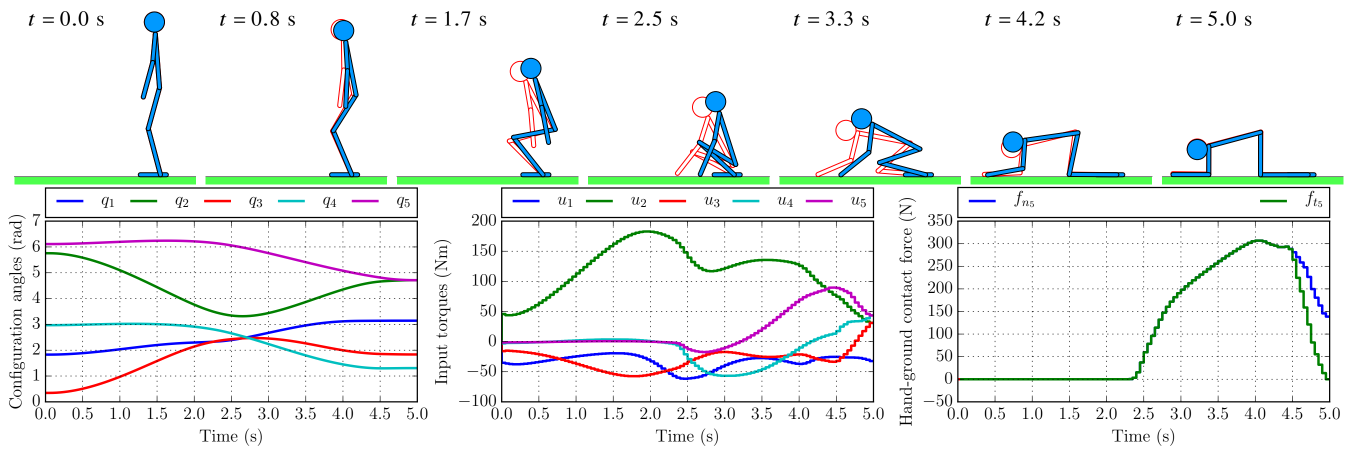

Fig. 6. Solutions of the standing-to-prone task ($\mathfrak{T}_2$). Set of active contacts (knee, elbow, hand): $\mathscr{F} = \{1, 4, 5\}$.

(leading to a substantial reduction in CPU time), while relaxation of the contact parameters mitigates numerical stiffness. Once the solution of such first optimization problem has been obtained, it is first mapped to the original (finer) integration grid through linear interpolation, and it is then used to warm-start the second, unsimplified optimization problem. Since the first optimization is intrinsically different from the second, we decided to warm-start only the primary optimization variables (not the dual variables nor the barrier parameter). The last column of Table II shows the level of performance achieved by the two-stage strategy: the reduction in the number of iterations with dynamic constraints varies between 74% and 93%, while CPU time was reduced by 78% to 94%. The computed trajectories are identical to those obtained by the ($\mathcal{A}$,$\mathcal{F}_{+}\mathcal{C}$,1$\mathcal{S}$) strategy.

D. Graphical Results

The results obtained with the two-stage strategy are shown in Figs. 5–8 for the four benchmark tasks. For each task, a sequence of frames are displayed where the final solution (light blue) is superimposed on the (interpolated) warm-start solution (red). In addition, configurations angles, input torques, and contact forces between the hand and the ground are represented as functions of time. Note that, since the friction coefficient is $\mu = 1$, the graphs of the normal and tangential contact forces (their absolute values) coincide when sliding occurs, while this is not the case in the presence of static friction.

We stress the fact that these results represent animations of the optimal trajectories synthesized by the NLP solver, and not forward integrations of the humanoid EoMs using the calculated, optimal input torques. In fact, unless the trajectories obtained from the optimization phase are uniformly stable, open-loop integrations cannot give acceptable results. This was confirmed as we tried different feed-forward simulations (with a finer discretization grid, a higher-order integration scheme, and a "sharper" contact model): in all cases, after a temporary coincidence with the ideal trajectories, the system's trajectories diverged, attesting that application of optimal control policies to unstable systems requires the presence of stabilizing controllers.

For all tasks, the solutions of the first optimization problem are very similar to the final ones, indicating that such simplified model does capture the essence of the various behaviors and related trajectories. The only exception is task $\mathfrak{T}_3$ (Fig. 7): because of a relatively low value of the weight coefficient $w_{\Delta\dot{q}}$, the final solution includes a second phase of the hand pushing on the ground that does not appear in

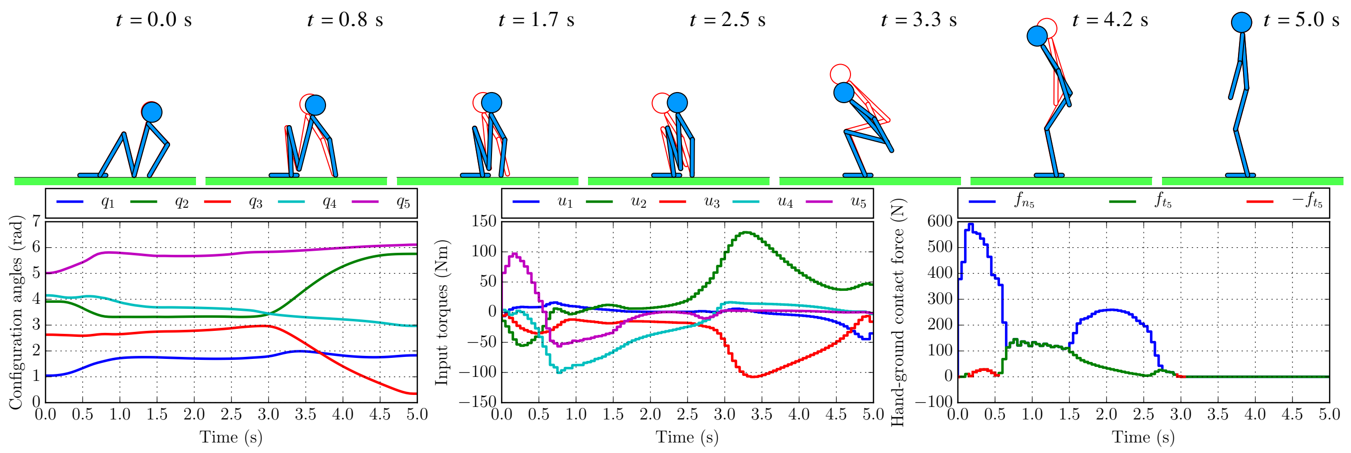

Fig. 7. Solutions of the supine-to-standing task ($\mathfrak{T}_3$). Top: frames of the final trajectory (light blue) and of the warm-start trajectory (red). Bottom, left to right: configuration angles, input torques, and absolute value of the hand-ground contact force (green or red depending on its sign) as functions of time. Set of active contacts (hip, hand): $\mathscr{F} = \{2, 5\}$.

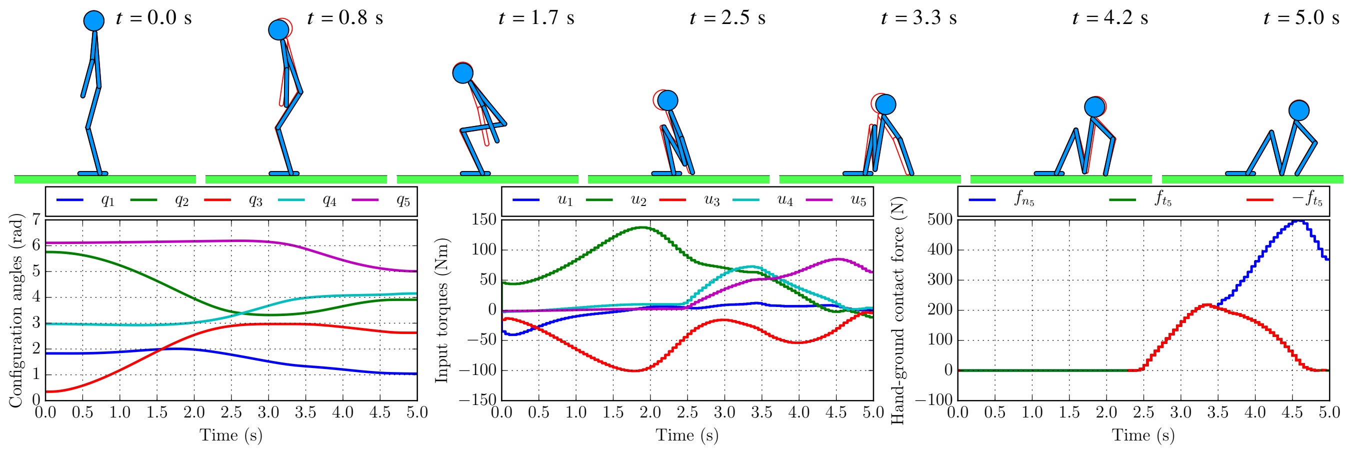

Fig. 8. Solutions of the standing-to-supine task ($\mathfrak{T}_4$). Set of active contacts (hip, hand): $\mathscr{F} = \{2, 5\}$.

the warm-start solution. This different behavior affects the CPU time of the second optimization, which is significantly higher than the average.

Animations of the frame sequences in Figs. 5–8 can be found in Section III-D of the video [24] accompanying this paper.

E. Remarks on the Pseudo-Static Model

Using the implicit midpoint rule, the DT first-order pseudo-static model appears as

$$G(\bar{q}_k) = Q\left(\bar{q}_k, \frac{q_{k+1} - q_k}{h}, u_k\right) \tag{8}$$

It is evident that this formulation does not relate the velocity values at two consecutive nodes in any way, making the acceleration cost term in the objective function indispensable to prevent wild velocity variations. Furthermore, also differences in configuration angles are not practically limited: whenever the relaxed Coulomb model in Fig. 3 is saturated (kinetic friction) or contacts are broken, (8) appears as an algebraic path constraint, and it requires that velocities be penalized. In light of these aspects, the pseudo-static model does not suffer coarse discretization grids that would cause, in the case of a dynamic model, poor approximations of the derivative functions. As a matter of fact, different tests carried out with the two-stage strategy confirmed that the CPU times required by the second optimization were larger if warm-started with a solution obtained with the complete dynamic model (6). The truncation errors accumulated in the computation of the coarse dynamic warm-start solution misled the second optimization more than the approximate pseudo-static model did.

It is interesting to notice that the reduction of collocation nodes pertaining to the first optimization goes in the direction of limiting: the complexity of the discovered behaviors and, as a consequence, the negative effects of the exploitation of a different dynamic model in the first phase. Finally, it should be noted that, using such a dynamic model, friction forces are the only horizontal actions and they need to balance one another. In light of this, the extension of this model to all the cases where a horizontal ambulation of the robot is sought requires the introduction of a horizontal resistance force (such as, for example, an overall inertial force).

IV. CONCLUSIONS

In this paper we have described and numerically assessed different formulations for direct trajectory optimization, applied here to a 2D humanoid robot that interacts with the environment through contacts. In particular, a two-stage strategy for the solution of the resulting NLP proved to be particularly efficient from a computational standpoint. Thanks to a smooth contact model (suitable for gradient-based numerical optimization algorithms) and to the use of a discretization scheme that can handle stiff dynamics, the entire process exhibited a high level of robustness. Several tests were performed with different contact parameters, weighting schemes, discretization grids: in none of them was the solver trapped into local infeasibility, and satisfactory solutions were obtained in all cases.

V. ACKNOWLEDGEMENT

This work is supported by the grant no. 611832 "WALK-MAN" within the FP7-ICT-2013-10 program "Cognitive Systems and Robotics" and by grant no. 600918 "PACMAN" - Probabilistic and Compositional Representations of Object for Robotic Manipulation - within the FP7-ICT-2011-9 program "Cognitive Systems".

REFERENCES

[1] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. Journal of Robotics Research (IJRR)*, vol. 33, no. 1, pp. 69–81, 2014.

[2] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.

[3] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," in *ACM Transactions on Graphics*, 2012.

[4] J. T. Betts, *Practical Methods for Optimal Control Using Nonlinear Programming*. SIAM, 2001.

[5] M. Diehl. (2014) Lecture notes on optimal control and estimation. [Online]. Available: http://syscop.de/wp-content/uploads/2015/03/oce_script.pdf

[6] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber, *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, ch. Fast Direct Multiple Shooting Algorithms for Optimal Robot Control, pp. 65–93. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-36119-0_4

[7] H. Bock and K. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *9th IFAC World Congress*, 1984.

[8] T. Tsang, D. Himmelblau, and T. Edgar, "Optimal control via collocation and non-linear programming," *Int. Journal of Control (IJC)*, vol. 21, pp. 763–768, 1975.

[9] W. Xi and C. Remy, "Optimal gaits and motions for legged robots," in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014, pp. 3259–3265.

[10] M. Gabiccini, A. Artoni, G. Pannocchia, and J. Gillis, "A computational framework for environment-aware robotic manipulation planning," in *International Symposium on Robotics Research (ISRR)*, 2015.

[11] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2012, pp. 137–144.

[12] I. Mordatch and E. Todorov, "Combining the benefits of function approximation and trajectory optimization," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[13] D. Pardo, L. Möller, M. Neunert, A. W. Winkler, and J. Buchli, "Evaluating direct transcription and nonlinear optimization methods for robot motion planning," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 946–953, 2016.

[14] P. de Leva, "Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters," *Journal of Biomechanics*, vol. 29, no. 9, pp. 1223 – 1230, 1996.

[15] J. Andersson, "A General-Purpose Software Framework for Dynamic Optimization," PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, October 2013.

[16] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[17] I. S. Duff, "MA57—a code for the solution of sparse symmetric definite and indefinite systems," *ACM Transactions on Mathematical Software (TOMS)*, vol. 30, no. 2, pp. 118–144, 2004.

[18] HSL, "A collection of Fortran codes for large scale scientific computation." 2014. [Online]. Available: http://www.hsl.rl.ac.uk

[19] I. Wolfram Research, *Mathematica*, ser. Version 10.2. Champaign, Illinois: Wolfram Research, Inc., 2015.

[20] J. Gillis and M. Diehl, "Hierarchical seeding for efficient sparsity pattern recovery in automatic differentiation," in *CSC14: The Sixth SIAM Workshop on Combinatorial Scientific Computing*, 2014.

[21] A. H. Gebremedhin, F. Manne, and A. Pothen, "What color is your Jacobian? Graph coloring for computing derivatives," *SIAM Review*, vol. 47, pp. 629–705, 2005.

[22] P. Wriggers, *Computational contact mechanics*. New York: Springer, 2006.

[23] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer US, 2008.

[24] T. Marcucci, M. Gabiccini, and A. Artoni, "A two-stage trajectory optimization strategy for articulated bodies with unscheduled contact sequences — submission accompanying video," https://youtu.be/FCL0cPYVKVc, Mar. 2016.