Efficient Multi-Contact Pattern Generation with Sequential Convex Approximations of the Centroidal Dynamics

Brahayam Ponton[1], Majid Khadiv[1], Avadesh Meduri[2] and Ludovic Righetti[1,2]

***Abstract*—This paper investigates the problem of efficient computation of physically consistent multi-contact behaviors. Recent work showed that under mild assumptions, the problem could be decomposed into simpler kinematic and centroidal dynamic optimization problems. Based on this approach, we propose a general convex relaxation of the centroidal dynamics leading to two computationally efficient algorithms based on iterative resolutions of second-order cone programs. They optimize centroidal trajectories, contact forces, and importantly the timing of the motions. We include the approach in a kino-dynamic optimization method to generate full-body movements. Finally, the approach is embedded in a mixed-integer solver to further find dynamically consistent contact sequences. Extensive numerical experiments demonstrate the computational efficiency of the approach, suggesting that it could be used in a fast receding horizon control loop. Executions of the planned motions on simulated humanoids and quadrupeds and on a real quadruped robot further show the quality of the optimized motions.**

I. Introduction

The computation of multi-contact motions remains a difficult yet important challenge for legged locomotion and manipulation in order to afford more versatile behaviors in complex environments. Of particular interest are methods that can compute such motions in real-time without making restrictive assumptions on the solution set. Indeed, they can provide the necessary adaptive behavior required in uncertain environments without trading-off motion versatility.

Very successful walking pattern generators often rely on simplified linear models of the dynamics [1] as they offer important computational advantages that make them suitable for receding horizon control [2–4]. Unfortunately, these models are fundamentally restricted to locomotion patterns with predefined gaits on quasi-flat grounds. While extensions of such models can enable the use of hands to maintain balance [5], they make substantial assumptions on the admissible gaits and are thus limited by the range of gaits they can generate.

Complete rigid body dynamics models including interaction dynamics, in principle, afford the synthesis of a wider range of behaviors for more complex motion tasks. Despite the inherent computational challenges, very impressive motions can be computed [6–16]. However such approaches are often limited for receding horizon control as they require the resolution of nonconvex, high dimensional optimization problems, often with complex nonlinear constraints such as complementarity constraints for contact dynamics.

This research was supported by New York University, the Max-Planck Society, the European Union's Horizon 2020 research and innovation programme (grant agreement No 780684 and European Research Council's grant No 637935), and the US National Science Foundation grant CMMI-1825993.

[1] Max Planck Institute for Intelligent Systems, Tübingen - Germany

[2] New York University, New York - USA

Middle-complexity options that decouple the pattern generation problem into simpler sub-problems have also been studied. They typically assume that a sequence of contact configuration is provided first, typically using efficient search algorithms for contact sequences [17–21]. Of special interests are methods based on the centroidal dynamics of the robot [1], [22], [23] which have become very popular recently [24–26]. Indeed, under mild assumptions on the kinematic and actuation feasibility, this model provides sufficient conditions to plan dynamically consistent full-body motions with multiple contacts. This model is simple enough to be amenable to online resolution and at the same time, expressive to plan complex behaviors [27–31]. It is then possible to combine momentum dynamics with a full kinematic model to plan highly dynamic motions [25]. This decomposition between centroidal dynamics and kinematics models was, for example, leveraged to create an alternating algorithm that efficiently computes full-body motions in multi-contact by iteratively solving two separate optimization problems until they reach consensus [26], [31]. This connection has then been further explored in [32], which proposed a method to optimize both centroidal and full-body motions using an Alternating Direction Method of Multipliers formulation.

While promising, approaches based on the centroidal momentum dynamics are inherently nonconvex and thus still challenging to solve efficiently. This led researchers to focus on the mathematical structure of the problem to derive more efficient methods. For example, convex bounds on the angular momentum rate (that maximizes the contact wrench cone margin) are used to minimize a worst-case bound on the l_1 angular momentum norm via convex optimization [33]. In [27], [28], the bilinear terms of the momentum dynamics and timings are handled by a dedicated multiple-shooting solver and, proxy constraints are used for handling whole-body limits based on an offline learning method. [30] exploits a linear approximation of the momentum dynamics based on a lower-dimensional space projection and an adaptive method for timing optimization to control a robot in multi-contact scenarios in a receding horizon fashion. In [34], [35], the interpretation of friction cones as dual twists allows computing online cones of feasible CoM accelerations. The resulting bilinear constraints are decoupled into linear pairs via a conservative trajectory-wide contact-stability criterion for online motion generation. Timings between contact switches are optimized online by solving an easy-to-solve nonlinear problem.

In [26], we further studied the problem structure and proposed an analytic decomposition of positive and negative definite terms of the problem Lagrangian based on the decomposition of angular momentum nonconvex terms. This led to a solver with improved convergence properties. In our

previous work [36], we proposed a convex relaxation of the problem that suggested the use of a proxy function to minimize angular momentum, namely the sum of the squares of the terms composing the nonconvex part of the dynamics. While computationally very efficient, this approach was limited as it did not allow the inclusion of an explicit target angular momentum in the cost function, therefore severely limiting the space of solutions. Moreover, the approach could not be used with the alternating full-body motion optimization method discussed above.

In this paper, extending our preliminary work [37], we study a general convex relaxation of the problem that allows the explicit inclusion of angular momentum objectives and naturally extends to the optimization of timing, a feature missing in most contributions on centroidal dynamics optimization. The main contributions of the paper are*

1 Exploiting the structure of the centroidal dynamics optimization problem, we propose two computationally efficient algorithms formulated as a sequence of convex second-order cone programs to compute physically consistent center of mass, angular momentum, and contact force trajectories and demonstrate how timing optimization can be efficiently included.
2 We show how our approach can be efficiently used with the kino-dynamic optimization method proposed in [31] to generate full-body physically-consistent movements. We further extend the approach to also include actuation limit constraints.
3 We extend the approach in a mixed-integer program to find dynamically consistent contact sequences and locations.
4 Finally, we evaluate the capabilities and limitations of our approach in simulation on several multi-contact scenarios for a biped and a quadruped robot, we study the benefits of timing optimization to extend the range of possible behaviors and demonstrate the execution of these movements on a real quadruped robot.

The software implementation of the algorithms presented in this paper is open-source and freely available [38]. We state the problem and present background material in Section II. In Section III, we detail the motion optimization approach and in Section IV the contacts planning approach using mixed integer programming. We present simulation and real robot results in Section V and discuss the features and limitations of our proposed framework in Section VI. Finally, we conclude in Section VII.

II. Preliminaries and Problem Formulation

In this section, we introduce the centroidal dynamics optimization problem for multi-contact locomotion in the larger context of full-body optimization. First, we provide an overview of the larger kino-dynamic optimization problem, present the structured approach used in our architecture to tackle it, and outline the centroidal dynamics optimization problem, which is the core focus of this paper. Our overall approach is summarized in Figure 1. From a task description, we first select a sequence of physically-feasible contact sequences using mixed-integer programming (Sec. IV). This sequence is used to optimize time-optimal full-body movements using our kino-dynamic solver (Fig. 3). These movements are then tracked with an instantaneous whole-body feedback controller.

*Part of the material was presented at the 2018 IEEE-RAS International Conference on Robotics and Automation [37]. Contribution 1 is an extension of this work, Contributions 2 and 3 are novel, Contribution 4 extends simulation results to Contribution 2 and 3 and presents real robot experiments.

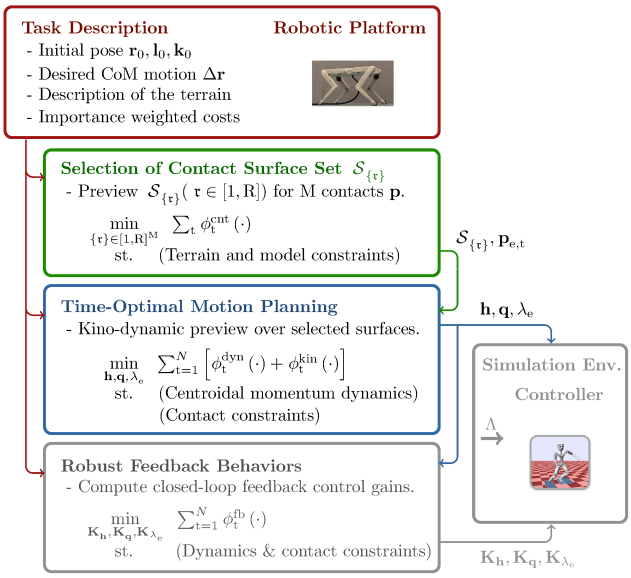


Fig. 1: Our architecture maps a high-level task description into functional motions. The initial state $\mathbf{r}_0, \mathbf{l}_0, \mathbf{k}_0$ of the robotic platform (simulated humanoid or a real quadruped robot), a desired CoM motion $\Delta\mathbf{r}$, a description of the R surfaces that compose the terrain and a set of costs $\phi_t^{\mathrm{cnt}}(\cdot), \phi_t^{\mathrm{kin}}(\cdot), \phi_t^{\mathrm{dyn}}(\cdot), \phi_t^{\mathrm{fb}}(\cdot)$ are used to select a set of surfaces $\mathcal{S}_{\{\mathfrak{r}\}}$ that support a dynamic motion, optimize a kino-dynamic motion over a discrete time horizon N, and synthesize a set of feedback gains $\mathbf{K_h}, \mathbf{K_q}, \mathbf{K}_{\lambda_e}$ that define closed-loop behaviors to be realized by an inverse dynamics controller as in [39–41].

A. Kino-dynamic optimization of multi-contact behaviors

To synthesize full-body multi-contact behaviors, we seek to efficiently solve an optimal control problem of the form

$$\min_{\mathbf{q}(t),\Lambda(t),\lambda_e(t)} \phi_{\mathrm{end}}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\Lambda,\lambda_e) + \int_0^{\mathcal{T}} \phi_t(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}},\Lambda,\lambda_e)\,\mathrm{dt} \quad \text{(1a)}$$

$$\text{subject to } \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{S}^T\Lambda + \sum_{e\in e_{\mathrm{cnt}}} \mathbf{J}_e(\mathbf{q})^T \lambda_e \quad \text{(1b)}$$

$$\mathbf{q}_{\mathrm{jnt}} \in [{}^{\min}\mathbf{q}_{\mathrm{jnt}}, {}^{\max}\mathbf{q}_{\mathrm{jnt}}] \quad \text{(1c)}$$

$$\Lambda \in [{}^{\min}\Lambda, {}^{\max}\Lambda] \quad \text{(1d)}$$

$$(\lambda_e, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \in \Omega \quad \text{(1e)}$$

which minimizes a performance cost $\phi(\cdot)$, composed of a terminal cost ϕ_{end} and the integral of a running cost ϕ_t, over a finite time horizon $\mathcal{T}$ under a set of physical constraints. It enforces the equations of motion for a floating-base rigid-body system (Eq. (1b)), joint and torque limits (Eqs. (1c)-(1d)), as well as contact forces, velocity and acceleration constraints (Eq. (1e)). Here, $\mathbf{q} = \left[\mathbf{x}^T\ \mathbf{q}_{\mathrm{jnt}}^T\right]^T$ denotes the robot posture composed of $\mathbf{x} \in SE(3)$, the pose of the floating-base

relative to an inertial frame, and $\mathbf{q}_{\text{jnt}} \in \mathbb{R}^n$, the joint positions, where n is the number of joints. $\Lambda(t) \in \mathbb{R}^n$ are joint torques and $\lambda_e(t) \in \mathbb{R}^6$ is the contact wrench of endeffector $e \in e_{\text{cnt}}$ (where e_{cnt} is the set of endeffectors in contact with the environment at the time in question). $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{(n+6)\times(n+6)}$ is the inertia matrix; $\mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n+6}$ a vector of generalized forces including Coriolis, centrifugal, gravity and joint friction forces. $\mathbf{S} = \begin{bmatrix}\mathbf{0}^{n\times 6} & \mathbf{I}^{n\times n}\end{bmatrix}$ is a selection matrix reflecting the system under-actuation and $\mathbf{J}_e(\mathbf{q}) \in \mathbb{R}^{6\times(n+6)}$ is the contact Jacobian of endeffector e. The pre-superscripts min and max for joint positions $\mathbf{q}_{\text{jnt}}$ and joint torques Λ denote their minimum and maximum limits. The set Ω denotes constraints such as friction or non-sliding contacts, that will be explicitly defined within the next subsection. Note that additional kinematic constraints could also be added to the problem without changing the reasoning below.

The problem described in Eq. (1) is nonlinear, nonconvex, and computationally intensive and we seek to formulate a more tractable approximation without sacrificing the versatility of motion synthesis. The equations of motion can be decomposed into actuated (superscript a) and unactuated parts (superscript u)

$$\mathbf{M}^u(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}^u(\mathbf{q},\dot{\mathbf{q}}) = \sum_{e\in e_{\text{cnt}}} \mathbf{J}_e^u(\mathbf{q})^T \lambda_e \tag{2a}$$

$$\mathbf{M}^a(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}^a(\mathbf{q},\dot{\mathbf{q}}) = \sum_{e\in e_{\text{cnt}}} \mathbf{J}_e^a(\mathbf{q})^T \lambda_e + \Lambda \tag{2b}$$

As shown in [39], the actuated part of the dynamics provides the necessary actuation torques needed to achieve any combination of desired acceleration $\ddot{\mathbf{q}}$ and contact forces λ_e. Thus, assuming sufficient actuation Λ, it is possible to ignore the actuated part of the equations of motion (Eq. (2b)) and base the synthesis of multi-contact behaviors only on the unactuated part (Eq. (2a)). As we will later show in the paper, it is nevertheless possible to add torque limits in the decoupled optimization problems. In [41], [42], it has been shown that the right-hand side of the unactuated part and the gravitational effects of the vector of nonlinear terms $\mathbf{h}^u(\mathbf{q},\dot{\mathbf{q}})$ that relate the acceleration of the floating-base to external contact forces, are equivalent to the robot centroidal momentum dynamics

$$\begin{bmatrix}\dot{\mathbf{l}}\\ \dot{\mathbf{k}}\end{bmatrix} = \underbrace{\begin{bmatrix} m\mathbf{g} + \sum_{e\in e_{\text{cnt}}} \mathbf{f}_e \\ \sum_{e\in e_{\text{cnt}}} \left((\mathbf{p}_e + \mathbf{R}_e^{x,y}\, \jmath_e - \mathbf{r}) \times \mathbf{f}_e + \mathbf{R}_e^z\, \tau_e\right)\end{bmatrix}}_{\text{From Newton-Euler dynamics}} \tag{3}$$

The center of mass position is denoted $\mathbf{r}$ and the linear and angular momentum expressed at the CoM are written as $\mathbf{l}$ and $\mathbf{k}$. m is the robot mass and $\mathbf{g}$ the gravity vector. The endeffector frame is located at the endeffector position $\mathbf{p}_e$, and it is oriented so that $\mathbf{R}_e^z \in \mathbb{R}^{3\times 1}$ is normal to the contact surface, and $\mathbf{R}_e^x, \mathbf{R}_e^y \in \mathbb{R}^{3\times 1}$ are aligned with the rectangular shape of the endeffector support surface in the desired motion direction. The rotation matrix $\mathbf{R}_e = \begin{bmatrix}\mathbf{R}_e^x & \mathbf{R}_e^y & \mathbf{R}_e^z\end{bmatrix} \in \mathbb{R}^{3\times 3}$ rotates quantities from endeffector to inertial frame. For instance, the endeffector force $\mathbf{f}_e$, expressed in the inertial frame, is equivalent in local endeffector coordinates to $\mathfrak{f}_e = \mathbf{R}_e{}^T \mathbf{f}_e$. The center of pressure (CoP) $\jmath_e \in \mathbb{R}^2$ expressed in local endeffector

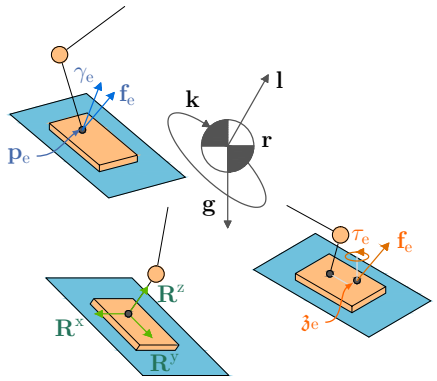


Fig. 2: The figure illustrates the representation used in the paper.

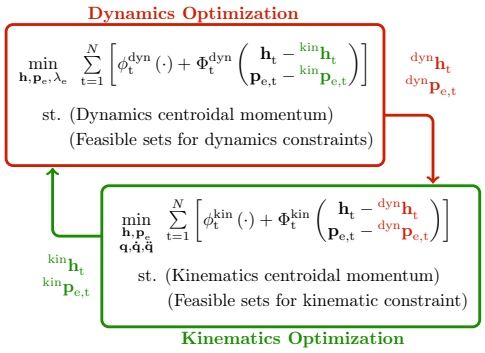


Fig. 3: Schematic of the kino-dynamic optimization approach that iteratively computes contact force λ_e and whole-body trajectories $\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}$ until convergence of the common set of variables: CoM $\mathbf{r}_t$, robot momenta $\mathbf{l}_t, \mathbf{k}_t$ and endeffector poses $\mathbf{p}_{e,t}$. The vector $\mathbf{h}_t$ is built by vertically stacking CoM and robot momenta. The pre superscripts kin and dyn relate the variables to the problem they are a solution for. The optimization objective ϕ is assumed to be separable and composed by $\phi_t^{dyn} + \phi_t^{kin}$. Finally, the cost penalties $\Phi_t^{dyn}, \Phi_t^{kin}$ ensure the consensus of the solutions at convergence.

frame and scalar torque τ_e at the CoP complete the description of the endeffector wrench. They can be equivalently described by a torque at $\mathbf{p}_e$ as $\gamma_e = (\mathbf{R}_e^{x,y}\, \jmath_e) \times \mathbf{f}_e + \mathbf{R}_e^z\, \tau_e$. The endeffector wrench can now be defined as $\lambda_e = \begin{bmatrix}\mathbf{f}_e^T & \gamma_e^T\end{bmatrix}^T$. Figure 2 depicts coordinate frames and the notation.

It has been shown [22] that the left-hand side of the unactuated part in Eq. (2a), under an appropriate coordinate transformation from the floating base to the robot's CoM, relates the robot rate of momenta expressed at the robot's center of mass $(\dot{\mathbf{l}},\dot{\mathbf{k}})$ to the robot velocity $\dot{\mathbf{q}}$ and acceleration $\ddot{\mathbf{q}}$ via the centroidal momentum matrix $\mathbf{M}_{\text{CoM}}^u(\mathbf{q}) \in \mathbb{R}^{6\times(n+6)}$.

$$\underbrace{\frac{d}{dt}\left[\mathbf{M}_{\text{CoM}}^u(\mathbf{q})\dot{\mathbf{q}}\right]}_{\text{From full-body kinematics}} = \mathbf{M}_{\text{CoM}}^u(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{M}}_{\text{CoM}}^u(\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix}\dot{\mathbf{l}}\\ \dot{\mathbf{k}}\end{bmatrix} \tag{4}$$

At this point, it becomes clear that the problem of finding feasible multi-contact motions can be reduced to the optimization of centroidal dynamics (Eq. (3)) and the optimization of full-body kinematics (Eq. (4)) as long as the motion-induced momentum agrees with the dynamic optimization. In [31],

an alternating algorithm to solve the optimal control problem (1) using this idea was proposed (see Fig. 3). It optimized centroidal dynamic motions and full-body kinematics separately but ensured through added cost penalties that both optimization problems come to an agreement on their common variables: CoM, momentum, and contact locations.

In this paper, we use the complete architecture shown in Figure 1 to evaluate our contributions, but our work mostly focuses on the centroidal dynamics optimization problem, which is sufficient to synthesize physically consistent motion behaviors.

B. Dynamic optimization with the centroidal dynamics

We now present in detail the centroidal dynamics optimization problem we are interested in, that synthesizes a motion plan (timing, contact wrenches, and momentum trajectories) under the momentum dynamics (Eq. (3)) and is optimal in terms of a desired quadratic performance objective. First, we discretize the dynamics equations using Euler's methods and then seek a local solution for the following problem:

$$\min_{\substack{\mathbf{h}_t,\Delta_t,\mathbf{p}_{e,t}\\ \mathbf{f}_{e,t},\mathfrak{z}_{e,t},\tau_{e,t}}} \sum_{t=1}^{N}\left[\phi_t^{\text{dyn}}\begin{pmatrix}\mathbf{h}_t,\Delta_t,\mathbf{p}_{e,t},\\ \mathfrak{z}_{e,t},\mathbf{f}_{e,t},\tau_{e,t}\end{pmatrix}+\Phi_t^{\text{dyn}}\begin{pmatrix}\mathbf{h}_t-{}^{\text{kin}}\mathbf{h}_t\\ \mathbf{p}_{e,t}-{}^{\text{kin}}\mathbf{p}_{e,t}\end{pmatrix}\right] \tag{5a}$$

$$\text{subject to } \mathbf{h}_t=\begin{bmatrix}\mathbf{r}_t\\ \mathbf{k}_t\\ \mathbf{l}_t\end{bmatrix}=\begin{bmatrix}\mathbf{r}_{t-1}+\frac{1}{m}\mathbf{l}_t\Delta_t\\ \mathbf{k}_{t-1}+\sum_{e\in e_{cnt}}\kappa_{e,t}\Delta_t\\ \mathbf{l}_{t-1}+m\mathbf{g}\Delta_t+\sum_{e\in e_{cnt}}\mathbf{f}_{e,t}\Delta_t\end{bmatrix} \tag{5b}$$

$$\kappa_{e,t}=(\mathbf{p}_{e,t}-\mathbf{r}_t)\times\mathbf{f}_{e,t}+\gamma_{e,t} \tag{5c}$$

$$\gamma_{e,t}=(\mathbf{R}_{e,t}^{x,y}\,\mathfrak{z}_{e,t})\times\mathbf{f}_{e,t}+\mathbf{R}_{e,t}^{z}\,\tau_{e,t} \tag{5d}$$

$$\mathbf{p}_{e,t}\in\mathcal{U}(\mathcal{S}_{\mathfrak{r}=\varphi(e,t)}) \tag{5e}$$

$$\Delta_t\in[{}^{\min}\Delta_t,{}^{\max}\Delta_t] \tag{5f}$$

$$\mathfrak{z}_{e,t}^{x,y}\in[{}^{\min}\mathfrak{z}^{x,y},{}^{\max}\mathfrak{z}^{x,y}] \tag{5g}$$

$$\left\|\mathfrak{f}_{e,t}^{x,y}\right\|_2\le\mu\,\mathfrak{f}_{e,t}^{z},\quad \mathfrak{f}_{e,t}^{z}>0 \tag{5h}$$

$$\left\|\mathbf{p}_{e,t}-\mathbf{r}_t\right\|_2\le{}^{\max}\mathcal{L}_e \tag{5i}$$

$$\Lambda_t=\left(\mathbf{M}^a({}^*\mathbf{q})\,{}^*\ddot{\mathbf{q}}+\mathbf{h}^a({}^*\mathbf{q},{}^*\dot{\mathbf{q}})-\sum_{e\in e_{cnt}}\mathbf{J}_e^a({}^*\mathbf{q})^T\,\lambda_{e,t}\right)\in[{}^{\min}\Lambda,{}^{\max}\Lambda] \tag{5j}$$

We minimize a quadratic cost (5a) that includes a running cost ϕ_t^{dyn} composed by user-defined task costs (such as reaching a CoM position or moving through a way-point) and regularization of control variables (such as contact wrenches or Euler discretization of time Δ_t). When the problem is solved in the context of the alternating kino-dynamic optimization procedure, it also includes a consensus cost Φ_t^{dyn} penalizing momentum trajectories and contact locations deviating from the solution of the kinematic optimization step. The problem is optimized over a discrete time horizon $N\approx\mathcal{T}/{}^0\Delta_t$ computed using the initial guess for the timestep variable Δ_t, that corresponds to the difference between time at step t and $t-1$.

The constraints (defined for all active endeffectors $e\in e_{cnt}$ and timesteps t) include consistency with the centroidal dynamics (5b)-(5d). Here, we have formulated the dynamics using torques at each contact's center of pressure and added an extra variable $\kappa_{e,t}$ which will facilitate the formulation of the time optimization algorithm. Other constraints include: constraints on the endeffector locations to remain on the assigned contact surface (5e) modeled as linear inequality constraints (cf. Section IV-A for a detailed explanation of $\mathbf{p}_{e,t}\in\mathcal{U}(\mathcal{S}_{\mathfrak{r}=\varphi(e,t)})$), box constraints to restrict the timestep variable (5f) between a lower ${}^{\min}\Delta_t$ and upper ${}^{\max}\Delta_t$ limits, constraints to maintain the CoP of the endeffectors (assumed to be rectangular) within the support region (5g) defined by the lower ${}^{\min}\mathfrak{z}^{x,y}$ and ${}^{\max}\mathfrak{z}^{x,y}$ upper limits, friction cone constraints (5h) (with friction coefficient μ) and a heuristic constraint to ensure that the contact locations remain reachable expressed as a distance from the CoM (5i) that cannot exceed a predefined value ${}^{\max}\mathcal{L}_e$. A linear time-varying approximation of the torque limits constraint (5j) along the motion trajectory ${}^*\mathbf{q},{}^*\dot{\mathbf{q}},{}^*\ddot{\mathbf{q}}$ optimized in the previous kinematics optimization problem can also be considered and provides the ability to adapt contact wrenches to satisfy torque limits.

In its general form, the optimization problem defined in Eq. (5) is nonconvex. Its nonconvexities are due to the cross products from the angular momentum dynamics and the bilinear terms from the timestep variable. In the next section, we leverage the structure of the problem and propose two algorithms based on convex relaxations to efficiently solve it. We then extend the approach to also optimally select contact surfaces that support a dynamic motion by embedding the dynamics model within a custom mixed-integer solver.

Remark 1: In general, we can write down the relationship between the contact forces and the CoM motion in two ways, 1) using the contact wrench sum (CWS) at the CoM and imposing contact wrench cone (CWC) constraints [33], [35], [40], [43] 2) using the contact forces (or wrench) at each endeffector and imposing directly contact force constraints [26], [27], [29], [37]. In this paper, we use the second approach. The main advantage of this approach is the capability of adapting the contact location of the endeffectors. The main caveat is that for more than one endeffector in contact (i.e. $e\ge 2$), the number of decision variables (i.e. $6\times e$) is more than the minimal representation of the centroidal wrench (i.e. 6). However, the cross product term between decision variables is inherent in the centroidal dynamics and our approach to dealing with the cross-product (and bilinear terms in general) is also applicable to a CWC formulation.

III. Centroidal Momentum Dynamics Optimization

This section presents our approach to solve the centroidal dynamics optimization based on an analytical decomposition of nonconvex bilinear expressions as a difference of quadratic functions, whose known curvature is exploited to design efficient iterative convex approximations. In the following, we analyze the nature of nonconvexities of problem (5), propose two convex relaxations to approximate them and, detail the optimization procedures and their convergence criteria.

A. Bilinear terms as difference of quadratic functions

Some constraints in problem (5) are affine (5e)-(5g), (5j) or second-order cones (SOC) (5h)-(5i) and thus convex;

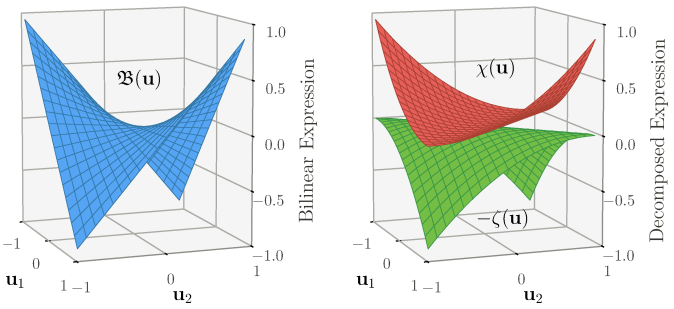


Fig. 4: Decomposition (as shown in *Definition* 2) of the bilinear form $\mathfrak{B}(\mathbf{u}) = \mathfrak{B}([\mathbf{u}_1{}^T, \mathbf{u}_2{}^T]^T) = \mathbf{u}_1 \cdot \mathbf{u}_2$ into a difference of quadratic expressions $\mathfrak{B}(\mathbf{u}) = \chi(\mathbf{u}) - \zeta(\mathbf{u})$ with $\chi(\mathbf{u}) = \frac{1}{4}\mathfrak{Q}(\mathbf{u}_1 + \mathbf{u}_2)$ and $\zeta(\mathbf{u}) = \frac{1}{4}\mathfrak{Q}(\mathbf{u}_1 - \mathbf{u}_2)$, where $\mathfrak{Q}(\cdot)$ is the quadratic function $\|\cdot\|_2^2$.

others however describe nonconvex constraints such as the momentum dynamics evolution when considering the timestep variable Δ_t as an optimization variable (5b) or torque cross products (5c)-(5d). Next, we show the common nature of all the nonlinearities and reformulate them in a way amenable to efficient approximations using iterative convex models.

The torque cross product $\ell \times \mathbf{f}$ between a length $(\mathbf{p}_e - \mathbf{r})$ in (5c) or $\mathbf{R}_e^{x,y}\, \mathfrak{z}_e$ in (5d)) and the force $\mathbf{f}_{e,t}$ can be written as

$$\ell \times \mathbf{f} = \begin{bmatrix} 0 & -\ell^z & \ell^y \\ \ell^z & 0 & -\ell^x \\ -\ell^y & \ell^x & 0 \end{bmatrix} \begin{bmatrix} \mathbf{f}^x \\ \mathbf{f}^y \\ \mathbf{f}^z \end{bmatrix} \tag{6a}$$

$$= \left[\overbrace{\begin{bmatrix} -\ell^z & \ell^y \end{bmatrix}}^{\mathbf{a}^x} \overbrace{\begin{bmatrix} \mathbf{f}^y \\ \mathbf{f}^z \end{bmatrix}}^{\mathbf{b}^x}, \overbrace{\begin{bmatrix} \ell^z & -\ell^x \end{bmatrix}}^{\mathbf{a}^y} \overbrace{\begin{bmatrix} \mathbf{f}^x \\ \mathbf{f}^z \end{bmatrix}}^{\mathbf{b}^y}, \overbrace{\begin{bmatrix} -\ell^y & \ell^x \end{bmatrix}}^{\mathbf{a}^z} \overbrace{\begin{bmatrix} \mathbf{f}^x \\ \mathbf{f}^y \end{bmatrix}}^{\mathbf{b}^z} \right] \tag{6b}$$

where the superscripts x, y, z reference to the components of the vectors $\ell, \mathbf{f} \in \mathbb{R}^{3\times 1}$, but then they also identify the vectors $\mathbf{a}^i, \mathbf{b}^i \in \mathbb{R}^{2\times 1}$ for $i \in \{x, y, z\}$, whose scalar product $\mathbf{a}^i \cdot \mathbf{b}^i$ is equivalent to the corresponding element of the cross product vector $(\ell \times \mathbf{f})^i$. Similarly, we notice that the nonconvexity in (5b) can be written as a scalar product between the timestep variable Δ_t and linear momentum $\mathbf{l}_t$, contact forces $\mathbf{f}_{e,t}$ and torque $\kappa_{e,t}$ variables. It means that all nonconvex constraints solely include equality constraints with bilinear terms.

Noticing that $\mathbf{a}^i \cdot \mathbf{b}^i = \frac{1}{4}\left\|\mathbf{a}^i + \mathbf{b}^i\right\|_2^2 - \frac{1}{4}\left\|\mathbf{a}^i - \mathbf{b}^i\right\|_2^2$, we reformulate all the bilinear expressions as differences of convex quadratic functions with known positive curvature, as was done in [26] and in the spirit of [44]. In other words, we can now decompose a bilinear expression with an indefinite curvature into quadratic terms with known curvature, which is key for the efficiency of our algorithm. To simplify the subsequent presentation, we define the following sets

Definition 1: Given a real vector space V, we define $\mathcal{Q}^+$ as the set of quadratic functions $V \to \mathbb{R}$ with a positive semi-definite Hessian matrix.

Definition 2: Given a real vector space V, the set $\mathcal{Q}^\pm$ is

$$\mathcal{Q}^\pm = \left\{ \mathfrak{B}\cdot : V \to \mathbb{R} \mid \mathfrak{B}(\mathbf{u}) = \chi(\mathbf{u}) - \zeta(\mathbf{u}) \text{ for } \chi, \zeta \in \mathcal{Q}^+ \right\} \tag{7}$$

where Figure 4 graphically illustrates this decomposition. In particular, the set $\mathcal{Q}^\pm$ is closed under scalar multiplication, addition and composition with affine functions,

$$\alpha(\bar{\mathbf{v}} \circ \mathbf{v}) + \beta(\bar{\mathbf{w}} \circ \mathbf{w}) \in \mathcal{Q}^\pm \tag{8}$$

for any $\alpha, \beta \in \mathbb{R}$, affine functions $\mathbf{v}(\cdot), \mathbf{w}(\cdot)$ and $\bar{\mathbf{v}}(\cdot), \bar{\mathbf{w}}(\cdot) \in \mathcal{Q}^\pm$. Consider for example Equation (5d) and assume for simplicity that $(\mathbf{R}_{e,t}^{x,y}\, \mathfrak{z}_{e,t}) \times \mathbf{f}_{e,t}$ is represented by the decomposition $\ell \times \mathbf{f}$, then each endeffector torque component becomes $\gamma_{e,t}^i = \mathbf{a}^i \cdot \mathbf{b}^i + (\mathbf{R}_{e,t}^z\, \tau_{e,t})^i$. The torque component $\tau_{e,t}$ could also be formulated as a difference of positive components $\tau = {}^+\tau - {}^-\tau$, where ${}^+\tau, {}^-\tau \geq 0$, as in [13] to embed them into the decomposition; however, this is not required. Then

$$\gamma_{e,t}^i = \overbrace{\underbrace{\left[\frac{1}{4}\left\|\mathbf{a}^i + \mathbf{b}^i\right\|_2^2\right]}_{\in \mathcal{Q}^+} - \underbrace{\left[\frac{1}{4}\left\|\mathbf{a}^i - \mathbf{b}^i\right\|_2^2\right]}_{\in \mathcal{Q}^+}}^{\in \mathcal{Q}^\pm} + (\mathbf{R}_{e,t}^z)^i\, \tau_{e,t} \tag{9}$$

In a similar manner, each endeffector torque component $\kappa_{e,t}^i$ (5c), can be decomposed parameterizing its cross product $(\mathbf{p}_e - \mathbf{r}) \times \mathbf{f}_e$ with vectors $\mathbf{c}^i$ and $\mathbf{d}^i$ as $\kappa_{e,t}^i = \mathbf{c}^i \cdot \mathbf{d}^i + \gamma_{e,t}^i$.

$$\kappa_{e,t}^i = \overbrace{\overbrace{\underbrace{\left[\frac{1}{4}\left\|\mathbf{c}^i + \mathbf{d}^i\right\|_2^2\right]}_{\in \mathcal{Q}^+} - \underbrace{\left[\frac{1}{4}\left\|\mathbf{c}^i - \mathbf{d}^i\right\|_2^2\right]}_{\in \mathcal{Q}^+}}^{\in \mathcal{Q}^\pm} + \overbrace{\gamma_{e,t}^i}^{\in \mathcal{Q}^\pm}}^{\in \mathcal{Q}^\pm} \tag{10}$$

where the vectors $\mathbf{c}^i, \mathbf{d}^i \in \mathbb{R}^{2\times 1}$ for $i \in \{x, y, z\}$ have been introduced in a similar fashion to Eq. (6b) to refer to the vectors whose scalar product $\mathbf{c}^i \cdot \mathbf{d}^i$ is equivalent to the corresponding component of the cross product $((\mathbf{p}_e - \mathbf{r}) \times \mathbf{f}_e)^i$. A similar analysis holds for each of the Cartesian components of the bilinear expressions within the dynamic constraints (5b), which can be decomposed into elements of $\mathcal{Q}^\pm$ as given by

$$\mathbf{l}_t^i\, \Delta_t = \frac{1}{4}\left\|\mathbf{l}_t^i + \Delta_t\right\|_2^2 - \frac{1}{4}\left\|\mathbf{l}_t^i - \Delta_t\right\|_2^2 \tag{11a}$$

$$\sum_{e \in e_{cnt}} \kappa_{e,t}^i\, \Delta_t = \frac{1}{4}\left\|\sum_{e \in e_{cnt}} \kappa_{e,t}^i + \Delta_t\right\|_2^2 - \frac{1}{4}\left\|\sum_{e \in e_{cnt}} \kappa_{e,t}^i - \Delta_t\right\|_2^2 \tag{11b}$$

$$\sum_{e \in e_{cnt}} \mathbf{f}_{e,t}^i\, \Delta_t = \frac{1}{4}\left\|\sum_{e \in e_{cnt}} \mathbf{f}_{e,t}^i + \Delta_t\right\|_2^2 - \frac{1}{4}\left\|\sum_{e \in e_{cnt}} \mathbf{f}_{e,t}^i - \Delta_t\right\|_2^2 \tag{11c}$$

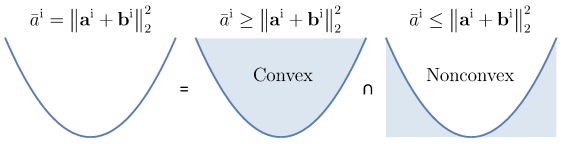
In the next section, we show how we can use this structure to approximate the problem using iterative convex relaxations.

B. Optimization with iterative convex relaxations

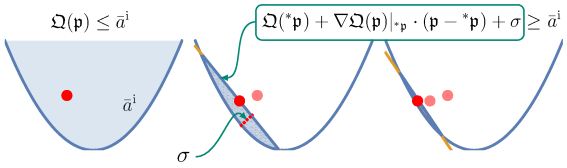
We now use the known curvature of the quadratic terms $\mathcal{Q}^+$ to build a convex approximation. We start by isolating the quadratic expressions into quadratic constraints by introducing scalar variables $\bar{a}^i, \bar{b}^i \in \mathbb{R}$. For example, Eq. (9) would become

$$\bar{a}^i = \left\|\mathbf{a}^i + \mathbf{b}^i\right\|_2^2, \quad \bar{b}^i = \left\|\mathbf{a}^i - \mathbf{b}^i\right\|_2^2 \tag{12a}$$

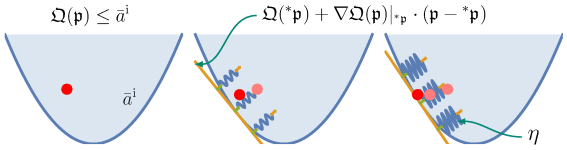
$$\gamma_{e,t}^i = \frac{1}{4}\left(\bar{a}^i - \bar{b}^i\right) + (\mathbf{R}_{e,t}^z)^i\, \tau_{e,t} \tag{12b}$$



(a) Nonconvex quadratic equality constraint as the intersection of convex and nonconvex quadratic inequality constraints. In this work, we use only the convex space of this constraint and a heuristic to guide solutions towards its boundary.



(b) Trust-region method: We first find a solution within the convex space $\mathfrak{Q}(\mathfrak{p}) \leq \bar{a}^i$ (our approximation variable $\bar{a}^i$ can take any value within the blue region). Then based on this solution, we iteratively build a trust-region that limits the search space to the boundaries. The parameter σ controls the distance between trust-region and quadratic constraint, thus the amount of constraint violation.



(c) Soft-constraint method: We first find a solution within the convex space $\mathfrak{Q}(\mathfrak{p}) \leq \bar{a}^i$ and based on this solution, we iteratively build a function underestimator, that allows us to include a cost that rewards selecting values close to it and thus close to the constraint boundary. Parameter η controls the desirability of selecting solutions close to the underestimator of the convex quadratic inequality constraint.

Fig. 5: Sequential approximation of quadratic expressions $\mathcal{Q}^+$ within its convex space using iterative convex relaxation methods.

where the introduction of the additional scalar variables $\bar{a}^i, \bar{b}^i$ renders the original equation (12b) linear and isolates the quadratic nonconvex expressions with known curvature into a pair of additional quadratic constraints (12a), whose very simple form will benefit the search of efficient convex approximations.

Figure 5a sketches the hyperplane defined by the nonconvex constraint (12a), conceived as the intersection of two inequalities, a convex $\bar{a}^i \geq \left\|\mathbf{a}^i + \mathbf{b}^i\right\|_2^2$ and a concave one $\bar{a}^i \leq \left\|\mathbf{a}^i + \mathbf{b}^i\right\|_2^2$. While it is difficult to search a solution in a high dimensional nonconvex space, it is easier to search within the space defined by the convex inequality and guide the optimization towards the constraint boundary, approaching in this way towards solutions with practical feasibility for the original nonconvex quadratic equality constraint.

To summarize, we systematically isolate all the quadratic expressions present in the optimization problem and replace them with new scalar optimization variables in order to render the original constraints linear. We then add simple equality constraints between the new variables and the quadratic terms. This allows us to move all the nonconvex elements of the problem into simpler terms in the form of quadratic equality constraints. We now propose two iterative methods based on SOC programs to deal with each of the quadratic equalities.

1) Trust-region method: In this approach, the main idea is to use a primal constraint to limit the convex search space to values close to the boundaries. In mathematical terms, the trust-region should constrain the problem to values of $\bar{a}^i$ near $\mathfrak{Q}(\mathfrak{p})$ (for simplicity of notation, we define $\mathfrak{p} = \mathbf{a}^i + \mathbf{b}^i$ and $\mathfrak{Q}(\cdot) = \|\cdot\|_2^2$). During the first iteration, an initial guess of the optimal problem values is obtained by searching over the entire relaxed convex search space. From there on, the trust-region is built based on the optimal problem values from the previous iteration ${}^*\mathfrak{p}$ and by reducing the desired allowed amount of constraint violation σ, as shown in Figure 5b.

Trust-Region Approximation of $\mathcal{Q}^+$ Expressions

In the case of $\mathcal{Q}^+$ expressions, thanks to the positive curvature of the constraint's hessian, a linear inequality constraint suffices to constrain the problem as desired.

$$\mathfrak{Q}(\mathfrak{p}) = \bar{a}^i \rightarrow \left\{ \mathfrak{Q}(^*\mathfrak{p}) + \nabla\mathfrak{Q}(\mathfrak{p})|_{^*\mathfrak{p}} \cdot (\mathfrak{p} - {}^*\mathfrak{p}) + \sigma \geq \bar{a}^i \right. \quad (13)$$

The linear constraint is built based on the optimal values of $\mathfrak{p}$ found in the previous iteration ${}^*\mathfrak{p}$ and σ is a positive threshold, big enough to provide a feasible interior to the intersection of the constraints, but also small enough so as to achieve the desired precision at convergence.

The benefits of constraining the problem in this way are twofold: firstly, we can easily refine the solution with values of $\mathfrak{p}$ around ${}^*\mathfrak{p}$ that satisfy the amount of desired constraint violation σ, and secondly, it provides a method to iteratively increase the approximation accuracy by reducing the value of σ, as required by convergence tolerances. We further note that if the hessian of this constraint were an indefinite matrix, this trust-region would lead to unbounded regions instead of constraining the problem as desired.

2) Soft-constraint method: Alternatively, a hard restriction of the search space could be replaced with a cost that biases the optimizer towards finding solutions close to the boundary of the constraint by pulling optimization variables towards a function underestimator, as shown in Figure 5c.

Soft-Constraint Approximation of $\mathcal{Q}^+$ Expressions

A cost heuristic is used to reward the selection of values for the variable $\bar{a}^i$ close to the function underestimator $(\mathfrak{Q}(^*\mathfrak{p}) + \nabla\mathfrak{Q}(\mathfrak{p})|_{^*\mathfrak{p}} \cdot (\mathfrak{p} - {}^*\mathfrak{p}))$, hyperplane that supports the function and was built based the optimal values of $\mathfrak{p}$ found in the previous iteration ${}^*\mathfrak{p}$.

$$\mathfrak{Q}(\mathfrak{p}) = \bar{a}^i \rightarrow \begin{cases} \mathfrak{Q}(\mathfrak{p}) \leq \bar{a}^i \\ \eta \left\| \mathfrak{Q}(^*\mathfrak{p}) + \nabla\mathfrak{Q}(\mathfrak{p})|_{^*\mathfrak{p}} \cdot (\mathfrak{p} - {}^*\mathfrak{p}) - \bar{a}^i \right\|_2^2 \end{cases} \quad (14)$$

η defines the desirability of selecting optimization values close to the underestimator, and thus enjoy practical feasibility for the nonconvex constraint.

Remark 2: As shown in Fig. 5, both methods iteratively approximate the problem as SOC programs, efficiently solvable with polynomial-time methods. In section sec. VI-B, we will

further discuss and compare the described methods.

C. Numerical optimization

In this section, we describe numerical aspects such as convergence criteria and algorithmic implementation details for both optimization problems.

1) Convergence criteria: The amount of constraint violation ε is used as the measure to decide upon convergence. It is defined as the supremum among the average errors of the state trajectory variables (15d), which are computed by comparing the values of the optimization variables ($\mathbf{r}_t$, $\mathbf{l}_t$, $\mathbf{k}_t$) that solve the approximate problem and the values obtained by integrating endeffector wrenches ($^{\text{seq}}\mathbf{r}_t$, $^{\text{seq}}\mathbf{l}_t$, $^{\text{seq}}\mathbf{k}_t$) that satisfy exactly all of the nonconvex constraints, as follows

$$^{\text{seq}}\mathbf{l}_t = \mathbf{l}_0 + \sum_{i=1}^{t}\left[m\mathbf{g} + \sum_{e\in e_{\text{cnt}}}\mathbf{f}_{e,i}\right]\Delta_i \tag{15a}$$

$$^{\text{seq}}\mathbf{r}_t = \mathbf{r}_0 + \frac{1}{m}\sum_{j=1}^{t}\left[\mathbf{l}_0 + \sum_{i=1}^{j}\left(m\mathbf{g} + \sum_{e\in e_{\text{cnt}}}\mathbf{f}_{e,i}\right)\Delta_i\right]\Delta_j \tag{15b}$$

$$^{\text{seq}}\mathbf{k}_t = \mathbf{k}_0 + \sum_{i=1}^{t}\left[\sum_{e\in e_{\text{cnt}}}(\mathbf{p}_{e,i} + \mathbf{R}^{x,y}_{e,i}\,\mathfrak{d}_{e,i} - {}^{\text{seq}}\mathbf{r}_i)\times\mathbf{f}_{e,i} + \mathbf{R}^{z}_{e,i}\,\tau_{e,i}\right]\Delta_i \tag{15c}$$

$$\varepsilon = \sup\left\{\underbrace{\sum_{t=1}^{N}\frac{\|\mathbf{r}_t - {}^{\text{seq}}\mathbf{r}_t\|_2^2}{N}}_{\varepsilon_{\mathbf{r}}}, \underbrace{\sum_{t=1}^{N}\frac{\|\mathbf{l}_t - {}^{\text{seq}}\mathbf{l}_t\|_2^2}{N}}_{\varepsilon_{\mathbf{l}}}, \underbrace{\sum_{t=1}^{N}\frac{\|\mathbf{k}_t - {}^{\text{seq}}\mathbf{k}_t\|_2^2}{N}}_{\varepsilon_{\mathbf{k}}}\right\} \tag{15d}$$

When the errors ε fall below a certain threshold for the constraint violation to be considered negligible for practical purposes, we consider that the algorithm has converged.

2) Algorithmic implementation details: To approximate the solution of problem (5), we iteratively solve an approximate problem (using an interior point solver for SOC programs based on [45]), where each nonconvex constraint (5b)-(5d) has been replaced by a convex approximation. At each iteration, we update the approximation (based on the optimal values of the previous iteration) and its parameters to reduce the constraint violation amount. The procedure is then repeated until convergence. For the trust-region method, the parameter σ is decreased using iteratively increasing powers of a value less than one, i.e. $\sigma \propto \nu^k$, where $\nu < 1.0$ and k denotes the iteration number. In a similar fashion, for the soft-constraint method, a value for the penalty parameter η is selected according to the desired precision to be achieved (typically within the range $[1e4, 1e6]$) and higher relative to other objectives, so that it is prioritized.

We also highlight that the formulation of torques $\gamma_{e,t}$ in Eq. (5d) separately of $\kappa_{e,t}$ in Eq. (5c) is required only when the torque limits constraint (5j) is used, as it depends on the contact wrench $\lambda_{e,t} = \left[\mathbf{f}_{e,t}^T \;\; \gamma_{e,t}^T\right]^T$. Otherwise, the torques $\gamma_{e,t}$ in Eq. (5d) can be directly embedded within the torque $\kappa_{e,t}$ in Eq. (5c), thus generating a problem of smaller size.

IV. Optimization of Contact Plans

In this section, we explain how contact locations can be optimized within problem (5) when they are considered optimization variables that belong to a given contact surface. We also describe an algorithm based on mixed-integer programming to efficiently select a sequence of terrain surfaces and contact locations consistent with the centroidal dynamics.

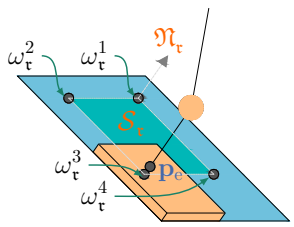


Fig. 6: The description of a terrain surface $\mathcal{S}_\mathfrak{r}$ comprises a set of coplanar corners $\omega^i_\mathfrak{r} \in \mathbb{R}^{3\times 1}$, where in this case $i \in [1,4]$. Out of them the following quantities can be computed: surface normal $\mathfrak{N}_\mathfrak{r} \in \mathbb{R}^{3\times 1}$, surface rotation $\mathfrak{R}(\mathcal{S}_\mathfrak{r}) \in \mathbb{R}^{3\times 3}$ (whose third column points in the direction of the surface normal), any surface point $^{\text{surf}}\omega_\mathfrak{r} = \omega^i_\mathfrak{r}$ and a membership constraint $\bar{\omega} \in \mathcal{U}(\mathcal{S}_\mathfrak{r}), \forall \bar{\omega} \in \mathcal{S}_\mathfrak{r}$, that simply defines the set of points $\bar{\omega} \in \mathbb{R}^{3\times 1}$ that lie on the terrain surface.

A. Membership of contact locations to terrain surfaces

Given a description of the terrain surface $\mathcal{S}_\mathfrak{r}$ (over which it is safe to make contact), a contact location can be optimized by including its membership constraint to surface $\mathcal{S}_\mathfrak{r}$ to the optimization problem. A terrain surface $\mathcal{S}_\mathfrak{r}$ (as defined in Fig. 6) is such that any contact point $\mathbf{p}_e$, selected from its interior, guarantees that the entire endeffector is in contact. The expression $\mathbf{p}_e \in \mathcal{U}(\mathcal{S}_\mathfrak{r})$ that constrains an endeffector position $\mathbf{p}_e$ to belong to surface $\mathcal{S}_\mathfrak{r}$ is defined as follows

$$\mathbf{p}_e \in \mathcal{U}(\mathcal{S}_\mathfrak{r}) \overset{\text{def}}{=} \begin{bmatrix} \Xi_\mathfrak{r} \\ \mathfrak{N}_\mathfrak{r} \\ -\mathfrak{N}_\mathfrak{r} \end{bmatrix} \mathbf{p}_e \leq \begin{bmatrix} \xi_\mathfrak{r} \\ \mathfrak{N}_\mathfrak{r} \cdot {}^{\text{surf}}\omega_\mathfrak{r} \\ -\mathfrak{N}_\mathfrak{r} \cdot {}^{\text{surf}}\omega_\mathfrak{r} \end{bmatrix} \tag{16}$$

Equation (16) defines a set of halfspaces, whose intersection constrains a contact point $\mathbf{p}_e$ to lie on a safe contact surface. For instance, $\Xi_\mathfrak{r}\,\mathbf{p}_e \leq \xi_\mathfrak{r}$ denote the halfspaces that define lateral limits of the terrain surface, while $\mathfrak{N}_\mathfrak{r} \cdot \mathbf{p}_e = \mathfrak{N}_\mathfrak{r} \cdot {}^{\text{surf}}\omega_\mathfrak{r}$ implies that the normal distance from the plane should be zero, i.e. the contact point has to lie on the terrain surface. Note that the row-size of the matrix $\Xi_\mathfrak{r}$ and vector $\xi_\mathfrak{r}$ depends on the number of halfspaces required to define the terrain region $\mathcal{S}_\mathfrak{r}$, while the column size of the matrix $\Xi_\mathfrak{r}$ is as $\mathbf{p}_e$, namely 3.

B. Dynamics-based contacts planning

Thus far, we have assumed that to solve problem (5) a set of terrain surfaces, from where contacts are selected, was given. Alternatively, a contact sequence could also be given by for example a contact planner such as [17], [46]. In the following, we propose a mixed-integer formulation that enables the selection of terrain surfaces and contact sequences based on a measure of dynamical robustness.

1) Terrain description and contact model: We now describe how a terrain is modeled and how contacts are selected within this description of the terrain using the notation of [20].

The terrain consists of a set of R convex, obstacle free regions $\mathcal{S}_\mathfrak{r}$ where $\mathfrak{r} \in \{1, \cdots, R\}$ and we consider the selection of

a sequence of M contact locations $\mathbf{p}_m$ where $m \in \{1, \cdots, M\}$. We note that the mapping between index m of the selected contact location $\mathbf{p}_m$ and, endeffector e and the range of timesteps t, in which endeffector location $\mathbf{p}_{e,t}$ is active, is predefined. For instance, we could optimize $M = 4$ contacts with $M/2$ contacts for each foot in a locomotion task, or we could optimize a larger number of contacts $M = 6$, where the 2 additional contacts are free slots to select hand contacts. Note that stance and flight timings can later be changed within the dynamics problem. Also $\mathfrak{r} = \varphi(e,t)$ maps e,t to surface $\mathfrak{r}$ chosen for contact m.

The matrix of binary variables $\mathcal{H} \in \{0,1\}^{(M-M_0)\times R}$ (indexed by contact $m \in \{1, \cdots, M - M_0\}$ and terrain surface $\mathfrak{r} \in \{1, \cdots, R\}$) defines the terrain surface $\mathcal{S}_\mathfrak{r}$, whose domain contains the contact location $\mathbf{p}_m$ (M_0 are contacts initially active and thus with a predefined pose). The model is defined as follows

$$\mathcal{H}_{m,\mathfrak{r}} \implies \mathbf{p}_m \in \mathcal{U}(\mathcal{S}_\mathfrak{r}) \tag{17a}$$

$$\sum_\mathfrak{r} \mathcal{H}_{m,\mathfrak{r}} \begin{cases} = 1, & \text{for feet contacts} \\ \leq 1, & \text{for hands contacts} \end{cases} \tag{17b}$$

$$1 - \sum_\mathfrak{r} \mathcal{H}_{m,\mathfrak{r}} \implies (\mathbf{f}_{e,t} = 0), \quad \text{for hands} \tag{17c}$$

$$\mathcal{H}_{m,\mathfrak{r}} \implies {}^{\text{cone}}\mathcal{F}_\mu \mathfrak{R}(\mathcal{S}_\mathfrak{r})\, \mathbf{f}_{e,t} \leq 0, \quad \text{friction cone} \tag{17d}$$

An element $\mathcal{H}_{m,\mathfrak{r}}$ being one implies the membership constraint $\mathbf{p}_m \in \mathcal{U}(\mathcal{S}_\mathfrak{r})$ as shown in Eq. (17a). Thus, $\mathcal{H}_{m,\mathfrak{r}}$ decides upon the terrain region from where a contact location can be selected. Integrality constraints (17b) enforce membership of a contact location to at most one terrain surface. When no contact region is selected (e.g. no hand contact), control variables such as contact forces should be inactive (Eq. (17c)). When a contact region is selected, local endeffector forces $\mathfrak{R}(\mathcal{S})^T \mathbf{f}_{e,t}$ must satisfy friction cone constraints, as in (17d). ${}^{\text{cone}}\mathcal{F}_\mu$ is a matrix function of μ such that its product with the local force, returns a vector of negative values.

2) Reachability constraints: Reachability constraints between footstep locations are selected based on kinematic reachability using linear inequalities such as in [47] for forward or lateral motions or, based on the intersection of SOC constraints [20] for more general settings. They can be described in a convex form using linear inequalities based on kinematic reachability such as in

$${}^{\min}\Delta\mathbf{p} \leq |\mathbf{p}_m - \mathbf{p}_{m-1}| \leq {}^{\max}\Delta\mathbf{p} \tag{18}$$

where two subsequent contacts are restricted to be within the bounds ${}^{\min}\Delta\mathbf{p}$ and ${}^{\max}\Delta\mathbf{p}$. Reachability constraints can also be described as in [20] using an intersection of SOC constraints

$$\sum_{\mathfrak{h}\in H} {}^{\text{sec}}\mathfrak{S}_{\mathfrak{h},m} = \sum_{\mathfrak{h}\in H} {}^{\text{sec}}\mathfrak{C}_{\mathfrak{h},m} = 1 \tag{19a}$$

$${}^{\text{sec}}\mathfrak{S}_{\mathfrak{h},m} \implies \begin{cases} {}^{\sin}\Theta_\mathfrak{h} \leq \theta_m \leq {}^{\sin}\Theta_{\mathfrak{h}+1} \\ {}^{\sin}\mathfrak{s}_m = {}^{\sin}\mathfrak{u}_\mathfrak{h}\, \theta_m + {}^{\sin}\mathfrak{v}_\mathfrak{h} \end{cases} \tag{19b}$$

$${}^{\text{sec}}\mathfrak{C}_{\mathfrak{h},m} \implies \begin{cases} {}^{\cos}\Theta_\mathfrak{h} \leq \theta_m \leq {}^{\cos}\Theta_{\mathfrak{h}+1} \\ {}^{\cos}\mathfrak{c}_m = {}^{\cos}\mathfrak{u}_\mathfrak{h}\, \theta_m + {}^{\cos}\mathfrak{v}_\mathfrak{h} \end{cases} \tag{19c}$$

$$\left\| \begin{bmatrix} \mathbf{p}^x_m \\ \mathbf{p}^y_m \end{bmatrix} - \left(\begin{bmatrix} \mathbf{p}^x_{m-1} \\ \mathbf{p}^y_{m-1} \end{bmatrix} + \begin{bmatrix} {}^{\cos}\mathfrak{c}_m & -{}^{\sin}\mathfrak{s}_m \\ {}^{\sin}\mathfrak{s}_m & {}^{\cos}\mathfrak{c}_m \end{bmatrix} \mathcal{P}_{1,2} \right) \right\| \leq \mathcal{D}_{1,2} \tag{19d}$$

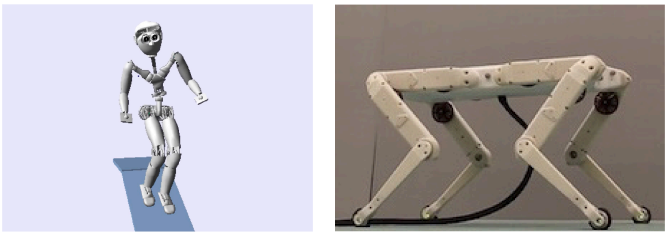
In the latter case e.g., a piecewise affine approximation of sine and cosine functions is used to model footsteps rotation $\theta_m \in \mathbb{R}$ in a convex form. The matrices of binary variables ${}^{\text{sec}}\mathfrak{S}, {}^{\text{sec}}\mathfrak{C} \in \{0,1\}^{H\times M_f}$ (indexed by affine approximation $\mathfrak{h} \in [1, H]$ and contact $m \in [1, M_f]$) are used to select the active affine approximation of sine or cosine $\mathfrak{h}$ for each footstep m. H denotes the number of affine functions used to approximate sine and cosine, and M_f the number of footstep contacts to be selected out of the total number of contacts M. As shown before, integrality constraints (Eq. (19a)) guarantee that only one approximation is active at each footstep m.

An element ${}^{\text{sec}}\mathfrak{S}_{\mathfrak{h},m}, {}^{\text{sec}}\mathfrak{C}_{\mathfrak{h},m}$ being one implies the activation of a single affine approximation for sine and cosine functions, as shown in (19b)-(19c). Each affine approximation is defined by a region of validity of the footstep rotation angle $\theta_m \in [{}^{\sin}\Theta_\mathfrak{h}, {}^{\sin}\Theta_{\mathfrak{h}+1}]$ (for sine) or $\theta_m \in [{}^{\cos}\Theta_\mathfrak{h}, {}^{\cos}\Theta_{\mathfrak{h}+1}]$ (for cosine) and, the corresponding affine approximation ${}^{\sin}\mathfrak{s}_m = {}^{\sin}\mathfrak{u}_\mathfrak{h}\, \theta_m + {}^{\sin}\mathfrak{v}_\mathfrak{h}$ (for sine) or ${}^{\cos}\mathfrak{c}_m = {}^{\cos}\mathfrak{u}_\mathfrak{h}\, \theta_m + {}^{\cos}\mathfrak{v}_\mathfrak{h}$ (for cosine), where ${}^{\sin}\mathfrak{u}_\mathfrak{h}, {}^{\sin}\mathfrak{v}_\mathfrak{h}, {}^{\cos}\mathfrak{u}_\mathfrak{h}, {}^{\cos}\mathfrak{v}_\mathfrak{h} \in \mathbb{R}$ are parameters that define slope and intercept values of each affine approximation. The footstep rotation angle θ_m, sine ${}^{\sin}\mathfrak{s}_m$ and cosine ${}^{\cos}\mathfrak{s}_m$ of this angle constitute optimization variables.

Finally, these variables are used to model the range of available positions for the next footstep (Eq. (19d)) based on the current footstep position and yaw angle as the intersection of two SOC constraints, parameterized by a pair of points $\mathcal{P}_{1,2} \in \mathbb{R}^{2\times 1}$ (located sideways of the footstep position $m-1$ and rotated by the yaw angle), and a pair of distances $\mathcal{D}_{1,2} \in \mathbb{R}$.

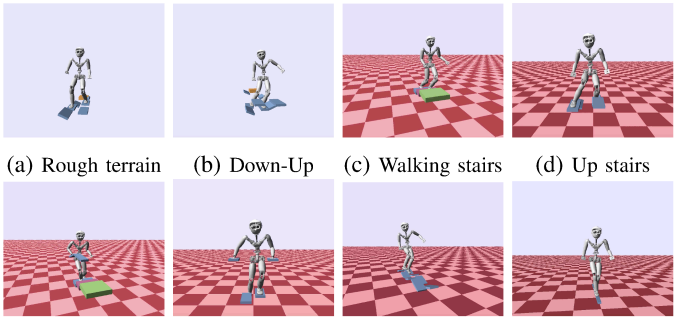
3) Dynamics model and objective function: To keep computational complexity low, in the mixed-integer approach to select contact sequences, we use a light version of problem (5), where we do not consider the endeffector torques $\gamma_{e,t}$ (in other words, a point contact model is assumed), we use a linear approximation of the friction cones and, either a centroidal momentum dynamics model with fixed or non-fixed timings. The objective function ϕ^{cnt}_t similarly to (5a) regularizes states and controls and also incorporates user-defined objectives.

4) Numerical optimization: To evaluate the performance of our method at synthesizing contact plans and selecting contact surfaces, we implement a custom mixed-integer solver able to solve a sequence of SOC programs. It relies on two functions to bound the optimal value of a given search space. The lower bound comes from a relaxation of the search space binary variables and the upper bound by any solution where the binary variables are actually binary. The rest of the constraints are treated using the iterative models previously described. The feasible search space is partitioned into convex sets and each partition bounded. The algorithm converges once global lower and upper bounds are close enough, otherwise, the partitions are refined and the search process is repeated. The implementation of the custom mixed-integer solver is based on a branch and bound method for global nonconvex optimization, as detailed in [48]. In simple scenarios, we use linear reachability constraints, and SOC constraints in more complex ones, as will be shown in Section V.



(a) Simulated humanoid robot (b) Real quadruped robot

Fig. 7: Robotic platforms used throughout the experimental section. Left, a simulated humanoid robot with 32 torque-controlled degrees of freedom is used in the SL simulation environment [49]. It has 7 degrees of freedom in each limb and 3 in the torso. Right, we show the quadruped robot 'Solo' with 8 torque-controlled joints [50].



(a) Rough terrain (b) Down-Up (c) Walking stairs (d) Up stairs

(e) Using hands (f) Up with hands (g) Tilted terrain (h) Narrow path

Fig. 8: Examples of time-optimized dynamic movement plans.

Cost	Functional Form	Scaling Order
CoM terminal cost	$\mathfrak{Q}(\mathbf{r}_N - (\mathbf{r}_0 + \Delta\mathbf{r}))$	$1e+4$
Time regularization	$\sum_t \mathfrak{Q}(\Delta_t - {}^0\Delta_t)$	$1e+3$
Momenta terminal cost	$\mathfrak{Q}(\mathbf{h}_N)$	$1e+2$
Endeffector consensus cost	$\sum_t \mathfrak{Q}(\mathbf{p}_{e,t} - {}^{kin}\mathbf{p}_{e,t})$	$1e+0$
Momenta consensus cost	$\sum_t \mathfrak{Q}(\mathbf{h}_t - {}^{kin}\mathbf{h}_t)$	$1e+0$
Momenta rate cost	$\sum_t \mathfrak{Q}(\dot{\mathbf{h}}_t)$	$1e-1$
Momenta running cost	$\sum_t \mathfrak{Q}(\mathbf{h}_t)$	$1e-2$
Force running cost	$\sum_t \mathfrak{Q}(\mathbf{f}_{e,t})$	$1e-3$
Torque running cost	$\sum_t \mathfrak{Q}(\tau_{e,t})$	$1e-3$

TABLE I: Example composition of the main components of the cost $\sum_t(\phi_t^{dyn} + \Phi_t^{dyn})$ used to synthesize the walking motion of Figure 8d.

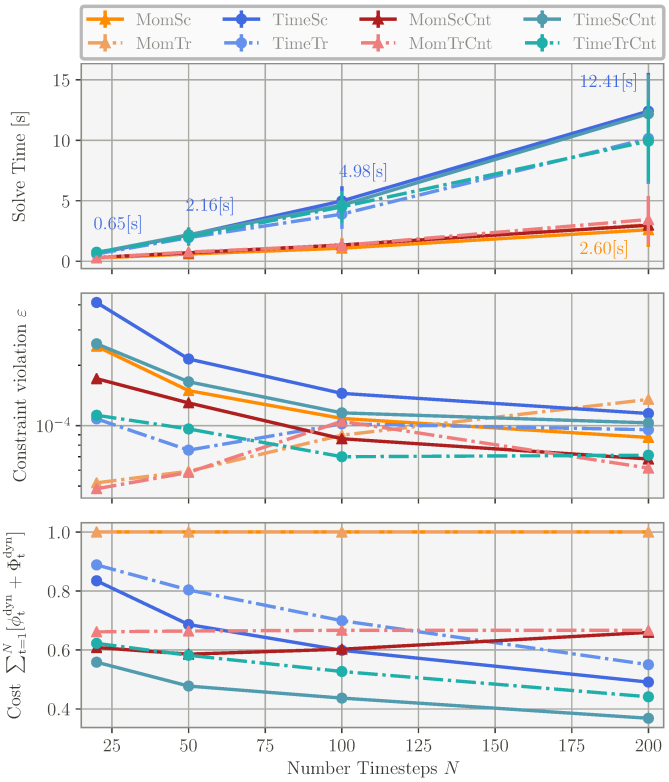


Fig. 9: *Top:* Roughly linear-time complexity of movement plans within the shown range of timesteps N: with or without time optimization $Time - Mom$, using soft-constraint or trust-region heuristics $Sc - Tr$, and with or without optimization of contact locations Cnt. *Center:* Corresponding normalized convergence errors or amount of constraint violation ε as given by (15d) and *Bottom:* numerical relative cost reduction of motions optimized including time and/or contact locations with respect to motions using fixed contacts and timings. Each datapoint averages information from 8 experiments (shown in Fig. 8) optimized using the same objective function but different number of timesteps N and heuristics.

V. Experimental Results

In this section, we show experimental results about the optimization of contact and movement plans using the algorithms previously described. We have tested them in several challenging multi-contact scenarios using simulated humanoid and quadruped robots and a real quadruped robot (Fig. 7a). The resulting motions are visible in the accompanying video.

A. On the optimization of movement plans

In this section, we analyze solutions of problem (5) in terms of convergence to feasibility (measured by the amount of constraint violation ε of the solution) and time complexity to converge to the desired feasibility threshold. We also present results regarding the qualitative improvement of motions that include time and/or contact locations in the optimization. Finally, we will show how full-body motions can be optimized using a kino-dynamic approach, how actuation limits can be included in the dynamics optimization, and the tracking performance of time-optimized movement plans.

1) Convergence to feasibility and time complexity: To analyze convergence properties and computational complexity of the algorithm, we use a set of 8 optimized motions (shown in Fig. 8) to gather statistics about its performance. Table I shows a typical cost function and the relative importance of the weighted costs used to optimize a motion. In Fig. 9, we present statistics about time complexity, convergence to feasibility, and relative cost reduction when using the same objective function but a different number of discretization timesteps and algorithmic settings. In particular, we look at what happens when the optimization includes or not time as an optimization variable *Time* vs. *Mom*, includes or not optimization of contact locations *Cnt*, uses the soft-constraint or trust-region relaxation approaches *Sc* vs. *Tr*. As an example, *MomSc* refers to a motion optimized with fixed timings, fixed contacts, and using the soft-constraint heuristic, while *TimeTrCnt* refers to a motion optimized including timings, contact locations, and using the trust-region heuristic.

First of all, in the center plot, we show the amount of

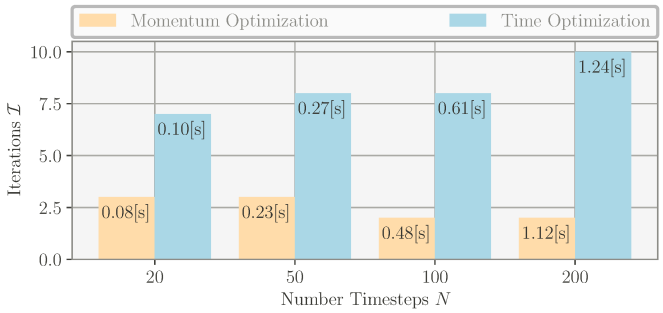


Fig. 10: Average number of iterations required to solve an optimization problem with or without time optimization for different number of discretization timesteps and, average time to solve each iteration. Each datapoint is based on 32 experiments, with different heuristic $Sc-Tr$ and with our without optimization of contacts Cnt.

constraint violation ε of the optimized solutions, as measured by (15d). We note that the algorithm converges when ε or its reduction from one iteration to another $\varepsilon_{\mathcal{I}} - \varepsilon_{\mathcal{I}-1}$ fall below a desired threshold (typically in the order of $1e-4$) and, as visible on the plot, our method converges in all experiments to the desired feasibility thresholds in all settings.

On the top, we show statistics about the time-complexity of the algorithm for convergence to the desired feasibility thresholds; in particular, this shows evidence of linear complexity in the number of timesteps for momentum and time optimization problems. We notice that for fixed-time optimization problems, neither heuristics nor the optimization of contact locations affect the solving time performance. Similar behavior can be seen for time optimization problems with the difference that the trust-regions are slightly faster than the soft-constraints.

Finally, on the bottom plot, we show numerically the relative reduction of the cost when optimizing time and contact locations. In orange tones, we see the reference normalized costs of momentum optimization problems using fixed contact locations and timings for trust-region and soft-constraint heuristics, namely *MomSc* and *TimeSc*. As expected both achieve a similar minimum and have thus the same normalized cost of one. As shown above, considering contact locations as optimization variables (in the form of linear constraints and over a given terrain surface) has minimum impact on solving time performance, yet it significantly reduces the objective value (between 35 and 40 percent) because this degree of freedom allows the optimizer to select motions with lower momentum values, e.g. motions with less lateral sway of the CoM (see *MomScCnt* and *MomTrCnt* in red tones).

The effect of time optimization on the objective value is dependent on the problem time horizon (or the number of timesteps N). For simplicity, we can assume that the value of one timestep is 0.1 seconds (which is the discretization time we use for fixed time optimization), and thus the horizontal axis spans between 2 and 20 seconds. For instance, in problems with short-time horizons such as those at the leftmost side, the cost difference between motions that consider or not time as an optimization variable is modest, but as the look-ahead horizon increases (right side) time optimization becomes a powerful way of shaping the motion to achieve lower costs. We notice that in this case the soft-constraint heuristic (*timeSc* and *timeScCnt*) finds in average slightly lower local minima than the trust-region heuristic (*timeTr* and *timeTrCnt*).

In Fig. 10, we show the average number of iterations required to solve a momentum or time optimization problem for a varying number of timesteps, as well as the average time required to solve each of these iterations. For instance, momentum optimization problems require 2-3 iterations, while time optimization problems 7-10. However, the difference in solving times of one iteration is small, e.g. for a time horizon of 2 seconds ($N = 20$) the solving times are 80 and 100 [ms] for momentum and time optimization problems respectively. This suggests that the approach could be used in a receding horizon setting. In such a setting, the optimizer could be warm-started from the previous solution to significantly increase resolution time (typically one would only need to solve one iteration of the problem for a short look-ahead horizon).

2) Qualitative improvement of solutions: Here, we discuss qualitative results that cannot be described from the statistical analysis above. We, therefore, restrict our analysis to specific instances of the problem. In Fig. 11 we show time-optimal results for a walking up tilted stairs motion traversed with two different values of the friction coefficient μ. In the first case ($\mu = 0.35$), the tendency is to increase the value of timestep variables Δ_t during double supports to have enough time to slowly accelerate the CoM while respecting physical constraints, resembling statically stable motions. In an environment with flat surfaces, the same approach would be valid even if the friction coefficient is further reduced (e.g. $\mu = 0.25$). However, in a terrain with tilted surfaces, such a strategy is not viable. In such a setting, even the fixed-time version of our algorithm cannot find a dynamically feasible solution. Yet, our time optimization approach is able to find a solution, whose main strategy is to quickly traverse the tilted surfaces to get to the uppermost flat contact surfaces. During this phase, lateral contact forces are exploited to the limit, and then a similar strategy to the previous case is found.

In Fig. 12, we show a walking upstairs motion using hand contacts. In our experiments, in such multi-contact scenarios, time optimization does not significantly change motion timings, as can be seen in the bottom plot (that graphically illustrates endeffector activations e_{cnt}) by comparing the timings of a fixed-time optimization problem (Mom) and those of a time optimization problem (Time). However, optimizing contact locations allows us to find motions with less CoM sway. This is visible, for example, in the CoM trajectories for a momentum optimization without optimization of contact locations *MomSc* or even a time optimization without optimization of contacts *TimeSc* and a momentum or time optimization that includes optimization of contact locations such as *MomScCnt* and *TimeScCnt* respectively. These motions are more energetically efficient and arguably easier to control with only a small additional computational cost in the optimization. Note that the top plot in Fig. 12 is a top view of the walking upstairs movement using hands, not to be confused with a planar motion.

3) Kino-dynamic full-body optimization: In this section, we show how our algorithm can be used in the kino-dynamic

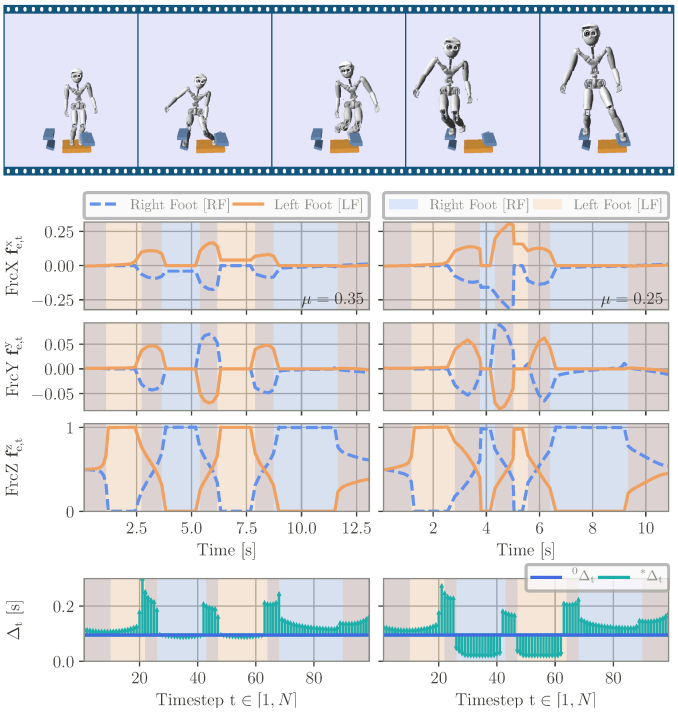


Fig. 11: Comparison of optimal normalized endeffector forces and timing results for two different values of friction coefficient μ. Timings ${}^0\Delta_t$ are the initial ones and ${}^*\Delta_t$ the final optimized ones.

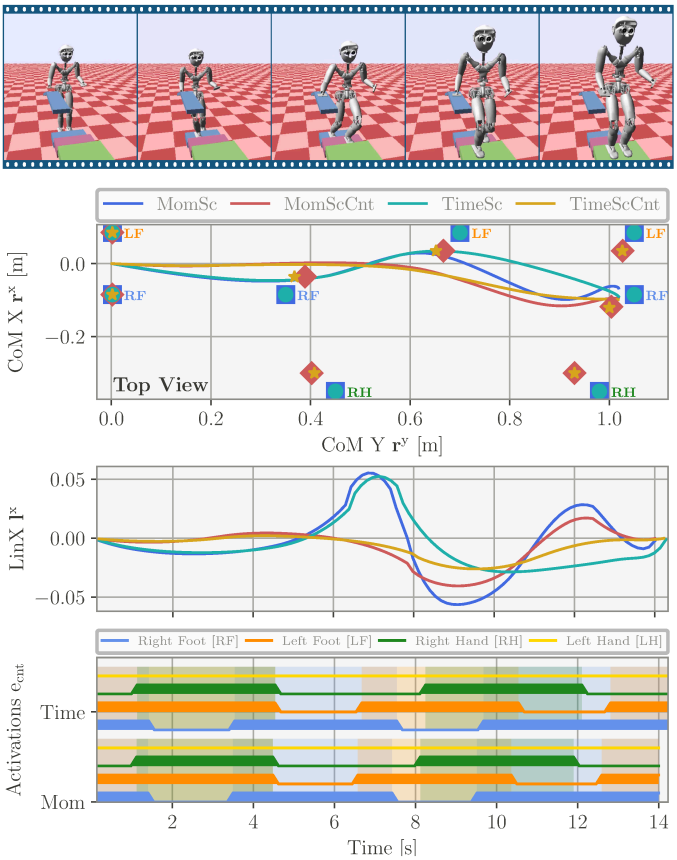


Fig. 12: Comparison between CoM and normalized linear momentum in the lateral direction for a walking upstairs motion using hands. The squares, circles, diamonds, and stars show the endeffector locations $\mathbf{p}_{e,t}$ optimized under different settings, as shown in the legend. Bottom plot shows the contact activation of endeffectors over the time horizon for momentum (Mom) and time (Time) optimization problems (low value is inactive and high value is active).

approach described in Section II, and illustrated in Figure 3, to generate whole-body time-optimal motions.

First, we use the climbing uneven stairs motion depicted in Fig. 8d to illustrate algorithmic convergence of our method to kino-dynamic consistency. In Fig. 13, we graphically compare (on the top 3 plots) kinematic ${}^{\text{kin}}\mathbf{h}$ and dynamic momentum trajectories ${}^{\text{dyn}}\mathbf{h}$ at the end of each dynamics optimization. We use dark colors to show dynamic trajectories ${}^{\text{dyn}}\mathbf{h}$, and the same, but light color, for kinematic ones ${}^{\text{kin}}\mathbf{h}$. Solid lines correspond to motions optimized using soft-constraints and dashed lines to motions optimized using trust-regions. It can be seen from the plots that they qualitatively converge to similar solutions, as it is difficult to distinguish them from each other.

On the bottom plot, we show how quantitatively the norms $\varepsilon_\mathbf{l}$ and $\varepsilon_\mathbf{k}$, that compare momentum trajectories obtained from optimal controls and the momentum trajectory variables that track desired kinematic momentum trajectories, decrease until convergence at each kino-dynamic iteration. Note that the first dynamics optimization (shown in red) takes the longest to converge and that trajectories optimized in subsequent iterations without using any information from previous ones converge faster (see e.g. how solid and dashed lines from the first iteration compare to those at subsequent iterations). In practice, however, by warm-starting the heuristics of dynamic optimizations with the results and information of previous iterations, the optimization problems can be solved much faster and with fewer iterations, as shown in dotted trajectories. Despite that at each iteration kinematic and dynamic momentum trajectories match, in practice, we use at least two iterations to converge to a motion easily executable on a physical simulator.

Note as well how linear momentum converges fast to high levels of precision, while angular momentum does it only to modest levels. See, for example, how solid and dashed lines achieve in 4 iterations the required precision for linear momentum errors $\varepsilon_\mathbf{l}$, while it takes around 8 for angular momentum errors $\varepsilon_\mathbf{k}$. This is due to the fact that on the one hand angular momentum depends on the CoM and can only achieve a higher precision once this variable has converged, and on the other hand due to the fact that given a CoM trajectory, angular momentum can be further optimized along it by exploiting the control degrees of freedom left.

Finally, we present results on a simulated quadruped robot, where we show in Fig. 14 the kino-dynamic trajectories of a galloping motion, very difficult to optimize due to the presence of simultaneous flight phases for all endeffectors, where only gravity is acting on the system. Despite this challenge, kino-dynamic trajectories converge qualitatively well thanks to the exploitation of optimal timing for all available endeffector forces.

4) Execution of movement plans: In this section, we show that optimal motion plans optimized in the previous section using a kino-dynamic approach can be executed in a physical simulator using the architecture described in Fig. 1.

In Fig. 15, we first show the tracking of an optimized

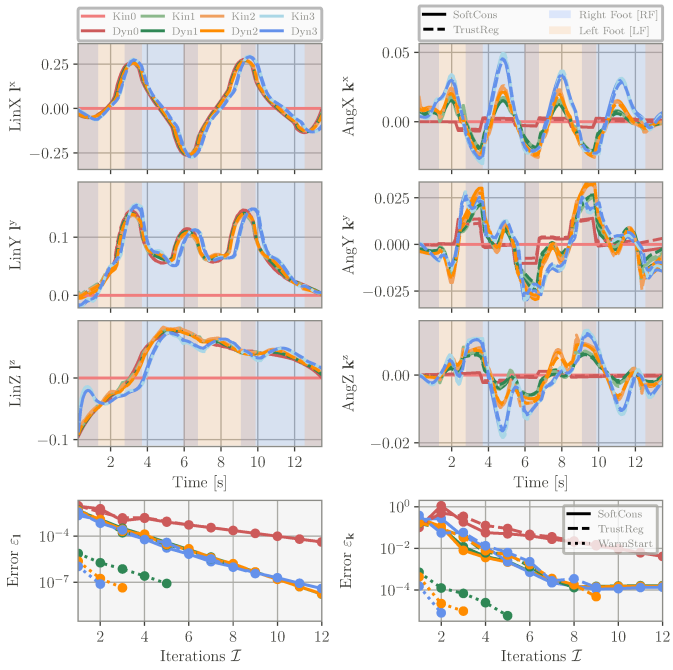


Fig. 13: This figure shows convergence to feasibility of each dynamics optimization for three kino-dynamic iterations. We compare desired kinematic momentum trajectories *Kin* and dynamic momentum trajectories *Dyn* (computed out of optimal controls) at the end of each dynamic optimization. Bottom plots (left for linear momentum and right for angular momentum) show how errors ε_l and ε_k decrease until convergence along each kino-dynamic iteration. Momentum values are normalized by robot mass. Vertical colored bars show the activation of each endeffector over time.

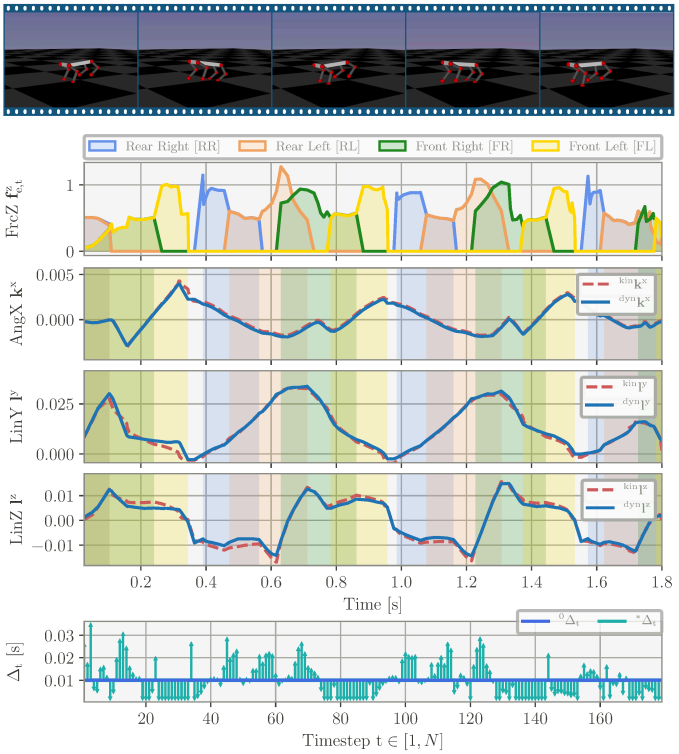


Fig. 14: Kino-dynamic results for the optimization of a galloping motion that includes simultaneous flight phases for all endeffectors.

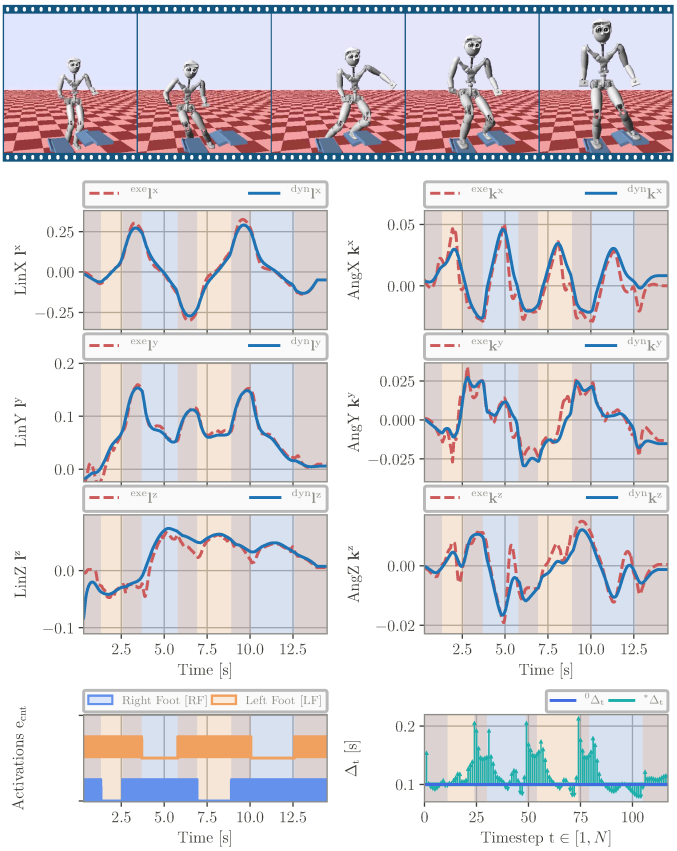


Fig. 15: Tracking of desired momentum trajectories for the climbing up stairs motion (shown in Fig. 8d) using time optimization.

movement plan for a robot climbing uneven stairs using inverse dynamics controllers [39] that realize closed-loop behaviors based on risk-sensitive feedback design [51] that explicitly considers process and measurement noise [52] to compute time-varying feedback gains. In our experience, such a controller leads to overall lower impedance gains in comparison to typical LQR design, which is beneficial to increase compliance at contact with an environment that differs from the ideal scenario used for planning. Note that such a feedback controller is important in this case, as the kino-dynamic optimizer is not used in a receding horizon fashion. The top three plots show the optimized momentum trajectories (${}^{\text{dyn}}\mathbf{l}, {}^{\text{dyn}}\mathbf{k}$ in blue) as well as their tracking (${}^{\text{exe}}\mathbf{l}, {}^{\text{exe}}\mathbf{k}$ in red). At the bottom left corner, endeffectors activation over time e_{cnt} are shown, as given by the optimal timings ${}^{*}\Delta_t$ at the bottom right corner.

In Fig. 16, we show that actuation limits are not always satisfied if they are not explicitly considered. For instance, on the left column, we analyze torques in the climbing up stairs motion (Fig. 8d). Here, the knee flexion-extension (KFE) joint torque exceeds its limit by 30 Nm (bottom-left in blue). To enforce torque limits, the solution of the kinematics problem $({}^{*}\mathbf{q}, {}^{*}\dot{\mathbf{q}}, {}^{*}\ddot{\mathbf{q}})$ is used to build a linear approximation of Eq. (2b) along the motion trajectory (used to build the constraint of Eq. (5j)). This constraint relates contact wrenches $\lambda_{\mathrm{e}}(\mathrm{t}) = \begin{bmatrix}\gamma_{\mathrm{e,t}} & \mathbf{f}_{\mathrm{e,t}}\end{bmatrix} \forall \mathrm{e} \in \mathrm{e}_{\text{cnt}}$ and torques $\Lambda(\mathrm{t})$, making it possible to adapt contact wrenches to satisfy torque limits. The

top three left plots show how the right foot's wrench can be adapted from a motion that does not satisfy torque limits (NoTrqLimPlan in blue) to one that does (TrqLimPlan in orange). Further, in green, the torque limit satisfied during execution is shown. Another way to satisfy torque limits is by redistribution of contact forces among the available endeffectors (Fig. 16 right). In this case, timesteps were kept constant, and the optimizer distributed contact forces in such a way that the left leg is supported by the left hand in order to synthesize a motion within the leg actuation limits. Joint torques plotted correspond to those degrees of freedom of left limbs that control the endeffector position. Furthermore, Fig. 17 shows the effect of torque limits on solve time performance.

Remark 3: While we only demonstrated the ability of our approach to include joint actuation limits in the dynamic optimization problem, it would also be straightforward to add such limits in the kinematic optimization problem. Indeed, it would be possible to add linear joint acceleration constraints using Eq. (2b) and the solution of the dynamic optimization problem to approximate the contact forces.

Finally, Fig. 18 compares the ability of the algorithm at synthesizing a dynamically feasible solution under different initial and final conditions. Initial conditions include varying CoM velocities in the horizontal plane and distinct contact supports (one or two feet), while the final condition is a contact configuration as the initial one (single or double support). A solution is colored in orange if, after one step, a motion trajectory with momentum values under a small threshold has not been found. The experiment suggests that optimal timings can significantly extend the regions where a feasible dynamical solution is attainable, under given physical conditions and objective function, as well as that timing adaptation is important beyond known results for flat-ground walking [53].

B. On the optimization of contact plans

This subsection discusses results on the surface selection and contacts planning algorithm using a mixed-integer program that makes use of a dynamics model to measure the quality of the motion induced by the selected contacts plan.

Figure 19 shows a schematic of the experiment setup, average timing results, and a comparison between cost decrease and solving time increase for each iteration of the problem internally solved. In the experiment a robot traverses an uneven terrain from the initial stepping stone (in orange) to the desired position forward using the desired number of contacts M. Further, the number of terrain stepping stones is adapted as shown in the statistics table on the number of regions axis. On the figure's top, the mean and one standard deviation of solving times are shown for several configurations of surfaces and number of contacts to optimize. Note how short contact plans can be quickly solved, while longer ones require more computational effort. In those cases, more efficient techniques for contact planning can be used [17]. For example, a predictive neural network could be used to speed up the evaluation of dynamically feasible contact sequences as in [54].

Finally, Fig. 19 (bottom-left) shows the cost evolution of a contacts optimization problem ($\sum_t \phi_t^{\text{cnt}}$ in blue) as well as the

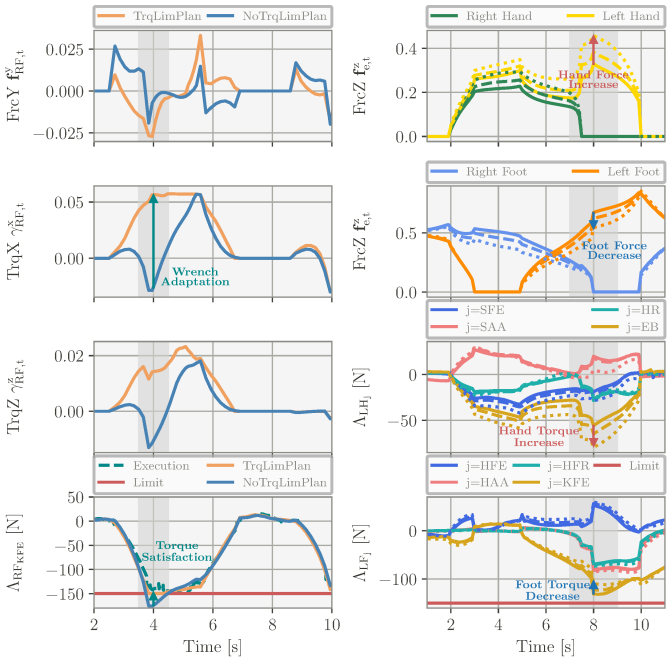


Fig. 16: Satisfaction of actuation limits: *To the left* (walking motion Fig. 8d), it can be achieved by adapting endeffector wrenches or timings; *To the right* (multi-contact fixed-time motion Fig. 8f), it can happen by redistribution of contact forces among active endeffectors.

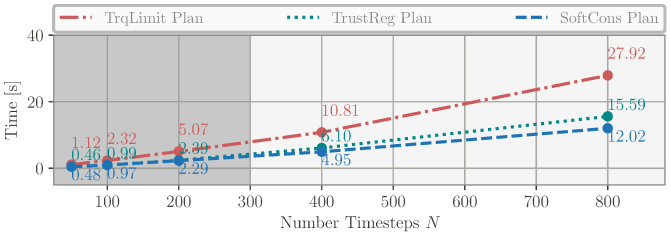


Fig. 17: Effect of actuation limit constraints on solving time of fixed-time optimization problems for different number of discretization timesteps. Results shown correspond to a walking down and up motion (Fig. 8b) using soft-constraints for torque limits.

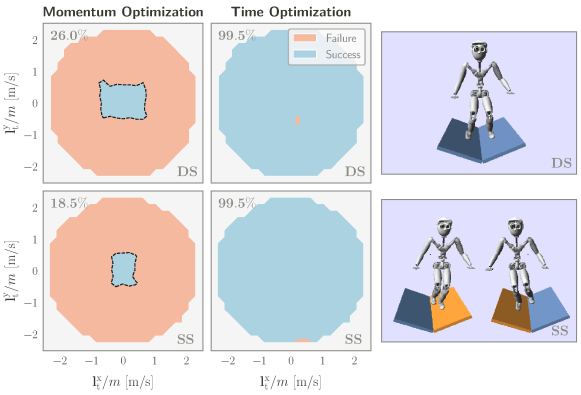


Fig. 18: Comparison of the regions where a dynamically feasible solution is attainable for single and double support experiments using fixed and optimal timings.

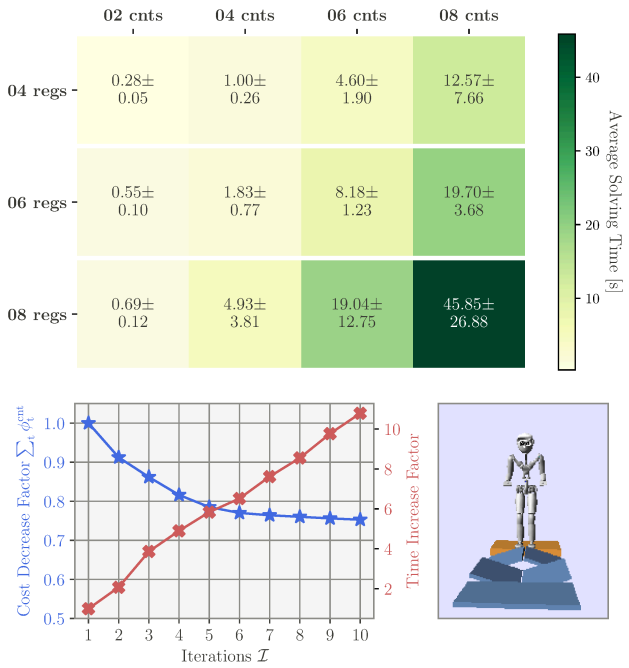


Fig. 19: Statistics about solving time for a contacts planning problem under different number of stepping regions and horizon of the number of contacts. It further compares the cost improvement and increment in solving time for different number of iterations $\mathcal{I}$.

time required to solved it (in red) as a function of the number of iterations $\mathcal{I}$ used to approximate the dynamic constraints. These values have been normalized by the values corresponding to $\mathcal{I} = 1$, such that both curves depict the cost decrease and time increase factors relative to those that use only one iteration. Notice how initially two additional iterations ($\mathcal{I} = 3$) reduce the cost by $\sim$15% while increasing the solving time by a factor 4. Towards the end, however, an additional iteration increases the solving time linearly but reduces the cost only minimally. This suggests that solving the problem to high-precision optimality (e.g. $\mathcal{I} = 10$) is impractical because of the large required solving time; however, a sub-optimal solution (e.g. $\mathcal{I} = 1$) is reasonable and can provide a good initialization contact plan for the motion optimization. The functional form of the cost function ϕ_t^{cnt} and importance weights are defined similarly to Table I.

C. Real robot experiments

This section presents the execution of kino-dynamic motion plans on our quadruped Solo [50]. Our main goal is to demonstrate that these plans are of sufficient quality to be executed on a real robot using only an instantaneous feedback controller and no re-planning. We use a passivity-based controller to track the optimized motions. The controller tracks desired CoM, angular momentum, base orientation, feet trajectories and also uses the desired feedforward centroidal wrench from the planner. This controller is described in detail in [50].

We consider three different scenarios to show the capability of the planner to generate feasible motions. In the first scenario, we provide the kino-dynamic planner with a periodic sequence of contact points to generate a trotting motion. In the second scenario, we consider a jumping motion with a flight phase. Finally, in the third scenario, we present a motion that combines a nontrivial sequence of contacts and a jumping motion. In all scenarios, we use the approach presented in Section II and III to generate kino-dynamically feasible motions. Note that for all the experiments we iterated only once between kinematic and dynamic optimizers. Note also that some of the motions presented here are the same motions used in [50] to evaluate the control law. We reproduce them here for completeness and focus our analysis on the motion plans not discussed in [50].

1) Scenario 1, trot: In the first scenario, we give a periodic contact sequence to the planner, where diagonal feet move forward as much as a step length in a specified time (Fig. 20, top row). Since the robot does not have the abduction/adduction hip joint, it is very important that the planner generates stable motions taking into account the robot full dynamics and that can be tracked by the controller without step adjustment. In our experiments, we noticed the importance of having fully consistent motion plans (and not solely centroidal dynamic motions), especially during contact transitions. Furthermore, it was also important to have a feedback controller explicitly tracking the desired centroidal wrench and feet trajectory. We were able to successfully execute trotting motions at various speeds. Moreover, in order to test the sensitivity of the motion plans to moderate environmental uncertainty, we planned a flat ground trot and successfully executed it on a seesaw. This result suggests that the optimized motions are sensitive neither to model mismatch nor small environmental changes. It is particularly interesting to note that we were able to execute rather long motions of around 10 [s] without re-planning.

2) Scenario 2, jump: To show the capability of the planner to generate highly dynamic motions, in this scenario we provide the planner with contact sequences with a flight phase. First, we implemented a jump in place (Fig. 21, top row), where the robot only needed to generate vertical thrust. In this scenario, the robot was able to jump 65 cm, while the robot's height in its natural standing phase is 24 cm. The generated plan is good enough such that the feedback controller is able to track the desired linear momentum in the vertical direction and realize the desired jump in place. We then implemented a forward jump on an 18 cm box (Fig. 21, bottom row). In this case, the planner needs to generate linear momentum in both vertical and horizontal directions to jump 60 cm forward and around 30 cm upward at the apex of the flight phase while ensuring that the generated angular momentum at take-off enables landing with the proper orientation.

3) Scenario 3, step and jump on obstacle: In this scenario, we present a motion that is a combination of transition between different multi-contact sequences, and a flight phase for jumping on an obstacle (Fig. 22). Here, our main goal is to showcase the capability of the planner in generating highly constrained multi-contact motion together with a highly dynamic motion. To step on the obstacle, the planner exploits the high range of motion of the robot hip joint and step on the obstacle without the need to change the base orientation to avoid collision of the front legs with the obstacle. Then,

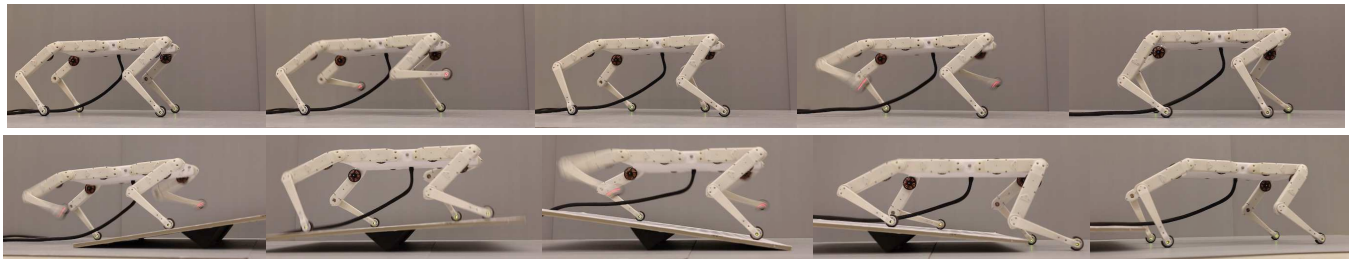


Fig. 20: Snapshots of the experiments in scenario 1; top) trot on flat surface, bottom) trot on seesaw

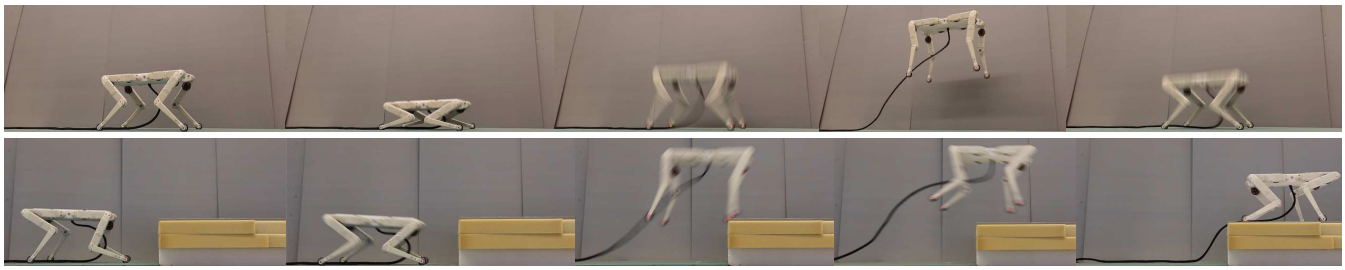


Fig. 21: Snapshots of the experiments in scenario 2; top) jump in place, bottom) jump on a box

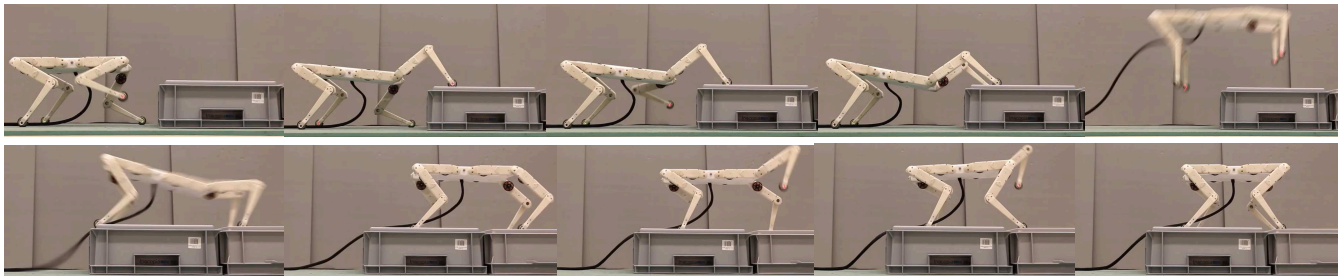


Fig. 22: Snapshots of the experiments in scenario 3; step and jump on an obstacle

through generating enough thrust on a non-coplanar set of contact points and in a nontrivial endeffectors configuration, the robot jumps on top of the obstacle. Finally, through another multi-contact set of changes in contact configuration, it brings back the joint configuration to the default one. This experiment scenario further illustrates the versatility of our optimizer to generate motions in complex environments.

VI. Discussion

A. Time and computational complexity

In general, finding a solution to the dense version of any of the convex approximations we solve, requires a polynomial time algorithm (of order $\mathcal{O}(\nu^{\frac{1}{2}}[\nu+\iota]\iota^2) \approx \mathcal{O}(\nu^{\frac{3}{2}})$, ν being the number of quadratic constraints and ι its size) [55]. However, within the problem size ranges of interest to us and thanks to the exploited problem sparsity patterns (e.g. due to time indexing), we observe (Fig. 9) that the problem has approximately linear time complexity. It is possible to note this linear tendency for both momentum and time optimization problems, despite their different rates of growth due to distinct problem sizes and even problems that consider actuation limits show this linear tendency (Fig. 17).

When considering torque limits the doubled computational effort due to the addition of 2nN inequality constraints for a problem with N timesteps and robot with n joints (≈ 32 in our case) can be reduced by considering only the weakest joints or only those involved in the motion. All in all, computation times are still lower than the planned horizon, making it possible to run the algorithm online (for example the next plan can be computed, while the current one is being executed).

B. On limitations and comparison of the approximations

Problem (5) is nonconvex and thus hard to solve. The proposed heuristics lighten to some extent the effort required to find a solution by searching for an approximate one within the convex space of the problem. This however comes with certain limitations. For instance, when using trust-regions, they might be inappropriately built leading to non-optimal solutions or even unsuitably initialized which could render the interior of the convex cone empty leading to primal infeasibility. For the soft-constraint method, the difficulty lies in finding an appropriate trade-off between two competing objectives: the amount of constraint violation and problem conditioning. An adaptive solution that iteratively reduces the value of the allowed amount of constraint violation σ works well for the trust-region heuristic, though care is required to slowly converge from the relaxed to the approximate problem without rendering the problem infeasible due to excessive reduction

of σ. For the soft-constraint method, a value high enough to prioritize the soft-constraint over the rest of the cost terms works well.

We have used both methods to synthesize a relatively high number of motions, so as to be able to successfully train a neural network [54]. From this experience, we highlight that both methods work equally well. However, we would like to remark on two cases where one would be more appropriate than the other. The first case would be when a certificate of optimality or infeasibility matters, e.g. to compute a viable set to be used as a terminal set constraint. In this case, the trust-region method is more appropriate as the slack or degree of constraint violation is controlled using a primal constraint and the certificate is valid for the given precision. The second case would be when the solver is to be warm-started not from information from previous iterations, but using a predictive model (e.g. a neural network). In this case, the soft-constraint method would not run into the risk of infeasibility due to an invalid initialization, making it a more appropriate approach to handle this case.

Notice that a single timeline was used to parameterize and optimize motions in eq. (5). However, this might be a limitation for more general and complex motions that require an independent timeline for each endeffector. Finally, notice that while the method is very general in nature and works well to solve problem (5), it is the case, as with any other nonlinear optimization method such as sequential quadratic programming [56], that it might not be appropriate or fail with other problem instances.

C. Stability of the computed motions

Our method generates dynamically feasible motions that satisfy general contact stability criteria such as [43]. If the final position of the robot has zero velocity, then we are guaranteed that the motion (if perfectly executed) will lead to a stable behavior, i.e. a behavior that will lead to the robot to stop and remain stabilized. Additionally, the construction of the feedback controllers ensures that the controlled motion will be locally stable, i.e. it will reject small perturbations. While we do not have any guarantees on the size of the region of stability, our experimental evaluations demonstrate that the motions are good enough to be executed in a simulator or on a real robot with substantially different dynamics. We noticed in our real-robot experiments that the synergy between the feedback controller and the motion plan is important and that none of them is solely responsible for successful execution of the motions, especially when executing a 10s long multi-contact motion.

Ideally, it would be desirable to use the optimizer in a receding horizon manner, raising the issue of closed-loop stability of the optimizer. Several methods have been proposed to ensure stability of model predictive control problems such as the use of a terminal equality constraint [57], terminal cost [58], terminal constraint set [59] or terminal cost and constraint set [60]. In this work, we use a terminal cost that keeps the terminal state within a viable set to generate balanced motions (see table I). This should thus lead to closed-loop stable behaviors.

Moreover, our approach exploits sequential convex approximations (cf. section III) to achieve polynomial-time convergence and provide a certificate of optimality or infeasibility for the motion to the desired precision. We highlight that these features do not come for free in any off-the-shelf solver. For instance, an off-the-shelf interior point method for general nonlinear problems will not take advantage of the structure of the problem as we do. This will result in a poor approximation of the nonconvex constraints unable to capture the global convex part of the problem, thus leading to slower convergence. Lastly, the certificate of optimality certifies that problem constraints are satisfied to the desired precision.

D. Cost definition and importance weights

As pointed out throughout this work, efficiency is a key concern. Consequently, the cost function (used to synthesize motions) is composed using convex quadratic expressions, as shown in Table I. The set of importance weights for these costs is, however, expected as an input (see Fig. 1), as it gives the user the flexibility to shape solutions using the knowledge about the particular robot and application. For instance, it allows expressing different preferences of endeffector force distributions in humanoid and quadruped robots. Similarly, a preference for highly dynamic and aggressive motions such as jumping (Fig. 14) over more conservative and slow motions (Fig. 8) can be expressed by lower penalties over control variables. However, automatically computing appropriate cost weights to generate desired behaviors remains an open research problem.

E. Comparison to other approaches

In [27], the motion and timings for a walking on stairs using a handrail scenario, given a sequence of contacts, are optimized in less than 5.5s. However, the multiple shooting solver used in this approach is closed-source to the best of our knowledge. In our approach, such a motion can be optimized in around 4.8s. In [9], one iteration of a multi-contact motion of 0.5s duration can be optimized within 0.05s. Thus, extrapolating, one iteration for a 7s motion could be optimized within 0.7s. This approach, however, does not take into account hard constraints. In our approach, the cost of such an iteration is around 0.61s. In [29], a bipedal motion of 4.4s is optimized within 4.1s together with the contact sequence but uses a simplified dynamics model, assuming for example a constant locked inertia tensor at the CoM. Our method would achieve a comparable time by optimizing 4 contacts within a time horizon of 5s. Our contacts planning approach based on mixed-integer programming is competitive only for small problems that optimize a few contacts, due to the combinatorial complexity of mixed-integer programs. For longer contact sequences, other state of the art approaches are more competitive, but typically use simplified dynamics to test for contact transition feasibility [17], [61], [62]. Note, however, that the kino-dynamic optimizer can be used to generate data and learn how to predict dynamic contact feasibility and significantly speed up contact search [54].

These few examples highlight the competitiveness of the presented method while enabling the resolution of the problem without simplifications. However, we are not yet capable to compute solutions for model predictive control (e.g. at 50Hz rate or above) and thus we require a feedback controller to stabilize the motion in between plan computations. Bringing such approaches to real-time rates while enabling full-body optimization remains an open problem, likely to require the design of dedicated numerical solvers and smart warm-start procedures. Lastly, we note that the receding horizon control of whole-body motions ensuring stability, robustness, and recursive feasibility, remains an open and exciting research problem.

VII. Conclusion

We have presented a structured and efficient algorithm for generating time-optimal motion plans for robots with arms and legs, as well as an approach to select a set of contact surfaces from a terrain description that supports such a motion. Finally, we have shown experimental evidence on a physical simulator and on a real quadruped robot that the algorithm is capable of efficiently generating dynamically feasible motion plans. Future work will include the extension of the algorithm to receding horizon control. The open-source repository [38] offers fully functional kino-dynamic demos, examples of tasks descriptions, and implementation details.

References

[1] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Resolved momentum control: humanoid motion planning based on the linear and angular momentum," in *IROS 2003*, pp. 1644–1650, IEEE, 2003.

[2] W. P.B., "Viability and predictive control for safe locomotion," in *IROS*, pp. 1103–1108, 2008.

[3] S. Kuindersma, F. Permenter, and R. Tedrake, "An efficiently solvable quadratic program for stabilizing dynamic locomotion.," *CoRR*, 2013.

[4] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion.," *IEEE Trans. Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[5] S. Mason, N. Rotella, S. Schaal, and L. Righetti, "An MPC walking framework with external contact forces," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1785–1790, IEEE, 2018.

[6] I. Mordatch, E. Todorov, and Z. Popovic, "Discovery of complex behaviors through contact-invariant optimization.," *ACM Trans*, 2012.

[7] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *IROS*, 2012.

[8] T. Erez and E. Todorov, "Trajectory optimization for domains with contacts using inverse dynamics," in *IROS*, pp. 4914–4919, 2012.

[9] J. Koenemann, A. D. Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard, "Whole-body model-predictive control applied to the HRP-2 humanoid," in *IROS*, pp. 3346–3351, IEEE, 2015.

[10] F. Farbod, N. Michael, W. Alexander, R. Gonzalo, and B. Jonas, "An efficient optimal planning and control framework for quadrupedal locomotion," *ICRA*, 2017.

[11] M. Neunert, F. Farshidian, A. W. Winkler, and J. Buchli, "Trajectory optimization through contacts and automatic gait discovery for quadrupeds," *IEEE RAL*, vol. 2, pp. 1502–1509, 2017.

[12] M. Giftthaler, M. Neunert, M. Stäuble, J. Buchli, and M. Diehl, "A family of iterative gauss-newton shooting methods for nonlinear optimal control," *CoRR*, vol. abs/1711.11006, 2017.

[13] M. Posa and R. Tedrake, "Direct trajectory optimization of rigid body dynamical systems through contact," in *WAFR*, 2012.

[14] I. R. Manchester, U. Mettin, F. Iida, and R. Tedrake, "Stable dynamic walking over rough terrain: theory and experiment," in *ISRR*, 2009.

[15] J. Koschorreck and K. Mombaur, "Modeling and optimal control of human platform diving with somersaults and twists," in *2011 Optimization and Engineering*, pp. 1–28, 2011.

[16] K. H. Koch, K. D. Mombaur, and P. Souères, "Optimization-based walking generation for humanoid robot," in *SyRoCo 2012*, pp. 498–504, International Federation of Automatic Control, 2012.

[17] S. Tonneau, A. D. Prete, J. Pettre, C. Park, D. Manocha, and N. Mansard, "An Efficient Acyclic Contact Planner for Multiped Robots," *IEEE Transactions on Robotics*, vol. 34, pp. 586–601, May 2018.

[18] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.

[19] Y. Lin and D. Berenson, "Humanoid navigation in uneven terrain using learned estimates of traversability," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 9–16, Nov 2017.

[20] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *Humanoids*, pp. 279–286, 2014.

[21] T. Nishi and T. Sugihara, "Motion planning of a humanoid robot in a complex environment using RRT and spatiotemporal post-processing techniques," *Int. J. Humanoid Robotics*, vol. 11, no. 2, 2014.

[22] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Auton. Robots*, vol. 35, pp. 161–176, Oct. 2013.

[23] T. Sugihara and Y. Fujimoto, "Dynamics analysis: Equations of motion," in *Humanoid Robotics: A Reference* (A. Goswami and P. Vadakkepat, eds.), Dordrecht: Springer Netherlands, 2016.

[24] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *ICRA*, pp. 3103–3109, 2013.

[25] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Humanoids*, 2014.

[26] A. Herzog, S. Schaal, and L. Righetti, "Structured contact force optimization for kino-dynamic motion generation," *IROS*, 2016.

[27] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *ICRA*, 2016.

[28] J. Carpentier and N. Mansard, "Multicontact locomotion of legged robots," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1441–1460, 2018.

[29] A. W. Winkler, D. C. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *RA-L*, pp. 1560–1567, 2018.

[30] H. Audren, J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida, "Model preview control in multi-contact motion-application to a humanoid robot," in *IROS 2014*, pp. 4030–4035, 2014.

[31] A. Herzog, N. Rotella, S. Schaal, and L. Righetti, "Trajectory generation for multi-contact momentum-control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Nov. 2015.

[32] R. Budhiraja, J. Carpentier, and N. Mansard, "Dynamics consensus between centroidal and whole-body models for locomotion of legged robots," *ICRA*, 2018.

[33] H. Dai and R. Tedrake, "Planning robust walking motion on uneven terrain via convex optimization," in *Humanoids*, pp. 579–586, MIT, Cambridge, USA, IEEE, Dec. 2016.

[34] C. Stephane and Q.-C. Pham, "When to make a step? Tackling the timing problem in multi-contact locomotion by TOPP-MPC," *Humanoids*, 2017.

[35] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3D COM accelerations," *Humanoid Robots (Humanoids)*, pp. 550–557, 2016.

[36] B. Ponton, A. Herzog, S. Schaal, and L. Righetti, "A convex model of humanoid momentum dynamics for multi-contact motion generation," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 842–849, IEEE, 2016.

[37] B. Ponton, A. Herzog, A. Del Prete, S. Schaal, and L. Righetti, "On time optimization of centroidal momentum dynamics," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5776–5782, IEEE, 2018.

[38] "Software implementation of the algorithms presented in the paper." https://github.com/machines-in-motion/kino_dynamic_opt.

[39] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *AR*, vol. 40, pp. 473–491, 2016.

[40] S.-H. Hyon, "Compliant terrain adaptation for biped humanoids without measuring ground surface and contact forces," *Robotics, IEEE Transactions on*, vol. 25, pp. 171 – 178, 03 2009.

[41] Y. Fujimoto, S. Obata, and A. Kawamura, "Robust biped walking with force interaction control between foot and ground," in *ICRA*, pp. 2030 – 2035 vol.3, 06 1998.
[42] P.-B. Wieber, "Holonomy and nonholonomy in the dynamics of articulated motion," *Fast Motions in Biomechanics and Robotics*, 2006.
[43] H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa, "A universal stability criterion of the foot contact of legged robots - adios zmp," in *ICRA*, vol. 2006, pp. 1976 – 1983, 06 2006.
[44] X. Shen, S. Diamond, Y. Gu, and S. P. Boyd, "Disciplined convex-concave programming," in *CDC 2016*, 2016.
[45] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *ECC*, pp. 3071–3076, 2013.
[46] Y.-C. Lin, B. Ponton, L. Righetti, and D. Berenson, "Efficient Humanoid Contact Planning using Learned Centroidal Dynamics Prediction," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, (Montreal), IEEE, 2019.
[47] M. Khadiv, A. Herzog, S. Moosavian, and L. Righetti, "Step timing adjustment: A step toward generating robust gaits," in *Humanoids*, 2016.
[48] S. Boyd, "Branch and bound methods." https://stanford.edu/class/ee364b/lectures/bb_slides.pdf.
[49] S. Schaal, "The sl simulation and real-time control software package," tech. rep., USC, Los Angeles, CA, 2009. clmc.
[50] F. Grimminger, A. Meduri, M. Khadiv, J. Viereck, M. Wüthrich, M. Naveau, V. Berenz, S. Heim, F. Widmaier, J. Fiene, *et al.*, "An open torque-controlled modular robot architecture for legged locomotion research," *arXiv preprint arXiv:1910.00093*, 2019.
[51] F. Farshidian and J. Buchli, "Risk sensitive, nonlinear optimal control: Iterative linear exponential-quadratic optimal control with gaussian noise," *ArXiv*, 2015.
[52] B. Ponton, S. Schaal, and R. Ludovic, "On the Effects of Measurement Uncertainty in Optimal Control of Contact Interactions," in *WAFR*, IEEE, 2016.
[53] M. Khadiv, A. Herzog, S. A. A. Moosavian, and L. Righetti, "Walking control based on step timing adaptation," *IEEE Transactions on Robotics*, pp. 1–15, 2020.
[54] Y.-C. Lin, B. Ponton, L. Righetti, and D. Berenson, "Efficient humanoid contact planning using learned centroidal dynamics prediction," *submitted to ICRA*, vol. 1, pp. 1–6, 2019.
[55] A. Nemirovski, *Interior Point Polynomial Time Methods in Convex Programming*. SIAM, 2004.
[56] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM Rev.*, vol. 47, no. 1, p. 99âĂŞ131, 2005.
[57] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations," *J. Optim. Theory Appl.*, vol. 57, pp. 265–293, May 1988.
[58] R. P. Bitmead, V. Wertz, and M. Gerers, *Adaptive Optimal Control: The Thinking Man's G.P.C.* Prentice Hall Professional Technical Reference, 1991.
[59] T. A. Johansen, "Approximate explicit receding horizon control of constrained nonlinear systems," *Automatica*, vol. 40, pp. 293–300, Feb. 2004.
[60] M. Sznaier and M. J. Damborg, "Suboptimal control of linear systems with state and control inequality constraints," *26th IEEE Conference on Decision and Control*, vol. 26, pp. 761–762, 1987.
[61] P. Fernbach, S. Tonneau, and M. Taïx, "Croc: Convex resolution of centroidal dynamics trajectories to provide a feasibility criterion for the multi contact planning problem," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, IEEE, 2018.
[62] Y.-C. Lin and D. Berenson, "Using previous experience for humanoid navigation planning," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 794–801, IEEE, 2016.

Brahayam Ponton received the B.Sc. degree in electronics and control engineering from the National Polytechnic University (EPN), Quito, Ecuador in 2011, the M.Sc degree in robotics from the Swiss Federal Institute of Technology Zürich (ETHZ), Zürich, Switzerland, in 2014 and Ph.D. degree in computer science from the Eberhard Karls Universität Tübingen, Tübingen, Germany, in 2019.

Majid Khadiv received the B.Sc. degree in mechanical engineering from the Isfahan University of Technology (IUT), Isfahan, Iran, in 2010, and the M.Sc. and Ph.D. degrees in mechanical engineering from the K.N. Toosi University of Technology, Tehran, Iran, in 2012 and 2017, respectively.
He is a Postdoctoral researcher with the Movement Generation and Control Group, Max-Planck Institute for Intelligent Systems, Tübingen, Germany. He joined the Iranian National Humanoid Project, Surena III, and worked as the Head of Dynamics and Control Group from 2012 to 2015. He also spent a one-year visiting scholarship under the supervision of Prof. L. Righetti at the Autonomous Motion Laboratory, Max-Planck Institute for Intelligent Systems. His main research interest is the control of legged robots.

Avadesh Meduri received his B.E (hons) in Manufacturing Engineering from Birla Institute of Technology and Science Pilani (BITS Pilani), India in 2019. He is currently a PhD student in the Mechanical and Aerospace Engineering Department at Tandon School of Engineering, New York University, USA.
He visited Movement Generation and Control Group at the Max-Planck Institute for Intelligent Systems to pursue his undergraduate thesis under the supervision of Prof. L. Righetti. His main research interests are contact and motion planning for legged robots.

Ludovic Righetti (Senior Member, IEEE) received an engineering diploma in computer science and a doctorate in science from the Ecole Polytechnique Federale de Lausanne, Switzerland, in 2004 and 2008, respectively.
He is an Associate Professor in the Electrical and Computer Engineering Department, the Mechanical and Aerospace Engineering Department, and the Center for Urban Science and Progress at the Tandon School of Engineering, New York University. He is also a Senior Researcher at the Max-Planck Institute for Intelligent Systems in Germany. His research focuses on the planning and control of movements for autonomous robots, with a special emphasis on legged locomotion and manipulation.