Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization

Alexander W. Winkler, C. Dario Bellicoso, Marco Hutter, and Jonas Buchli

***Abstract*—We present a single trajectory optimization formulation for legged locomotion that automatically determines the gait sequence, step timings, footholds, swing-leg motions, and six-dimensional body motion over nonflat terrain, without any additional modules. Our phase-based parameterization of feet motion and forces allows to optimize over the discrete gait sequence using only continuous decision variables. The system is represented using a simplified centroidal dynamics model that is influenced by the feet's location and forces. We explicitly enforce friction cone constraints, depending on the shape of the terrain. The nonlinear programming problem solver generates highly dynamic motion plans with full flight phases for a variety of legged systems with arbitrary morphologies in an efficient manner. We validate the feasibility of the generated plans in simulation and on the real quadruped robot ANYmal. Additionally, the entire solver software *TOWR*, which used to generate these motions is made freely available.**

***Index Terms*—Legged robots, motion and path planning, optimization and optimal control, humanoid and bipedal locomotion.**

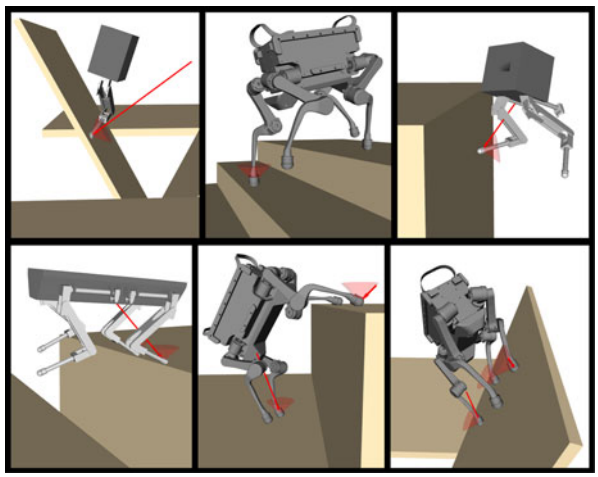

Fig. 1. Motions produced by the solver *TOWR* [2] for single-legged hoppers, bipeds and quadrupeds seen in video at https://youtu.be/0jE46GqzxMM.

I. Introduction

PLANNING physically feasible motions for legged systems is difficult. A core difficulty is that base movement cannot be directly generated, but results from contact of the feet with the environment. Therefore, the generated forces acting at these contact points must be carefully planned to achieve a desired behavior. Unfortunately, there are strong restrictions on these forces, e.g. a force can only be generated if the foot is touching the environment or feet can only push into the ground, not pull on it.

Due to the complexity of theses restrictions, hand-crafting valid trajectories for all these interdependent quantities (body, feet, forces) is tedious. Instead, Trajectory Optimization (TO) [1] can be used to generate motions in a more general, automated way. The user specifies only the high-level task, while the optimizer determines the motions and forces given these locomotion specific restrictions. The research problem is how to transcribe this continuous-time optimization problem into one with finite number of decision variables and constraints to be solvable by a Nonlinear Programming Problem (NLP) solver. This approach is attractive, because once the problem has been properly modeled, the program would, in an ideal case, produce motions (Fig. 1) for *any* high-level task, solving legged locomotion planning on a general level.

A. Related Work

In the following we categorize existing approaches to legged locomotion by their used physical model and by which aspects of the motion (e.g. body height and orientation, step sequence, timings) are fixed in advance and which are determined by an optimizer.

1) Dynamic Models: There exist a wide variety of approaches using the Linear Inverted Pendulum (LIP) model, that optimize only over the Center of Mass (CoM) position, while using predefined footholds and step timings. By modeling the robot as an inverted pendulum, the position of the Center of Pressure (CoP), or Zero-Moment-Point (ZMP) [3], can be used as a substitute for the contact forces and is used to control the motion of the CoM. This fast and effective approach has been successfully applied in bipedal and quadrupedal locomotion on real hardware [4]–[7]. This demonstrates that even such

Manuscript received September 10, 2018; accepted January 11, 2018. Date of publication February 6, 2018; date of current version February 22, 2018. This letter was recommended for publication by Associate Editor Z. Li and Editor N. Tsagarakis upon evaluation of the reviewers' comments. This work was supported by the Swiss National Center of Competence in Research Robotics. The work of J. Buchli was supported by a Swiss National Science Foundation Professorship Award. *(Corresponding author: Alexander W. Winkler.)*

A. W. Winkler and J. Buchli are with the Agile and Dexterous Robotics Laboratory, ETH Zurich, Zurich 8092, Switzerland (e-mail: winklera@ethz.ch; buchlij@ethz.ch).

C. D. Bellicoso and M. Hutter are with the Robotics Systems Laboratory, ETH Zurich, Zurich 8092, Switzerland (e-mail: bellicoso@mavt.ethz.ch; mahutter@ethz.ch).

This letter has supplemental downloadable multimedia material available at http://ieeexplore.ieee.org, provided by the authors. The Supplementary Materials contain contains a video (simulation + real robot) demonstrating the described motions. This material is 38.6 MB in size.

Digital Object Identifier 10.1109/LRA.2018.2798285

drastic model simplifications can be valid and useful. However, imposing *where* these contact forces will be acting (by predefining the footholds) strongly restricts the possible base motions. A slight relaxation is to still define *when* each foot is in contact, but allow the algorithm to determine the best location for the foothold. Together with a simplified model this results in a very fast solver that can also be used online [8], [9]. It is also possible to adapt step timings or foothold locations for robust real robot execution. Many other variations of using these simplified models to generate impressive results have been shown by [10]–[17].

In order to generate more complex motions, involving vertical movement and changes in body orientation, a more sophisticated dynamics model becomes necessary. The (6+n)-dimensional full rigid-body dynamics consider the mass and inertia of each link and defines the relation between joint torques and base- and joint-accelerations. This model takes into account changing CoM positions and inertia properties based on leg configurations and Coriolis forces generated by leg motions and has been used by [18]–[21].

Another common dynamic model used in TO is the Centroidal dynamics [22], which projects the effects of all link motions onto the 6-dimensional base. The idea is that once a physically feasible motion for the unactuated base has been found, the leg torques can be readily calculated using standard, fully-actuated inverse dynamics. The input that drives this system are the contact forces, as opposed to the CoP in the LIP or the joint torques in the full rigid-body dynamics model. Variations of this model have been successfully used by [23]–[26] to optimize for a wide variety of dynamic motions for biped robots, including demonstrations on real bipeds.

Common to all these approaches is that some part of the motion is specified beforehand. The different levels include specifying (i) only order of feet in contact (ii) order and times when each foot is in contact (iii) order, times and position of each foot in contact. This decoupling can increase optimization speed, however, it often introduces hand crafted heuristics to link these separated problems. These can become hard to tune for more complex problems and often limit the range of achievable motions. The following discusses approaches how (i)–(iii) can be automatically determined.

2) Contact Schedule Optimization: Before searching for the optimal forces at a given time, it is necessary to know if a foot is touching the environment and therefore can even exert *any* force. This reasoning about which of the feet should be in contact at a given time, or stated differently, when it is necessary to generate forces at which foot is the problem of finding a "contact schedule".

Since feet are modeled as discrete variables $\mathbb{N}$ (e.g. left-foot, right-foot), Integer Programming can be used to optimize the contact-schedule and footholds, independent from the dynamics of the system [26]–[28]. While this decoupling increases speed and allows quick solutions for each separate problem, it also necessitates heuristics that limit the range of achievable motions. Additionally, Integer Programming becomes very complex as the number of decision variables increases. Another common approach is to use a soft-contact model, which approximates the inherently hard contact surfaces as spring-damper systems [29]. The downside to this approach is that these virtual spring-damper models must be very stiff to most accurately resemble the real surface. These abrupt changes in force from a foot hitting this stiff surface hinder convergence of the optimizer. To avoid this, the problem can also be solved by formulating a Linear Complementary Problem (LCP), which enforces that either the foot is zero distance from the contact surface (touching the environment), or the force is zero [30]–[32]. These approaches produced impressive results and are closest to the work presented in this letter.

B. Contributions

We extend the above works with a *single* TO formulation for legged locomotion that automatically determines the gait-sequence, step timings, footholds, swing-leg motions, 6D body motion and required contact forces over non-flat and inclined terrain. No prior footstep planning is necessary, since our formulation directly generates the complete motion given only a desired goal position and the number of steps. We directly enforce friction constrains, which allow the algorithm to use inclined surfaces in physically feasible ways to complete desired tasks. Our algorithm extends existing algorithms that have the above capabilities in the following ways:

1) We generate motions for multiple steps in only a few seconds while still optimizing over the gait sequence. This is due to our novel phase-based parameterization of the feet and forces that keep the optimization variables continuous, and thereby the problem solvable by an NLP solver.
2) Our NLP formulation is able to automatically generate motions with full-flight phases, which are essential for highly dynamics motions.

II. Trajectory Optimization Formulation

The complete TO formulation presented in this letter can be seen in Fig. 2. The initial and desired final state of the system, the total duration T and the amount of steps $n_{s,i}$ per foot i is provided. With this information the algorithm finds a trajectory for the linear CoM position $\mathrm{r}(t)$, its orientation $\theta(t)$, the feet motion $\mathrm{p}_i(t)$ and the contact force $\mathrm{f}_i(t)$ for each foot, while automatically discovering an appropriate gait pattern defined by $\Delta T_{i,j}$.

To ensure physically feasible behavior, a simplified Centroidal dynamics model is used that relates feet position and forces with the CoM motion. Additionally, kinematic restriction between base and foot position are enforced in Cartesian space. This robot model is explained in Section III.

Independent from the base motion and dynamics, there are various restrictions on feet motions and forces described in Section IV. To ensure that feet do not slip and forces are produced only when touching the terrain, additional constraints are necessary. We introduce a novel parameterization of the variables based on each individual foot's swing and stance durations $\Delta T_{i,j}$, which allows to automatically determine the gait sequence and timings.

The presented NLP formulation allows to generate dynamic motions with full flight-phases for systems with various number of feet in just a few seconds. It can handle non-flat terrain, e.g.,

$$
\begin{aligned}
\text{find} \quad & \mathbf{r}(t) \in \mathbb{R}^3 && \text{(CoM linear position)} \\
& \boldsymbol{\theta}(t) \in \mathbb{R}^3 && \text{(base euler angles)} \\
& \text{for every foot } i: \\
& \quad \Delta T_{i,1} \ldots, \Delta T_{i,2n_{s,i}} \in \mathbb{R} && \text{(phase durations)} \\
& \quad \mathbf{p}_i(t, \Delta T_{i,1}, \ldots) \in \mathbb{R}^3 && \text{(foot position)} \\
& \quad \mathbf{f}_i(t, \Delta T_{i,1}, \ldots) \in \mathbb{R}^3 && \text{(force at foot)} \\
\text{s.t.} \quad & [\mathbf{r}, \boldsymbol{\theta}](t{=}0) = [\mathbf{r}_0, \boldsymbol{\theta}_0] && \text{(initial state)} \\
& \mathbf{r}(t{=}T) = \mathbf{r}_g && \text{(desired goal)} \\
& [\ddot{\mathbf{r}}, \dot{\boldsymbol{\omega}}]^T = \mathbf{f}_d(\mathbf{r}, \mathbf{p}_1, \ldots, \mathbf{f}_1, \ldots) && \text{(dynamic model)} \\
& \text{for every foot } i: \\
& \quad \mathbf{p}_i(t) \in \mathcal{R}_i(\mathbf{r}, \boldsymbol{\theta}), && \text{(kinematic model)} \\
& \quad \text{if foot } i \text{ in contact}: \\
& \quad\quad \dot{\mathbf{p}}_i(t \in \mathcal{C}_i) = \mathbf{0} && \text{(no slip)} \\
& \quad\quad p_i^z(t \in \mathcal{C}_i) = h_{terrain}(\mathbf{p}_i^{xy}) && \text{(terrain height)} \\
& \quad\quad \mathbf{f}_i(t \in \mathcal{C}_i) \cdot \mathbf{n}(\mathbf{p}_i^{xy}) \geq 0 && \text{(pushing force)} \\
& \quad\quad \mathbf{f}_i(t \in \mathcal{C}_i) \in \mathcal{F}(\mu, \mathbf{n}, \mathbf{p}_i^{xy}) && \text{(friction cone)} \\
& \quad \text{if foot } i \text{ in air}: \\
& \quad\quad \mathbf{f}_i(t \notin \mathcal{C}_i) = \mathbf{0} && \text{(no force in air)} \\
& \quad \textstyle\sum_{j=1}^{2n_{s,i}} \Delta T_{i,j} = T && \text{(total duration)}
\end{aligned}
$$

Fig. 2. Decision variables and constraints defining the legged locomotion TO problem. The brown quantities show those aspects of the formulation that are related to environmental contact (Section IV), while the other rows apply in general to floating base systems.

walking over stairs and jumping over gaps. In Section V we analyze selected motions, as well as validate the feasibility of the generated motion plans on a real quadruped.

A. Parameterization of Optimization Quantities

The 6D base motion is represented by the linear CoM position $\mathrm{r}(t) \in \mathbb{R}^3$, while the orientation is parameterized by Euler angles $\theta(t) \in \mathbb{R}^3$. We use fourth-order polynomials of fixed durations strung together to create a continuous spline and optimize over the polynomial coefficients. For each foot's motion $\mathrm{p}_i(t) \in \mathbb{R}^3$, we use multiple third-order polynomials per swing-phase, and a constant value $\in \mathbb{R}^3$ for the stance phase. For each foot's force profile $\mathrm{f}_i(t) \in \mathbb{R}^3$, multiple polynomials represent each stance phase and zero force is set during swing-phase. The duration of each phase, and with that the duration of each foot's polynomial, is changed based on the optimized phase durations $\Delta T_{i,j} \in \mathbb{R}$. Since the decision variables fully describe both the input (forces) and the state evolution (base and feet motion), the optimization can be considered a "simultaneous direct" method (as e.g., Collocation) [33].

III. Robot Model

A. Kinematic Model

Instead of directly constraining joint angles, as is done in full-body joint-space TO, we consider how the joint limits constrain the Cartesian foot position. We approximate each foot's workspace by a cube of edge length $2\mathrm{b}$ (see Fig. 3),

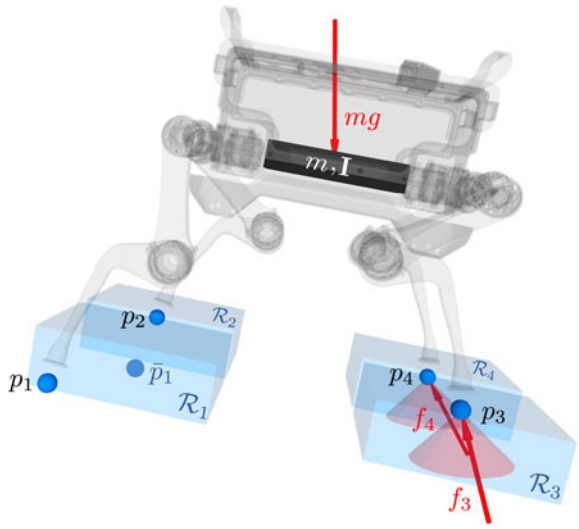

Fig. 3. The robot model known by the optimizer. The robot kinematic model is conservatively approximated by keeping the respective foot p_i inside the range of motion $\mathcal{R}_i$ of each foot. The dynamics are approximated by a single rigid-body with mass m and inertia $\mathbf{I}$ located at the robots CoM (Centroidal dynamics). This can be controlled by the contact forces f_i of the feet in contact with the environment, while keeping these forces inside the friction cone. The gray overlayed ANYmal model [34] is only for visualization and not known by the optimizer.

centered at the nominal position of each foot $\bar{\mathrm{p}}_i$ relative to the CoM. We assume the joint limits are not violated if every foot i lies inside a cube, given by

$$\mathrm{p}_i(t) \in \mathcal{R}_i(\mathrm{r}, \theta) \quad \Leftrightarrow \quad |\mathbf{R}(\theta)[\mathrm{p}_i(t) - \mathrm{r}(t)] - \bar{\mathrm{p}}_i| < \mathrm{b}, \quad (1)$$

where $\mathbf{R}(\boldsymbol{\theta})$ is the rotation matrix from the world frame to the base frame. This condition is enforced at regularly sampled states along the trajectory.

B. Dynamic Model

If we restrict the base orientation to be fixed, $\dot{\boldsymbol{\theta}}(t) = 0$, as well as the walking height to remain constant, $\mathbf{r}_z(t) = h$, it is possible to represent the system as a LIP using $\ddot{\mathbf{r}}_{xy} = (\mathbf{r}_{xy} - \mathbf{p}_{cop,xy})gh^{-1}$. However, keeping the base at constant height and orientation restricts the range of achievable motions, especially on rough terrain where some footholds can only be reached when also tilting the base. Additionally, by replacing the effect of individual contact forces with a single CoP, important information is lost, e.g., keeping each individual force inside the corresponding friction cone cannot be enforced anymore. Finally, situations with all feet in the air cannot be represented with this model, as a CoP $\mathbf{p}_{cop,xy}$ must always exist. For the above reasons, we decide this model is not expressive enough to represent the motions we wish to generate.

Another possibility for the dynamic model are the very accurate joint-space rigid-body dynamics, using joints torques as input. However, the main difficulty of legged locomotion remains in finding a physically feasible motion for the under-actuated base given the locomotion specific restrictions seen in Fig. 2. These physical constraints can be most intuitively expressed in Cartesian-, not joint-space, as they relate to the environment.

Once a physically feasible motion for the base has been found the joint torques can easily be obtained using inverse dynamics.

We choose to remain in Cartesian space, since this allows to formulate the relevant variables and constraints in a more simple way and model the legged robot using simplified Centroidal dynamics (see Fig. 3). The CoM linear $\ddot{\mathbf{r}}$ and angular $\dot{\boldsymbol{\omega}}$ acceleration are determined by

$$m\ddot{\mathbf{r}}(t) = \sum_{i=1}^{n_i} \mathbf{f}_i(t) - m\mathbf{g}$$

$$\mathbf{I}\dot{\boldsymbol{\omega}}(t) + \boldsymbol{\omega}(t) \times \mathbf{I}\boldsymbol{\omega}(t) = \sum_{i=1}^{n_i} \mathbf{f}_i(t) \times (\mathbf{r}(t) - \mathbf{p}_i(t)), \qquad (2)$$

where m is the mass of the robot, n_i the number of feet, $\mathbf{g}$ is the gravity acceleration and $\boldsymbol{\omega}(t)$ represents the angular velocity that can be calculated from the optimized Euler angles $\boldsymbol{\theta}(t)$ and rates $\dot{\boldsymbol{\theta}}(t)$ (see Appendix VI-B). We use a constant rotational inertia $\mathbf{I} \in \mathbb{R}^{3\times3}$ calculated for the robot in the nominal pose. This assumes that either the limb masses are negligible compared to the torso or that the limbs do not deviate significantly from their default pose. These assumptions make the dynamics of the robot independent of the joint configuration and express them solely in Cartesian space. For the presented robots and motions the above assumptions introduce only negligible modeling error while keeping the formulation simpler and the solver fast. Further discussion of the dynamic model is postponed to Section V.

To ensure physical behavior of the motion, we enforce (2) at regular time intervals along the trajectory. Additionally, we constrain the acceleration at the junction between two base polynomials to be equal, as jumps in acceleration would imply jumps in force or foot position, which we do not allow.

C. Contact Independent Dynamic Model

Until now, the dynamic model (2) can just as well represent a flying drone. What makes it specific to legged systems is that the restriction on the forces and feet positions abruptly change depending on whether a foot is in contact with the environment or not. These discretely switching contact configuration and therefore discretely switching constraints are difficult to handle. For example, NLP formulations don't naturally allow constraints to simply be turned on or off arbitrarily during the iterations. This is why the sequence and durations of contacts are often specified in advance when using an NLP to solve the legged locomotion problem.

To partially simplify the problem, the robot model can be viewed independently from concepts such as contacts or phases and the discontinuities can be handled where they actually occur – in the individual foot motion and forces. As a consequence of treating every foot separately, concepts that described multiple feet at once, such as *phases* or *contact configurations* can be simplified to binary *in contact* or *not*.

The following section does not include any robot dependent quantities (kinematic, dynamic) anymore, nor is it dependent on the base motion. From now on each foot is treated separately and is only affected by the terrain and the physical constraints coming from non-slip, frictional contact of rigid bodies.

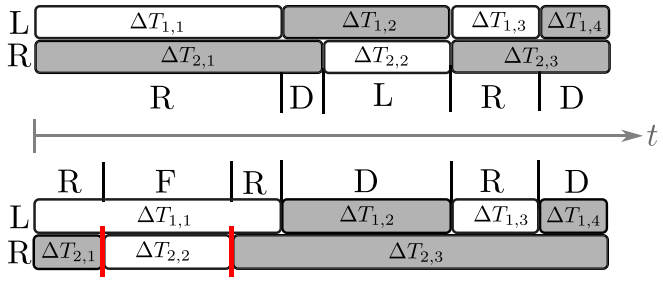

Fig. 4. Two different contact schedules for a bipedal robot. L = left foot in contact, R = right foot in contact, D = both feet standing, F = flight phase. The change of gait is achieved solely by adapting the stance durations (red delimiters) of the right foot.

IV. Contact Model

In this section we first explain how arbitrary gaits can be generated by modifying the durations of each individual foot's swing and stance phase. We then describe how we exploit this knowledge to formulate an NLP with continuous optimization variables, that is still able to optimize over the gait sequence. Finally, we describe how we model the physical constraints between the terrain and the foot motion and forces.

A. Contact Schedule Optimization

A biped walk can be characterized by phases (L, R, D, F) as seen in Fig. 4. If the number of possible phases is low, these can possibly be predefined in a sensible, intuitive way for a given task. However, as the number of contact points increases (e.g., quadruped, bipeds with hands, allowing other body parts in contact), it becomes highly complex to determine which of feet $\in \mathbb{N}$ should make contact in what order to achieve a desired task.

However, we can observe that more than two phases only exist when viewing multiple feet simultaneously. When looking at a single-legged hopper, there exist *exactly* two phases – a contact phase $\mathcal{C}$ and a flight phase. Furthermore, these two phases always alternate: After the foot is in contact, it will be in a flight phase, then again in contact, etc.

Analogously, we can view multi-legged robots as having independent feet, each alternating between contact and flight. What varies to generate the different gaits are the durations of each foot's swing and stance phase. Fig. 4 shows that solely by changing the phase durations $\Delta T_{i,j} \in \mathbb{R}$ of the right foot, a completely different gait can be generated. Since the phase durations are continuous, these can be readily optimized by NLP solvers and Integer Programming can be avoided.

B. Feet Motion and Forces Parameterization

To exploit this regularity, each dimension of the quantities $\mathbf{p}_i(t), \mathbf{f}_i(t)$ is described by alternating sequences of constant values and cubic polynomials

$$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3, \quad a_i = f(\Delta T_, x_0, \dot{x}_0, x_1, \dot{x}_1)$$

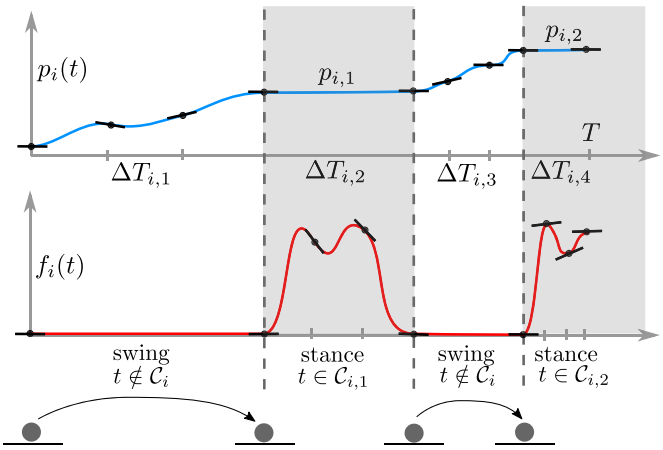

Fig. 5. Phase-based parameterization of foot i's motion $\mathbf{p}_i$ and force $\mathbf{f}_i$. Each phase (swing or stance) is represented by either a constant value or a sequence of cubic polynomials with continuous derivatives at the junctions. The optimizer is able to modify the phase durations $\Delta T_{i,j}$, thereby changing the shape of the functions. Performing this for all feet allows to generate arbitrary gait patterns, while still using continuous decision variables $\Delta T_{i,j}$.

as shown in Fig. 5. Instead of optimizing over polynomial coefficients, we use the value x and derivative $\dot{x}$ at the end-points ("nodes") and its duration ΔT to fully define each polynomial (see Appendix VI-A). This so-called "Hermite" parameterization is more intuitive, since the optimization variables directly describe the state. Furthermore, the node used as the end of the previous polynomial can also be used as the starting node of the next, which ensures continuous foot velocity and force changes over the trajectory.

In Fig. 5 we use three polynomials of equal duration $\Delta T_{i,j}/3$ to represent each swing phase of the foot motion and each stance phase of the foot force. These can represent typically varying force and motion profiles while still keeping the problem as small as possible. The other phases are represented by a constant value for a duration of $\Delta T_{i,j}$. We predefine the maximum number of steps $n_{s,i}$ each foot can take. Note that this is not a strong restriction, as phase durations can always be set close to zero if less steps are required.

The times at which foot i is in contact for the sth time are given by

$$\mathcal{C}_{i,s} = \left\{ t | 0 < t - \textstyle\sum_{j=1}^{2s-1} \Delta T_{i,j} < \Delta T_{i,2s} \right\}. \tag{3}$$

This uses the intuition that every new foothold s is preceded by exactly two phases j (swing and stance). The set of all times that foot i is in contact is denoted by $\mathcal{C}_i = \cup_{s=1}^{n_{s,i}} \mathcal{C}_{i,s}$.

The individual polynomials carry information about whether they represent a swing or stance phase, and this never changes. However, the algorithm still has the flexibility to change the contact state at a given time by adapting the relevant phase durations and thereby make this time fall onto a polynomial of different contact state. Therefore, by changing these durations, all contact schedules/gaits can be generated. Since we are changing the durations, we must ensure that the total duration of each foot's motion and force spline ends at the specified total time. Therefore, we have the additional constraint for every foot i that $\Delta T_{i,1} + \cdots + \Delta T_{i,2n_{s,i}} = T$.

With this parameterization we directly impose that a foot in contact does not slip, more specifically, that the velocity of the foot motion during stance phase is zero. This is not a constraint of the optimization problem, but is ensured directly by our parameterization through a single, constant position variable $\mathbf{p}_{i,s}$ as

$$\dot{\mathbf{p}}_i(t \in \mathcal{C}_{i,s}) = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{p}_i(t \in \mathcal{C}_{i,s}) = \mathbf{p}_{i,s} = \text{const.} \tag{4}$$

If a foot is not in contact, no force can be produced. Therefore, we set each constant value representing the force in the flight-phase to zero as

$$\mathbf{f}_i(t \notin \mathcal{C}_i) = \mathbf{0}. \tag{5}$$

The above restrictions (4), (5) are handled *before* starting the optimization and are equivalent to the LCP constraint $\dot{\mathbf{p}}_i(t)\mathbf{f}_i(t) = 0$ used in other TO formulations with automatic gait discovery. However, instead of checking this conditions at every sampling time t along the trajectory during the optimization, our phase-duration based optimization allows us to predefine this condition a-priori. This simplifies the problem for the solver and decreases computation time.

As seen in Fig. 5, we additionally ensure that the foot motion and force profiles are smooth at phase junctions (continuously differentiable) and thereby easier for the gradient based solver to handle. Physically this is not required as contact with the environment can be impulsive, which abruptly zeros the foot velocity and spikes the contact force.

C. Terrain Height Constraint

A foot is only in contact if it is touching the terrain. Therefore, the height of the foot during contact must match the terrain at that 2D foot position $\mathbf{p}_{i,s}^{xy} = (p_{i,s}^x, p_{i,s}^y)$. The continuous height map $h_{\text{terrain}}(x, y)$ can be either manually specified if the objects in the environment are known or be generated from stereo camera data. We constrain the variable representing the constant foothold height of foot i during stance phase s by

$$p_i^z(t \in \mathcal{C}_{i,s}) = p_{i,s}^z = h_{\text{terrain}}(\mathbf{p}_{i,s}^{xy}). \tag{6}$$

D. Stance Force Constraints

For physically correct locomotion it is necessary that forces can only push into the contact surface, and not pull on it. For flat ground and an LIP model this can be equivalently formulated as keeping the CoP inside the area spanned by the feet in contact. Since our formulation has explicit values for the contact forces we can directly constrain these as

$$f_n(t \in \mathcal{C}_{i,s}) = \mathbf{f}^T(t)\mathbf{n}(\mathbf{p}_{i,s}^{xy}) \geq 0, \tag{7}$$

where $\mathbf{n}(x, y)$ denotes the normal vector defining the slope of the terrain at position x, y. The scalar product extracts the component of the force that is orthogonal to the terrain. For flat ground, $\mathbf{n}(x, y) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and the constraint simplifies to $f^z(t) \geq 0$.

It follows from Coulombs law that pushing stronger into a surface allows to exert larger side-ways forces without slipping. This is equivalent to keeping tangential forces f_{t1}, f_{t2} inside the friction cone defined by the friction coefficient μ as $\sqrt{f_{t1}^2 + f_{t2}^2} < \mu f_n$. We approximate this friction cone by a friction pyramid, enforcing an upper and lower bound for the force in both tangential directions $\mathbf{t}_1, \mathbf{t}_2$. This pyramid approximation introduces only negligible error but linearizes this constraint, simplifying the problem for the NLP solver. The constraint is given by

$$\begin{aligned} &-\mu f_n < f_{\{t_1,t_2\}} < \mu f_n \\ \Leftrightarrow \quad &|\mathbf{f}^T(t)\mathbf{t}_{\{1,2\}}(\mathbf{p}_{i,s}^{xy})| < \mathbf{f}^T(t)\mathbf{n}(\mathbf{p}_{i,s}^{xy}). \end{aligned} \tag{8}$$

V. Results

This section discusses the variety of motions generated with the presented algorithm for a single-legged hopper, a biped robot and the quadruped robots ANYmal [34] and HyQ [35]. First, the motion plans, fulfilling all the specified physical constraints, are analyzed and discussed. Secondly, we demonstrate a subset of motions in the realistic physics simulator Gazebo as well as on the quadruped robot ANYmal [34].

This requires a controller that is able to reliably track the generated motion-plans by incorporating current sensor data in order to calculate the appropriate joint torques. This is not a trivial task, and just as much a research topic as generating the plans. Our controller solves a hierarchy of tasks using optimization to most accurately track the plans and is described in detail in [36]. In simulation, we demonstrate highly dynamic and full-body walking, trotting, pacing and galloping, all produced by the same method and tracked with the same controller. Additionally, we show that despite model mismatches, sensor noise, torque tracking inaccuracies and delays the motion plans are robust enough to be tracked on a real system. A quadruped trot and walk with optimized full 6D-body motion and directly planned contact forces is executed on a real system. These examples are another form of validation that our formulation produces motion plans that are physically feasible.

The accompanying video[1] shows a visualization of the generated motion plans using inverse kinematics and simulation and real robot experiments. The biped gap crossing example below can be seen in the video at 01:09 and is shown in Fig. 6. The motions are optimized with *TOWR* [2] and visualized with *XPP* [37], which also provide ROS bag files of some optimized motions.

A. Example: Biped Gap Crossing

1) Dynamic Consistency: A main focus in TO is to generate motions plans that are physically feasible. For shooting methods that optimize only over the inputs and integrate to get the state the dynamics are always fulfilled. For the used simultaneous method the dynamic constraint (2) is enforced only at discretized times (red nodes). We therefore calculate the true base vertical acceleration $\ddot{r}^z_{\text{true}} = m^{-1}(f_L^z + f_R^z) - g$ (black dashed line) from (2) using the optimized forces and compare it to the optimized result $\ddot{r}^z$ (black solid line). The same evaluation can be done for the other five base coordinates $\ddot{r}^x, \ddot{r}^y, \dot{\omega}^x, \dot{\omega}^y, \dot{\omega}^z$. As can be seen, these values coincide exactly at the node values, as these are enforced by hard constraints, and deviate only slightly in between, e.g. during the dynamic sequence at $t = 2.0$–2.2 s. The Root-Mean-Squared-Error (RMSE) for the base vertical acceleration is $1.8433 \frac{m}{s^2}$.

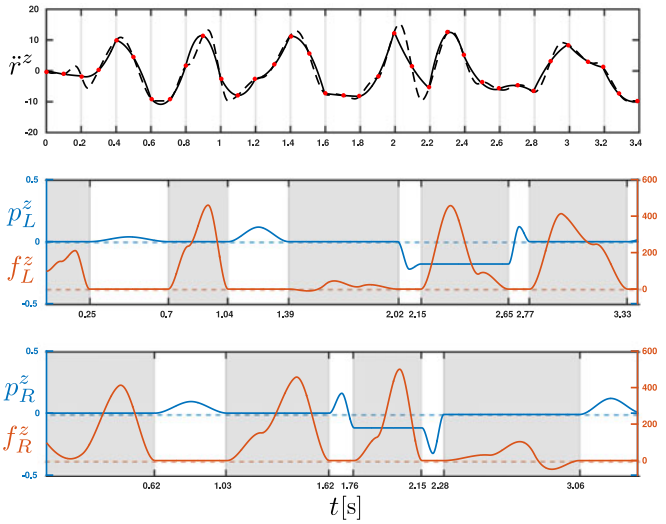

Fig. 6. A generated motion plan for a 20 kg-bipedal robot crossing a 1 m wide gap (see video at 01:09). The plots show the base vertical acceleration $\ddot{r}^z(t)$ as well as the vertical position and vertical force of the left (L) and right (R) leg. The planned base vertical acceleration is compared to the vertical body acceleration that results from evaluating (2) with the current footholds and forces. In this example we use fourth-order polynomials of duration 0.2 s for the base motion parameterization. The red nodes, spaced 0.1 s apart, show the times at which the dynamic constraint is enforced in the NLP. We use two third-order polynomials to parameterize each foot motion in swing-phase (white area), and three third-order polynomials for each force profile per foot in stance-phase (shaded area).

In case more accuracy is required, dynamic constraints can be enforced at a finer grid, e.g. every 50 ms. However, one must keep in mind that (i) the dynamic model is an approximation of the true dynamic system and will also never be perfectly accurate (ii) enforcing the dynamics exactly might be unnecessary since the controller cannot even track the desired motions exactly, due to sensor noise, inaccurate force tracking and delays. Taking the above into account, sensible model accuracy must be chosen for each hardware, controller and problem individually.

2) Foot Contact Constraints: The foot height $p^z(t)$ for the left and right foot is shown by the blue lines. We notice that the constant segments during stance phase (gray areas) are mostly at the tableau height $h_{\text{terrain}} = 0$ (blue dotted line) as required by (6). The z-positions lower than zero are where the biped steps into the sides of the gap while traversing it. In order the have gradient information available, we model the gap $h_{\text{gap}}(x, y)$ as a 5 m deep parabola, instead of a discretely changing ground height. This helps the NLP solver to converge to a solution.

Vertical forces $f^z(t)$ only exist whenever the corresponding foot is in stance phase (gray area). This is enforced by the parameterization of the force profile given by (5). During the

[1]Video of generated motions: https://youtu.be/0jE46GqzxMM.

TABLE I
NLP SPECS FOR BIPEDAL GAP CROSSING

Horizon T: 4.4 s	$\mathrm{d}t$-kinematic: 0.05 s	Variables: 926
Goal x: 3.7 m	$\mathrm{d}t$-dynamic: 0.1 s	Constraints: 1543
Steps/foot $n_{s,L/R}$:5	Iterations: 21	**T-solve (Ipopt): 4.1 s**

stance phase the force can only push into the terrain as required by (7). Since the normal direction of the terrain changes at the side of the gaps, negative z-forces are also physically feasible, as seen at $t = 2.8$ s.

3) Automatic Gait Discovery: The algorithm is able to automatically change the initially provided gait sequence and timings depending on the terrain and desired task. The motion was initialized with a walking gait, with short two-leg support phases between every step. As can be seen in Fig. 6, flight-phases have been automatically inserted at e.g., $t = 0.62$–0.7 s. We constrain each phase duration variable $\Delta T_{i,j}$ to be greater than 0.1 s in this example to avoid very quick swing- or extremely short stance phases. Flight-phases allow the solver to respect kinematic limits while still covering a large distance with few steps. Capabilities such as these show the advantages of optimizing all the aspects of the motion simultaneously.

4) Fast Solver: As seen in Table I our formulation is able to generate the sample biped motion in 4.1 s. This includes optimization over the contact sequence, finding 6D-body, foothold and swing-leg motions as well as enforcing friction constraints. The kinematic constraint (1) is enforced every 0.05 s, while the dynamic constraint (2) is checked every 0.1 s. Generating a motion of two steps per leg for a quadruped robot takes ∼300 ms. For the other motions shown in the video, with time horizons around 5 s involving dozens of steps, the algorithm takes ∼20 s. These values vary depending on the problem definition, terrain and other parameters. Nonetheless, other approaches that can generate similarly complex motions often take orders of magnitude longer. The results where obtained using C++ code interfaced with Interior Point Method solver Ipopt [38] on an Intel Core i7/2.8 GHz Quadcore laptop. The Jacobians of the constraints are provided to the solver analytically, which is important for performance. Another factor that speeds up the optimization, is that we do not use a cost function, as observed in Fig. 2. This also reduces the amount of tuning parameters, as the relative importance of the constraints does not have to be quantified – they all have to be fulfilled for a motion to be physically feasible.

B. Limitations and Future Work

For humanoid robots with heavy limbs deviating far from its nominal configuration the simplified Centroidal dynamics model (2) might not be sufficient. If a more accurate dynamic representation is necessary, the model can be refined by calculating the joint angles $\mathbf{q}$ from the optimized foot positions $\mathbf{p}$ using inverse kinematics and updating $\mathbf{I}(\mathbf{q})$. Likewise, optimized foot velocities $\dot{\mathbf{p}}$ can be converted into angular velocities $\boldsymbol{\omega}_i$ of each leg link (e.g., using the leg Jacobian) and used in (2) explicitly. These refinements will increase the nonlinearity of the dynamic constraint and possibly computation time, but can still be handled by the proposed formulation and NLP solver.

The solver enforces the terrain constraints (terrain collision, unilateral force, friction cone) only at the junctions of the 3rd-order feet polynomials. Since these polynomials are usually short, violation of the constraints in between is often negligible. However, especially when a foot motion polynomial enforces the terrain constraint (6) only at the borders and an obstacle is in between, undesired terrain collision can occur.

Finally, for highly uneven terrain the solver is sometimes trapped in local minima. One way to simplify the problem for the solver is to fix the step timings, thereby not optimizing over the gait sequence. As long as the range of motion is large enough to reach the desired goal in the specified number of steps the solver consistently finds solutions to the problem and the solution time rapidly decreases.

VI. CONCLUSION

We presented a TO formulation that is able to efficiently generate complex, highly dynamic motions for a variety of legged systems over non-flat terrain, while also optimizing over the contact sequence. The feasibility of the motion plan is demonstrated in simulation and on a real quadruped. In the future we will transfer even more motions to the real system and also use this fast motion planner in an Model Predictive Control (MPC) fashion.

APPENDIX

A. Hermite Parameterization

Each cubic polynomial $x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$ can either be parameterized by its coefficients *or* the value and first derivative and the end points x_0, x_1 and the duration ΔT as

$$
\begin{aligned}
a_0 &= x_0 \\
a_1 &= \dot{x}_0 \\
a_2 &= -\Delta T^{-2}[3(x_0 - x_1) + \Delta T(2\dot{x}_0 + \dot{x}_1)] \\
a_3 &= \Delta T^{-3}[2(x_0 - x_1) + \Delta T(\dot{x}_0 + \dot{x}_1)].
\end{aligned}
$$

B. Euler Angles and Rates to Angular Velocities

The transformation [39] from the optimized Euler angles $\boldsymbol{\theta}$ (order of application: yaw, pitch, roll) and rates $\dot{\boldsymbol{\theta}}$ to the angular velocities in world frame are

$$
\boldsymbol{\omega} = \mathbf{C}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \begin{bmatrix} \cos(\theta_y)\cos(\theta_z) & -\sin(\theta_z) & 0 \\ \cos(\theta_y)\sin(\theta_z) & \cos(\theta_z) & 0 \\ -\sin(\theta_y) & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{bmatrix}
$$

$$
\dot{\boldsymbol{\omega}} = \dot{\mathbf{C}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}.
$$

REFERENCES

[1] J. T. Betts, "Survey of numerical methods for trajectory optimization," *J. Guid., Control, Dyn.*, vol. 21, no. 2, pp. 193–207, 1998. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/2.4231

[2] A. W. Winkler, "TOWR—An open-source trajecetory optimizer for legged robots in C++." 2018. [Online]. Available: http://wiki.ros.org/towr

[3] M. Vukobratović and B. Borovac, "Zero-moment point—Thirty five years of its life," *Int. J. Humanoid Robot.*, vol. 1, no. 1, pp. 157–173, 2004.

[4] S. Kajita *et al.*, "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, pp. 1620–1626. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1241826

[5] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Fast, robust quadruped locomotion over challenging terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 2665–2670. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5509805

[6] A. W. Winkler *et al.*, "Path planning with force-based foothold adaptation and virtual model control for torque controlled quadruped robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 6476–6482.

[7] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5148–5154.

[8] A. W. Winkler, F. Farshidian, M. Neunert, D. Pardo, and J. Buchli, "Online walking motion and foothold optimization for quadruped locomotion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 5308–5313.

[9] A. W. Winkler, F. Farshidian, D. Pardo, M. Neunert, and J. Buchli, "Fast trajectory optimization for legged robots using vertex-based ZMP constraints," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2201–2208, Oct. 2017.

[10] I. Mordatch, M. de Lasa, and A. Hertzmann, "Robust physics-based locomotion using low-dimensional planning," *ACM Trans. Graph.*, vol. 29, no. 4, 2010, Article No. 71. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1778765.1778808

[11] H. Diedam, D. Dimitrov, P.-B. Wieber, K. Mombaur, and M. Diehl, "Online walking gait generation with adaptive foot positioning through linear model predictive control," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2008, pp. 1121–1126.

[12] A. Herdt, N. Perrin, and P. B. Wieber, "Walking without thinking about it," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2010, pp. 190–195.

[13] V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. R. De Pieri, and D. G. Caldwell, "A reactive controller framework for quadrupedal locomotion on challenging terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2554–2561.

[14] C. Gehring *et al.*, "Towards automatic discovery of agile gaits for quadrupedal robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 4243–4248.

[15] H.-W. Park, P. M. Wensing, and S. Kim, "Online planning for autonomous running jumps over obstacles in high-speed quadrupeds," in *Proc. Robot., Sci. Syst.*, 2015. [Online]. Available: http://www.roboticsproceedings.org/rss11/p47.html

[16] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères, "A reactive walking pattern generator based on nonlinear model predictive control," *IEEE Robot. Autom. Lett.*, vol. 2, no. 1, pp. 10–17, Jan. 2017.

[17] C. Mastalli *et al.*, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 1096–1103.

[18] G. Schultz and K. Mombaur, "Modeling and optimal control of human-like running," *IEEE/ASME Trans. Mechatronics*, vol. 15, no. 5, pp. 783–792, Oct. 2010.

[19] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne, "Locomotion skills for simulated quadrupeds," in *Proc. ACM SIGGRAPH*, 2011, Paper 59. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1964921.1964954

[20] F. Farshidian, M. Neunert, A. W. Winkler, G. Rey, and J. Buchli, "An efficient optimal planning and control framework for quadrupedal locomotion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 93–100.

[21] D. Pardo, M. Neunert, A. W. Winkler, R. Grandia, and J. Buchli, "Hybrid direct collocation and control in the constraint-consistent subspace for dynamic legged robot locomotion," in *Proc. Robot., Sci. Syst.*, 2017. [Online]. Available: http://www.roboticsproceedings.org/rss13/p42.html

[22] D. E. Orin, A. Goswami, and S. H. Lee, "Centroidal dynamics of a humanoid robot," *Auton. Robots*, vol. 35, no. 2/3, pp. 161–176, 2013.

[23] M. Kudruss *et al.*, "Optimal control for multi-contact, whole-body motion generation using center-of-mass dynamics for multi-contact situations," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, pp. 684–689, 2015. [Online]. Available: https://homepages.laas.fr/ostasse/drupal/sites/homepages.laas.fr.ostass e/files/15-ichr-kudruss.pdf

[24] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 3555–3561.

[25] A. Herzog, S. Schaal, and L. Righetti, "Structured contact force optimization for kino-dynamic motion generation," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2016, pp. 2703–2710.

[26] B. Ponton, A. Herzog, S. Schaal, and L. Righetti, "A convex model of humanoid momentum dynamics for multi-contact motion generation," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2016, pp. 842–849.

[27] A. Ibanez, P. Bidaud, and V. Padois, "Emergence of humanoid walking behaviors from mixed-integer model predictive control," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2014, pp. 4014–4021.

[28] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2015, pp. 279–286.

[29] M. Neunert, F. Farshidian, A. W. Winkler, and J. Buchli, "Trajectory optimization through contacts and automatic gait discovery for quadrupeds," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1502–1509, Jul. 2017.

[30] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–8, 2012.

[31] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 69–81, 2013. [Online]. Available: http://ijr.sagepub.com/content/33/1/69.short

[32] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 295–302.

[33] M. Diehl, H. G. Bock, H. Diedam, and P. B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics* (Lecture Notes in Control and Information Sciences), vol. 340. Berlin, Germany: Springer, 2006, pp. 65–93.

[34] M. Hutter *et al.*, "ANYmal—A highly mobile and dynamic quadrupedal robot," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2016, pp. 38–44.

[35] C. Semini, "HyQ–Design and development of a hydraulically actuated quadruped robot," Ph.D. dissertation, Istituto Italiano di Tecnologia, Genoa, Italy, 2010. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:HyQ +-+Design+and+Development+of+a+Hydraulically+Actuated+Quadruped+Robot#0

[36] C. D. Bellicoso, F. Jenelten, P. Fankhauser, C. Gehring, J. Hwangbo, and M. Hutter, "Dynamic locomotion and whole-body control for quadrupedal robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 3359–3365.

[37] A. W. Winkler, "XPP—A collection of ROS packages for the visualization of legged robots." 2017. [Online]. Available: http://wiki.ros.org/xpp

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

[39] C. Gehring *et al.*, "Kindr library—Kinematics and dynamics for robotics," 2016. [Online]. Available: https://docs.leggedrobotics.com/kindr/cheatsheet_latest.pdf. Accessed on: Nov. 25, 2017.