Article

Offline motion libraries and online MPC for advanced mobility skills

Marko Bjelonic[1] ⬤**, Ruben Grandia[1], Moritz Geilinger[2],
Oliver Harley[1]** ⬤**, Vivian S Medeiros[3], Vuk Pajovic[1], Edo Jelavic[1],
Stelian Coros[2] and Marco Hutter[1]**

Abstract

We describe an optimization-based framework to perform complex locomotion skills for robots with legs and wheels. The generation of complex motions over a long-time horizon often requires offline computation due to current computing constraints and is mostly accomplished through trajectory optimization (TO). In contrast, model predictive control (MPC) focuses on the online computation of trajectories, robust even in the presence of uncertainty, albeit mostly over shorter time horizons and is prone to generating nonoptimal solutions over the horizon of the task's goals. Our article's contributions overcome this trade-off by combining offline motion libraries and online MPC, uniting a complex, long-time horizon plan with reactive, short-time horizon solutions. We start from offline trajectories that can be, for example, generated by TO or sampling-based methods. Also, multiple offline trajectories can be composed out of a motion library into a single maneuver. We then use these offline trajectories as the cost for the online MPC, allowing us to smoothly blend between multiple composed motions even in the presence of discontinuous transitions. The MPC optimizes from the measured state, resulting in feedback control, which robustifies the task's execution by reacting to disturbances and looking ahead at the offline trajectory. With our contribution, motion designers can choose their favorite method to iterate over behavior designs offline without tuning robot experiments, enabling them to author new behaviors rapidly. Our experiments demonstrate complex and dynamic motions on our traditional quadrupedal robot ANYmal and its roller-walking version. Moreover, the article's findings contribute to evaluating five planning algorithms.

Keywords

Robotics, wheeled and legged locomotion, model predictive control, offline motion library, trajectory optimization

1. Introduction

When humans and animals engage in complex locomotion behaviors, the motions involve a fast interaction between the feet and the environment's contact. Additionally, whole-body coordination operating near physical limits or over challenging obstacles is demanding, requiring look-ahead planning to decide the upcoming steps carefully. The synchronization of fast interactions and look-ahead planning is a fascinating research direction. Planning such physically feasible motions for (wheeled-)legged robots, as shown in Figure 1, is challenging since the whole-body movement results from contact forces at the feet (or wheels). Therefore, we need to carefully choose the interaction at these contact points with the environment to achieve the desired behavior while respecting physical laws.

1.1 Offline and online optimal-control

Designing high-dimensional trajectories, for example, whole-body trajectories including ground reaction forces, is

difficult. In most cases, it is not feasible to hand-craft due to the complexity of the constraints. When analyzing the literature in Section 2, it appears that optimal-control algorithms can be separated from a practical point of view into two groups: offline and online algorithms.

The usage of trajectory optimization (TO) (see reviews by Betts (1998); Rao (2009)) offers the chance to solve the locomotion problem over the task's time horizon while optimizing a higher-dimensional variable space composed of the whole-body trajectory, gait sequences, and timings. With a task-specific goal, these variables are found automatically while considering constraints imposed by physical restrictions. In the last few years, legged locomotion

[1]Robotic Systems Lab, ETH Zürich, Zurich, Switzerland
[2]Computational Robotics Lab, ETH Zürich, Zurich, Switzerland
[3]Department of Mechanical Engineering, PUC-Rio, Rio de Janeiro, Brazil

Corresponding author:
Marko Bjelonic, Robotic Systems Lab, ETH Zürich, Leonhardstrasse 21, Zürich 8092, Switzerland.
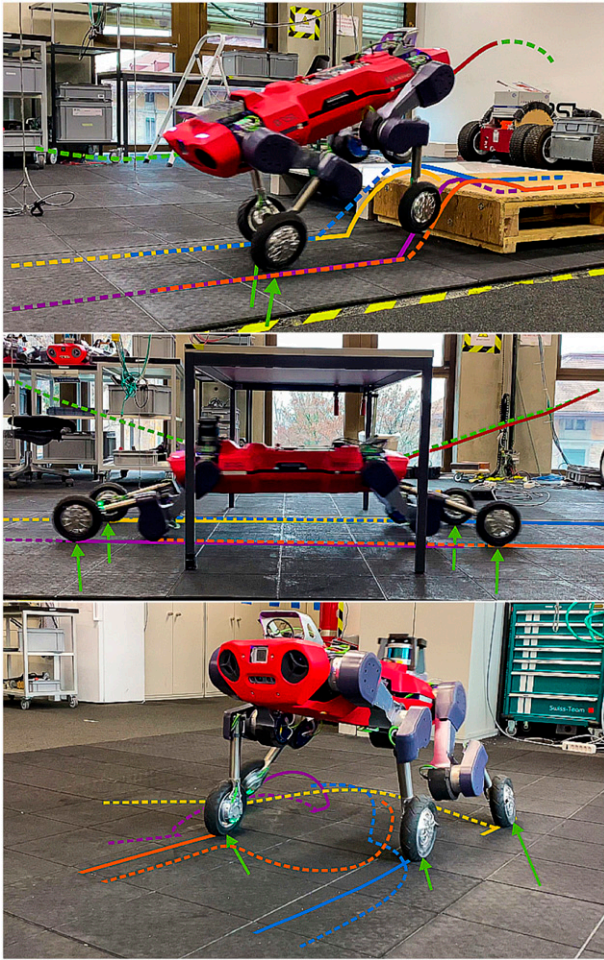Email: marko.bjelonic@mavt.ethz.ch

Figure 1. Our quadrupedal robot ANYmal (Hutter et al., 2017) with wheels executes complex and agile skills through offline TO and online MPC. With our novel approach, the robot achieves dynamic and artistic motions in challenging terrain. First image: Terrain-aware locomotion over a 0.20 m high step (32 % of leg length). Second image: Ducking motion under a table. Third image: Dynamic 180°turn in approximately 1 s. The dotted lines show the TO's offline trajectories of the whole-body, while the solid lines show the online solution of the MPC. A video available at https://youtu.be/39rRhTqcQc0 showing the results accompanies this article.

research experienced a surge in TO approaches that solve the locomotion problem over long-time horizons from scratch (Geilinger et al. 2018, 2020; Herzog et al., 2015; Jelavic and Hutter 2019; Mastalli et al., 2017; Medeiros et al., 2020; Melon et al., 2020; Mordatch et al., 2012; Park et al., 2016; Posa et al., 2014; Todorov 2011; Winkler et al., 2018; Yeganegi et al., 2019). Most of these approaches share one aspect in common that their algorithms run offline and, due to the computational complexity, reliable online execution on the real robot becomes one of the main challenges.

Instead, a large amount of pioneering work in model predictive control (MPC) (see reviews by Morari and Lee (1999); Mayne et al. (2000); Bertsekas (2005); Diehl et al.

(2009)) focuses on the online execution of whole-body optimization problems on the real robot in a task-generic fashion (Caron and Pham, 2017; Bjelonic et al., 2021; Bledt and Kim 2019; Bledt et al., 2018; Dantec et al., 2020; Dai et al., 2014; Erez et al., 2013; Farshidian et al., 2017; Grandia et al., 2019a, 2019b; Herdt et al., 2010; Henze et al., 2014; Koenemann et al., 2015; Laurenzi et al., 2018; Mason et al., 2018; Neunert et al., 2018; Paparusso et al., 2020; Zimmermann et al., 2015). Due to fast update rates, MPC finds solutions on the fly while correcting the motion under unforeseen conditions, such as modeling errors and external disturbances, thus offering feedback control. In contrast to TO, these motions are generated over relatively short-time horizons and neglect complex optimization variables and constraints, such as gait timings and higher-order model complexities, to speed up the solver time. Therefore, the optimization may not guarantee an optimal solution for tasks requiring a longer time horizon and a more accurate physical representation, for example, parkour over obstacles. The characteristics of both fields can be summarized as follows:

Offline trajectory optimization. Offline computations of long-time horizons and task-specific problems are solved from scratch while considering higher-order model complexities.

Online model predictive control. Online computations of short-time horizons and task-generic problems are solved by a shifted guess from the previously computed optimal solution and explore task simplifications for computational reasons.

1.2 Contribution

We formalize, evaluate, and analyze a complete planning and control solution for the relatively underexplored domain of hybrid wheeled-legged locomotion. Our article's theoretical contribution is a whole-body[1] approach to locomotion planning that combines offline motion libraries and online MPC, generating complex and dynamic motions for (wheeled-)legged robots. The former finds complex motions enabling whole-body coordination near robot limits and over challenging obstacles (see Figure 2) with task-specific time horizons. We then use these offline trajectories as the online MPC's cost, which explores reactive optima over a fixed look-ahead. Moreover, we can store offline trajectories in a motion library where each motion can be composed into a single maneuver.

We showcase three state-of-the-art TO algorithms for the offline computation of complex trajectories, namely an interactive TO, a terrain-aware TO, and a sampling-based method, allowing motion designers to choreograph complex motions. With this demonstration of different offline motion planners, we verify that our complex motion composition can be extended with other algorithms providing whole-body trajectories.

The main conceptual contribution of this work is the real-time execution of the offline trajectories on the real robot. To
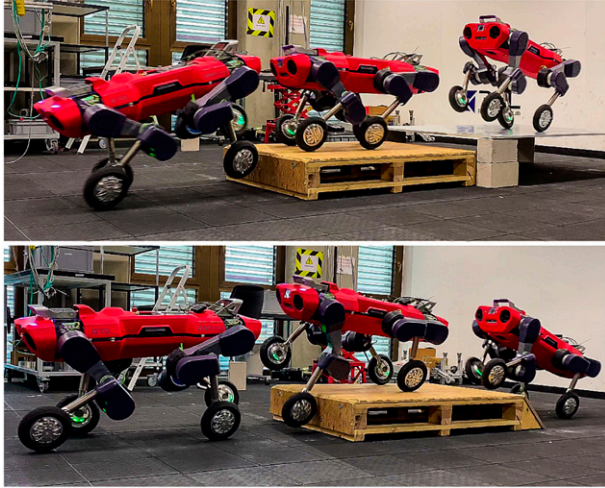
Figure 2. Our quadrupedal robot ANYmal with wheels overcomes two steps in a row (upper image) and steps up-and-down (lower image) through the terrain-aware TO and MPC. The maximum step height is 0.20 m (32 % of leg length) and the robot locomotes the obstacle course with a maximum speed of 1.5 m/s while only lifting its legs when needed.

this end, our online optimization is based on a whole-body MPC that closes the offline-to-online gap. Similar to the TOs, it is based on a single task formulation that simultaneously optimizes feet (or wheels) and torso motions. Due to the real-time joint velocity and ground reaction force optimization based on a kinodynamic model, the MPC can follow a high variety of different offline motions, thus is general with respect to (w.r.t.) the task. Also, we discuss the importance of incorporating a look-ahead along the offline trajectories, which enables the anticipation of future events. Traditional MPC with a short-sighted plan can generate a sub-optimal solution for tasks over longer time horizons. In our work, the augmentation of the MPC's cost with offline trajectories avoids these problems.

As shown in Figures 1 and 13, the experiments demonstrate complex and dynamic motions on our traditional four-legged robot ANYmal and its roller-walking version, further demonstrating the findings' generalizability and approach's applicability to any robot with legs and wheels.

Succinctly, the six main findings (C1) to (C6) of this article can be summarized as follows:

(C1) Whole-body coordination. The online MPC coordinates offline trajectories of complex, whole-body motions operating near robot limits while being responsive to the terrain.

(C2) Offline-to-online gap. The online MPC optimizes from the measured state, resulting in feedback control, which adds a certain degree of robustness and reliability to the offline trajectory's execution by recomputing solutions on the fly and reacting to unforeseen conditions, such as modeling errors and external disturbances. Also, the MPC anticipates future events of the offline trajectory.

(C3) Rapid motion prototyping. Executing offline trajectories on the real robot is a tedious task requiring tuning robot experiments. Due to the properties (C1) and (C2) of

our online motion planner, motion designers can choose their favorite motion generator to iterate over behavior designs offline without tuning robot experiments, enabling them to author new behaviors rapidly.

(C4) Motion composition. Motion designers can compose multiple offline trajectories into a single maneuver. Since these (composed) trajectories are fed into the MPC as a cost (and not as a constraint or initialization method) and due to the look-ahead of the MPC, our approach makes it possible to smoothly blend between trajectories even in the presence of discontinuities, avoiding hand-tuned heuristics at these transitions.

(C5) Performance and generalization. The robot can conduct artistic motions, dance, and find optimal solutions over non-flat terrain. Our framework is not restricted to our roller-walking robot ANYmal and is generalized to any robotic creature with legs and wheels.

(C6) Evaluation of offline motion planners. The article's findings serve as an evaluation of three offline motion planners for ANYmal.

2. Related work

In the following, we categorize existing approaches to legged locomotion planning by offline TO and online MPC methods and, finally, review techniques that combine both optimal-control approaches.

2.1 Offline trajectory optimization

Most of the related work in the field of TO for legged locomotion precomputes complex trajectories over a task-specific time horizon. Due to offline generation and a focus on higher-order optimizations, reducing the computation time becomes a lower priority compared to MPC. As such, TO algorithms choose from a set of models that captures the real robot's dynamics more accurately. For example, the rigid body dynamics (RBD) model only assumes non-deformable links and the equations of motion (EOM) can be rewritten as the Centroidal dynamics (CD) model without loss of generality (Orin et al., 2013; Kuindersma et al., 2016). Budhiraja et al. (2019) discuss the dynamic consensus between centroidal and the RBD models. The single rigid body dynamics (SRBD), on the other hand, assumes that the legs are massless and, as can be seen below, is another commonly used model in TOs (Herzog et al., 2015).

Winkler et al. (2018) propose a phase-based end-effector parameterization for simultaneously optimizing gait and whole-body motion over non-flat terrain. A nonlinear programming (NLP) algorithm solves the problem, and, as a model, the authors deploy the SRBD of a legged robot. The approach is augmented by Medeiros et al. (2020) to incorporate wheeled locomotion without stepping. The optimization problem, however, becomes prone to local minima and, as such, depends on a good initialization. Melon et al. (2020) propose a learning-based scheme to

solve this problem by initializing the NLP based on offline experiences. In contrast, Mordatch et al. (2012) and Carius et al. (2019) present a contact invariant TO formulation to synthesize legged robots' motions. As shown by Deits and Tedrake (2014) and Aceituno-Cabezas et al. (2017), mixed-integer optimization problems can solve the combinatorial complexity of simultaneous gait and motion planning.

Recently, research in hybrid locomotion showed a rise of new TO methods. Skaterbots show impressive results based on the CD model of robots with wheels and legs (Geilinger et al. 2018, 2020). In contrast to the work of Winkler et al. (2018) and Medeiros et al. (2020), the authors of Skaterbots have access to the leg's kinematics, which becomes even more critical for wheeled-legged robots since the estimation of the rolling direction is required. Thus, the work of Medeiros et al. (2020) requires a heuristic that approximates the rolling constraint. A whole-body, kinematic TO (Jelavic and Hutter 2019) and a sampling-based method (Jelavic et al., 2021) showcase static motions for legged excavators in visualization only.

In contrast to MPC, some TO methods offer the chance to be immune to local minima when the domain has discontinuous dynamics (Erez et al., 2012). These algorithms, however, are often too slow to be applied online on the real robot, though impressive results are being showcased in visualization and inside simulation environments. On the real robot, the related work struggles with the execution of these motions. For example, Winkler et al. (2018) only manage to run a few simple trajectories on flat terrain. The authors use a tracking controller that only looks at one set point at a time to execute the offline trajectories.

2.2 Online model predictive control

The task of MPC is to find the robot's continuous-time motion over a fixed time horizon, that is, the torso's trajectories and feet (or wheels). Moreover, the algorithm must carefully plan the ground reaction forces to achieve the desired behavior.

To reduce the model complexity and enable real-time execution, the research community is experimenting with simplifying the real robot's dynamics. For example, by modeling the robot as a linear inverted pendulum (LIP) model, the zero-moment point (ZMP) developed by Vukobratović and Borovac (2004), or the center of pressure (COP), is used to control only the motion of the center of mass (COM) position and acts as a substitute for the contact forces (Bellegarda et al., 2018; Bellicoso et al., 2018; Bjelonic et al. 2019, 2020; Caron et al., 2017; De Viragh et al., 2019; Jenelten et al., 2020; Kalakrishnan et al. 2010, 2011; Krause et al., 2012; Mason et al., 2018; Mastalli et al., 2020; Sardain and Bessonnet 2004; Winkler et al. 2015, 2017; Zucker et al., 2011). Additionally, the locomotion planning's high-dimensional problem can be decomposed into a separate torso and feet (or wheels) planning problems.

Thus, the resulting two lower-dimensional sub-tasks become more tractable (Bellicoso et al., 2018; Bjelonic et al., 2020; Buchanan et al., 2020; Di Carlo et al., 2018; Englsberger et al., 2014; Focchi et al. 2017, 2020; Griffin et al., 2019; Kajita et al., 2001; Jenelten et al., 2020; Kalakrishnan et al., 2010; Klamt and Behnke 2018; Laurenzi et al., 2018; Kudruss et al., 2015; Naveau et al., 2017; Park et al., 2015; Rebula et al., 2007; Tsounis et al., 2020; Wieber 2006; Wieber et al., 2016; Zucker et al., 2011).

In our previous work (Bjelonic et al., 2021), we verify that the full set of possible solutions cannot be discovered by such a decomposed method. The individual components do not consider the complete set of physical constraints. Instead, a single task approach should be deployed that treats the continuous-time decision problem as a whole without breaking down the problem into several sub-tasks. Such holistic solutions automatically discover complex and dynamic motions that are impossible to find through hand-tuned heuristics at the cost of higher computation times. Also, a single set of parameters for all behaviors makes the algorithm general w.r.t. the task, which is crucial when executing different offline trajectories. Another finding shows that the LIP model does not accurately capture the real model's accuracy, and higher-dimensional models, for example, SRBD models, are required for more complex tasks.

In the last few years, traditional legged locomotion research experienced a large amount of pioneering work in MPC that now reliably runs single task optimizations based on an SRBD or CD model (Caron and Pham 2017; Dai et al., 2014; Erez et al., 2013; Farshidian et al., 2017). Bledt and Kim (2019) implement a regularized predictive controller for simultaneous footstep and ground reaction force optimization. The authors show their results on the quadrupedal robot, MIT Cheetah 3 (Bledt et al., 2018). Similarly, Grandia et al. (2019a, 2019b) and Neunert et al. (2018) deploy an MPC on the ANYmal robot (Hutter et al., 2017). In contrast, the former work introduces a kinodynamic model, which defines the SRBD model along with the kinematics for each leg.

MPC offers robust and reactive control for even high-dimensional (wheeled-)legged robots. Running MPC over a short-time horizon does not result in optimal motions over the task's full receding horizon. The effects of using a short-sighted optimization become evident when considering tasks that involve crossing challenging obstacles like stepping stones (Dantec et al., 2020; Grandia et al., 2021; Magaña et al., 2019; Mastalli et al., 2020) and stairs (Fankhauser et al., 2018; Jenelten et al., 2020). Here, the robot makes unnecessary steps over the terrain patches, and depending on the difficulty of the obstacle, the robot can be guided into inescapable situations. Moreover, the MPC gets stuck in local minima when the domain has discontinuous dynamics (Erez et al., 2012).

2.3 Combination of trajectory optimization and model predictive control

Neunert et al. (2016) propose a method that uses a sequential linear quadratic (SLQ) algorithm in an MPC setting to unify the optimization of trajectories over multiple seconds and tracking within only a few milliseconds. The results, however, are only shown on a hexacopter and a ball balancing robot. Running these algorithms with a planning horizon of multiple seconds in real-time on a switched system like legged robots, as demonstrated by Neunert et al. (2018) and Farshidian et al. (2017), becomes unfeasible and requires a novel control approach. In contrast, Li et al. (2020) formulate a single optimization problem posed over a hierarchy of two models achieving trajectories over a longer time horizon than traditional MPC with a single model. To achieve this, the authors fall back to a simplified model at longer time horizons. The MPC posed over two models face the same challenges as the traditional MPC approach. It misses the ability to solve task-specific problems, is prone to local minima and the simplified model has a limited solution space at the system's limitations and over challenging obstacles. Similarly, Zimmermann et al. (2015) generate motion plans for the immediate future using higher-fidelity models, while coarser models are used to create motion plans with longer time horizons. All motions are shown on a bipedal robot in simulation.

Erez et al. (2012) use offline TO to find the limit-cycle solution of an infinite-horizon, average-cost optimal-control task and apply its quadratic approximation as the online MPC's terminal cost. Their work, however, is only verified in a simulation environment. In aeronautical research, Lapp and Singh (2004) use the two optimal-control paradigms by feeding the TO's offline solution as a cost term to the MPC.

The combination of offline TO and online MPC has not been studied in depth in the field of (wheeled-)legged robotics. Boston Dynamics' bipedal robot Atlas demonstrates dynamic and parkour-like motions over challenging obstacles (Boston Dynamics 2019). The video's description and a presentation by Kuindersma (2020) reveal that the company is working on a similar approach, as introduced in our article. A TO transforms high-level descriptions of each motion into dynamically-feasible reference motions tracked using an MPC. Due to Atlas's missing publications, there is no in-depth knowledge about Boston Dynamics' locomotion framework.

3. Complex motion composition

In the following, we give an overview of our complex motion composition framework that combines the advantages of offline TO and online MPC. Figure 3 visualizes our complete locomotion controller that is discussed in more detail in the following sections.

3.1 Problem formulation and solution

Optimizing and executing a complex motion on the real robot comes with different and opposing challenges,
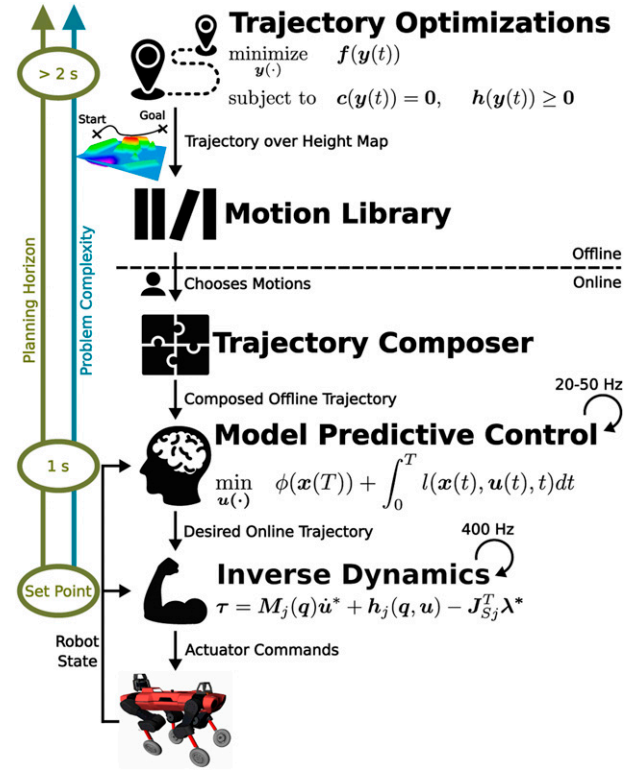

Figure 3. Overview of our complex motion composition. The TOs transform high-level tasks into dynamically feasible motions stored in a motion library. Based on the operator's chosen motions, individual motions from this library are composed into a single offline trajectory with a total time horizon TTO. This trajectory is fed into the MPC, optimizing joint velocities and contact forces over a shorter time horizon TMPC. Finally, the inverse dynamics transforms the desired online trajectory into actuator commands.

making our hierarchical structure's design crucial. On the one hand, we aim for a general-purpose solution that can take any offline trajectory as input for real-time execution. These offline trajectories can contain different polynomial representations with a variable number of nodes. Concatenating multiple trajectories results in jumps that are difficult to track and, in some cases, might result in the online optimization failing. On the other hand, our online optimization needs to be self-contained. The system should act independently without relying on perfectly modeled offline trajectories even under unpredicted disturbances.

We, therefore, propose to integrate the offline trajectory as a cost to our online MPC that contains three essential characteristics: (1) prediction over a short horizon, (2) flexible offline trajectories as inputs, and (3) self-contained online execution. The prediction over a short-time horizon enables the robot to anticipate the offline trajectory. Even in more significant tracking discrepancies, the robot can plan its whole-body trajectory back towards the desired offline motion. In contrast, as shown in the results, tracking only one set point at a time does not result in a robust motion execution. The second characteristic, that is, flexible offline trajectories as inputs, comes through integrating the offline

trajectory as a cost term. The motion designer does not need to make sure that the composed motions are continuous or dynamically feasible. Adding the offline trajectory as a hard constraint or initializing the optimization might make the problem unfeasible. The last characteristic ensures that even in large tracking offsets, the online planner can still find robust motions. In the next section, we present our approach in more detail.

3.2 Overview

Our approach to optimizing complex locomotion strategies for (wheeled-)legged robots consists of a hierarchical structure, as visualized in Figure 3. First, the Trajectory Optimizers transform high-level tasks, for example, user-guided or terrain-aware maneuvers, into dynamically-feasible trajectories. These trajectories are stored in a Motion Library that is accessible by an operator at run time. Moreover, the operator can choose multiple motions. Given the commanded motion reference(s), the Trajectory Composer concatenates each trajectory into a single Offline Trajectory fed into our Model Predictive Control algorithm, which tracks this trajectory and smoothly blends from one maneuver to the next. Finally, the Inverse Dynamics computes actuator commands that are sent to the robot, and a state estimator predicts the robot's state.

3.3 Timings and problem complexity

It is essential to look more closely at planning horizons, problem complexities, and update rates of each module in Figure 3. When it comes to the former two categorizations, we can see that the planning horizon and problem complexity increase vertically. The inverse dynamics is only looking at one set point at a time, while the MPC looks-ahead with a fixed planning horizon of one second. In contrast, the TO generates complete maneuvers, often incorporating a more complex dynamic model with a task-specific time horizon up to multiple seconds. Due to this long-time horizon and the complexity of the task, for example, dynamic behaviors at the robot's limits or terrain-aware maneuvers over obstacles, the motions are computed offline. The composition of these offline trajectories results in even longer time horizons. With the decrease of the problem's complexity from top to bottom, the MPC and the inverse dynamics can achieve update rates on the robot of around 20–50 Hz and 400 Hz, respectively.

In the following sections, we introduce our main theoretical contributions, the definition of the offline trajectory including the three offline motion generators, the online MPC, and the trajectory composer in more detail.

4. Offline motion generation

In general, the offline TO's problem is expressed as a nonlinear optimization problem with objective $f(y(t))$,

equality constraint $c(y(t))$, and inequality constraint $h(y(t))$, that is

$$\underset{y(\cdot)}{\text{minimize}} \quad f(y(t)) \tag{1a}$$

$$\text{subject to} \quad c(y(t)) = 0, \quad h(y(t)) \geq 0, \tag{1b}$$

where the optimization variable $y(t)$ includes the robot's whole-body trajectory over a task-specific horizon T_{TO}, that is, the 6D torso motion, the end-effector[2] motion, including individual joint motions, and end-effector contact forces. In some TO, the optimization problem may include additional variables like gait sequences and timings, which increases the problem's dimensionality and complexity.

4.1 Offline trajectory

In this work, the combination of offline TO and online MPC comes through the offline trajectory. The offline motion generation might include additional optimization variables that are not considered by the online MPC. To this end, we split the optimization variable $y(t)$ in (1) into

$$y(t) = \begin{bmatrix} x_{\text{TO}}^T & u_{\text{TO}}^T & \cdots \end{bmatrix}^T \tag{2}$$

where $x_{\text{TO}}(t)$ is the desired state vector, $u_{\text{TO}}(t)$ is the control input vector, and the three dots indicate the additional optimization variables of the offline TO. These two vectors form the offline trajectory $\begin{bmatrix} x_{\text{TO}}^T & u_{\text{TO}}^T \end{bmatrix}^T$, which is being tracked by the MPC in Section 5, and are given by

$$x_{\text{TO}}(t) = \begin{bmatrix} \boldsymbol{\theta}^T & \boldsymbol{p}^T & \boldsymbol{\omega}^T & \boldsymbol{v}^T & \boldsymbol{q}_j^T \end{bmatrix}^T \tag{3a}$$

$$u_{\text{TO}}(t) = \begin{bmatrix} \boldsymbol{\lambda}_E^T & \boldsymbol{u}_j^T \end{bmatrix}^T \tag{3b}$$

where $x_{\text{TO}}(t) \in \mathbb{R}^{12+n_j}$ is the desired state vector and $u_{\text{TO}}(t) \in \mathbb{R}^{3n_e+n_j}$ is the desired control input vector at time t, with $n_j = 12$ is the number of joints and $n_e = 4$ is the number of legs in the case of a quadrupedal robot. Note that the additional four degrees of freedom (DOF) of the wheels are not explicitly in the desired state or input vector. More precisely, the MPC in Section 5 models the wheel as a moving point contact. This convention enables us to model conventional point-foot and wheels by merely changing the kinematic constraints (Bjelonic et al., 2021). The elements $\boldsymbol{\theta}(t)$, $\boldsymbol{p}(t)$, $\boldsymbol{\omega}(t)$, $\boldsymbol{v}(t)$ and $\boldsymbol{q}_j(t)$ of the state vector refer to the orientation of the torso in Euler angles, the position of the torso in world frame W, the angular rate of the COM in torso frame B, the linear velocity of the COM in torso frame B, and the joint positions, respectively. Moreover, the control inputs are the end-effector contact forces $\boldsymbol{\lambda}_E(t)$ and joint velocities $\boldsymbol{u}_j(t)$. Alternatively, this offline trajectory can include a different set of variables, for example, replacing joint velocities with joint torques and reducing dimensionality. We found that this set of variables enables highly dynamic motions and offers flexibility for the motion designer since some TOs, for example, in Section 4.4, do not provide joint torques.

In the following, we present two TOs and a combination between sampling and optimization-based planning that solve various tasks while incorporating different model complexities. The solution of each TO provides the offline trajectory $x_{\text{TO}}(t)$ and $u_{\text{TO}}(t)$ over a task-specific horizon T_{TO}.

4.2 Interactive trajectory optimization

One way to generate offline motions is to use the interactive TO, that is, a tool allowing a two-way flow of information between the motion optimizer and designer. Motion designers can add and modify high-level motion goals, such as target COM position and orientation at specific moments in time. Meanwhile, trajectory optimization finds motion plans that satisfy kinematic and dynamic constraints. Also, the immediate visual feedback allows the motion designer to choreograph complex motions interactively.

The interactive TO is based on the work of Geilinger et al. (2018) and Geilinger et al. (2020). In the following, we briefly summarize the optimization problem solved by the interactive TO.

4.2.1 Problem formulation. The motion generation's model is based on the CD model and complemented with geometric constraints to ensure the generated motions' consistency with the robot's kinematics. Depending on the type of end-effector, for example, traditional point feet or actuated wheels, constraints on the ground reaction forces and the end-effector's state trajectories are instantiated. We find that this approach strikes a favorable balance between predictive power, simplicity, and computational efficiency for rapid motion prototyping.

Based on a robot's morphology and high-level motion goals, trajectory optimization finds a motion plan

$$y(t) = \begin{bmatrix} x_{\text{cd}}^T & x_{\text{torso}}^T & x_{\text{leg},1}^T & \cdots & x_{\text{leg},n_e}^T & q_j^T \end{bmatrix}^T \quad (4)$$

with

$$x_{\text{cd}}(t) = \begin{bmatrix} \theta_{\text{cd}}^T & p_{\text{cd}}^T & \omega_{\text{cd}}^T & v_{\text{cd}}^T \end{bmatrix}^T \quad (5a)$$

$$x_{\text{leg},i}(t) = \begin{bmatrix} p_{E_i}^T & v_{E_i}^T & \lambda_{E_i}^T & w_{E_i} & \alpha_{E_i} \end{bmatrix}^T \quad (5b)$$

$$x_{\text{torso}}(t) = \begin{bmatrix} \theta^T & p^T & \omega^T & v^T \end{bmatrix}^T \quad (5c)$$

The centroidal coordinate frame is represented by the robot's orientation and COM position in world frame W, $\theta_{\text{cd}}(t)$ and $p_{\text{cd}}(t)$, whereas its angular and linear velocity are described by $\omega_{\text{cd}}(t)$ and $v_{\text{cd}}(t)$. The state of each end-effector i is captured by $x_{\text{leg},i}(t)$ consisting of the contact position and its velocity, $p_{E_i}(t)$ and $v_{E_i}(t)$ in world coordinates, and the ground reaction forces $\lambda_{E_i}(t)$ at the contact position. For end-effectors equipped with wheels, $w_{E_i}(t)$ and $\alpha_{E_i}(t)$ denote the wheel speed and the wheel's orientation, both in world coordinates. The variables describing the state of the torso $X_{\text{torso}}(t)$ are introduced in (3).

With the motion plan's parameters in place, we define the cost terms and constraints that arise from the dynamics and kinematics that govern a robot's motion in Appendix A. At the core of this interactive TO lies its interactive capabilities and underlying CD model. The former gives the motion designer interactive control over the robot's motion by formulating a set of objectives at specific times.

4.3 Terrain-aware gait and trajectory optimization

Our second TO in this work relies on a terrain-aware approach to optimize gait and motion simultaneously. Motion designers merely add start and goal poses of the robot's torso to the optimization problem, including a terrain representation as a 2.5D elevation map. Meanwhile, TO finds motion plans that satisfy kinematic and dynamic constraints over non-flat terrain.

The terrain-aware gait and TO formulation for (wheeled) legged robots is presented in the following section and is based on the work of Winkler et al. (2018). In contrast to Medeiros et al. (2020), the optimization problem can generate simultaneous driving and stepping motions, that is, hybrid locomotion. Here, we overview the optimization problem generating regular walking and hybrid locomotion trajectories.

4.3.1 Problem formulation. The motion designer provides an initial and desired final state of the robot, the total time horizon T_{TO}, and a gait schedule including an approximate duration of each leg's contact and swing timings. With this information and a given height map $h_{\text{terrain}}(x, y)$ of the terrain, the algorithm finds

$$y(t) = \begin{bmatrix} x_{\text{torso}}^T & x_{\text{leg},1}^T & \cdots & x_{\text{leg},n_e}^T \end{bmatrix}^T \quad (6)$$

with

$$x_{\text{torso}}(t) = \begin{bmatrix} \theta^T & p^T & \omega^T & v^T \end{bmatrix}^T \quad (7a)$$

$$x_{\text{leg},i}(t) = \begin{bmatrix} r_{E_i}^T & v_{E_i}^T & \lambda_{E_i}^T & \Delta T_{i,j}^T \end{bmatrix}^T \quad (7b)$$

where most of the variables are introduced in (3), and Figure 4 visualizes each optimization variable. Due to the missing kinematics of each leg, that is, no joint information, the TO finds the end-effector's position $r_{E_i}(t)$ of each leg i w.r.t. the COM and its velocity $v_{E_i}(t)$ w.r.t. the world frame W instead, while automatically discovering appropriate phase durations j of leg i defined by $\Delta T_{i,j}(t)$.

Appendix B describes each of the optimization problem's constraints. At the core of this terrain-aware TO lies its rough terrain capability and underlying SRBD model. Here, the former integrates the information of the height map $h_{\text{terrain}}(x, y)$ and makes sure that the end-effector stays in contact with the ground while the leg is in stance phase.

4.3.2 Inverse kinematics. The optimized trajectories $x_{\text{torso}}^*(t)$ and $x_{\text{leg},i}^*(t)$ in Section 4.3.1 miss the required joint
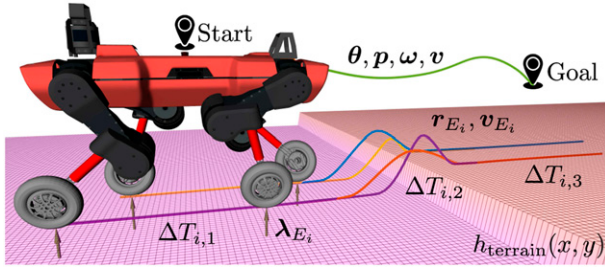
Figure 4. Overview of the terrain-aware gait and TO's variables. Here, the robot overcomes a step with a height of 0.15 m, and the visualization shows the optimized trajectories xtorso(t) and xleg,i(t). Each of the variables is introduced in Section 4.3.1.

information of the offline trajectory introduced in Section 4.1. To this end, we compute the joint position trajectory $\boldsymbol{q}_j^*(t)$ and joint velocity trajectory $\boldsymbol{u}_j^*(t)$ through inverse kinematics. Moreover, we use an iterative inverse kinematics approach of Goldenberg et al. (1985) to obtain the joint positions $\boldsymbol{q}_j^*(t)$, while the joint velocities are calculated through $\boldsymbol{u}_j^*(t) = \boldsymbol{J}_E^+(\boldsymbol{q}_j^*(t))\dot{\boldsymbol{r}}_E(t)$, where $\boldsymbol{J}_E = \partial \boldsymbol{r}_E/\partial \boldsymbol{q}_j \in \mathbb{R}^{3n_e \times n_q}$ is the end-effector Jacobian w.r.t. the torso frame B and $(\cdot)^+$ is the Moore–Penrose inverse of a matrix.

4.4 Combined sampling and optimization-based planning

Optimizing gait timings over challenging terrain is a complex problem for (wheeled-)legged robots and can be solved through sampling-based methods. In our work, we adopt the approach of Jelavic et al. (2021) that is designed for legged excavators and apply it to our wheeled-legged robot ANYmal. This offline motion generator combines an initialization and refinement step. The former is a sampling-based planner that samples robot poses through a Rapidly-Exploring Random Tree (RRT) (Karaman and Frazzoli 2011) and the second step refines the motion with a non-linear optimization, producing kinematically feasible and statically stable offline trajectories.

5 Online model predictive control

The offline generated trajectory in Section 4.1 is fed into our online MPC as a cost, allowing us to reactively optimize along the offline trajectory even in the presence of discontinuous transitions and unpredicted disturbances. This feedback control re-optimizes the offline trajectories from the measured state. The following sections describe the underlying optimization, as well as the integration of the offline trajectories into the online execution.

5.1 Problem formulation

The main advantage of MPC is that it allows the current input to be optimized while considering future desired states. Such an optimization is achieved by optimizing over a finite time-horizon of T_{MPC}, repeatedly, enabling the

anticipation of future events. At each iteration of the MPC, we solve the optimization problem based on

$$\underset{\boldsymbol{u}(\cdot)}{\text{minimize}} \quad \phi(\boldsymbol{x}(T_{\text{MPC}})) + \int_0^{T_{\text{MPC}}} l(\boldsymbol{x}(t),\boldsymbol{u}(t),t)\mathrm{d}t \quad (8\text{a})$$

$$\text{subject to} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t),t) \quad (8\text{b})$$

$$\boldsymbol{x}(0) = \boldsymbol{x}_0 \quad (8\text{c})$$

$$\boldsymbol{g}_1(\boldsymbol{x}(t),\boldsymbol{u}(t),t) = 0 \quad (8\text{d})$$

$$\boldsymbol{g}_2(\boldsymbol{x}(t),t) = 0 \quad (8\text{e})$$

$$\boldsymbol{h}(\boldsymbol{x}(t),\boldsymbol{u}(t),t) \geq 0 \quad (8\text{f})$$

where $\boldsymbol{x}(t)$ is the state vector and $\boldsymbol{u}(t)$ is the control input vector at time t, which form the whole-body trajectory, and each vector is described in (3a) and (3b), respectively. Figure 5 visualizes the trajectories of both vectors while overcoming a step. The cost function in (8a) consists of the time-varying running cost $l(\cdot)$, and the cost $\phi(\cdot)$ at the terminal state $\boldsymbol{x}(T_{\text{MPC}})$. In addition, solutions need to satisfy the system dynamics (8b), initial condition (8c), and further equality (8d), (8e), and inequality constraints (8f). The SLQ formulation[3] of Grandia et al. (2019a, 2019b) and Farshidian et al. (2017), which is a differential dynamic programming (DDP) based algorithm (Mayne 1966), calculates the feedback policy for continuous-time systems. Moreover, the algorithm computes via quadratic approximations of the value function a time-varying, state-affine control policy through an alternating iteration of simulation (forward pass) and the optimization (backward pass). We use a Lagrangian method, a penalty method, and a relaxed barrier function to handle the state-input equality constraint (8d), the pure state equality constraint (8e), and the inequality constraint (8f), respectively. As described by Grandia et al. (2019a, 2019b), frequency-shaped cost functions robustify the solutions in the presence of compliant contacts and bandwidth limitations due to actuator dynamics.

In contrast to Bjelonic et al. (2021), which does not incorporate any offline trajectories or perception, our article's contribution comes through the MPC's cost function that incorporates the TO's offline trajectories. The set of constraints that we have chosen makes the MPC a general-purpose optimization problem that can deal with slight mistakes from the motion designer while considering the terrain and offline trajectory in front of the robot. Our article continues with more details about the implementation of the cost function and constraints.

5.1.1 Cost function. As discussed in Section 3.1, incorporating the offline trajectory given by $\boldsymbol{x}_{\text{TO}}(t)$ and $\boldsymbol{u}_{\text{TO}}(t)$ into the MPC as a cost (instead of a hard constraint or initialization method) offers flexibility for the motion designer since the trajectory does not need to be dynamically feasible nor continuous. We achieve this by feeding the two vectors of the TO as a cost term into
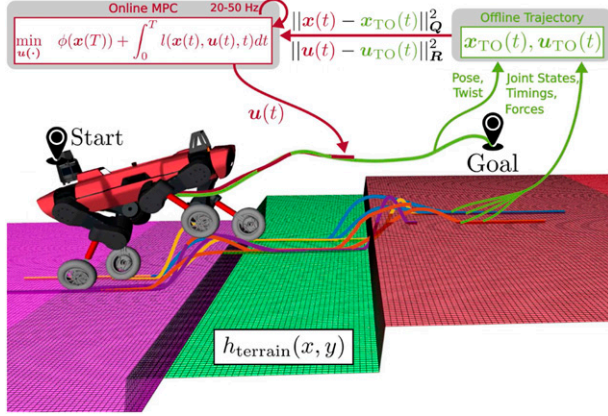
Figure 5. The optimized trajectory of the MPC while tracking the offline trajectory of Figure 4. Here, the robot's front legs are stepping up the obstacle, and the visualization shows the optimized trajectories $x(t)$ and $u(t)$. Each of the variables is introduced in Section 5.1.

$$l(\boldsymbol{x},\boldsymbol{u},t) = \frac{1}{2}\tilde{\boldsymbol{x}}(t)^T \boldsymbol{Q}\tilde{\boldsymbol{x}}(t) + \frac{1}{2}\tilde{\boldsymbol{u}}(t)^T \boldsymbol{R}\tilde{\boldsymbol{u}}(t) \qquad (9)$$

where $\boldsymbol{Q}$ is a positive semi-definite Hessian of the state vector error $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_{\mathrm{TO}}(t)$ and $\boldsymbol{R}$ is a positive definite Hessian of the control input vector error $\tilde{\boldsymbol{u}}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_{\mathrm{TO}}(t)$. Similarly, the final cost is defined by

$$\phi(\boldsymbol{x}(T_{\mathrm{MPC}})) = \frac{1}{2}\tilde{\boldsymbol{x}}(T_{\mathrm{MPC}})^T \boldsymbol{Q}_{\mathrm{final}}\tilde{\boldsymbol{x}}(T_{\mathrm{MPC}}) \qquad (10)$$

where $\boldsymbol{Q}_{\mathrm{final}}$ is the positive semi-definite Hessian of the final state vector error $\tilde{\boldsymbol{x}}(T_{\mathrm{MPC}}) = \boldsymbol{x}(T_{\mathrm{MPC}}) - \boldsymbol{x}_{\mathrm{TO}}(T_{\mathrm{MPC}})$. For example, Figure 5 shows the optimized state vector $\boldsymbol{x}(t)$ and control input vector $\boldsymbol{u}(t)$ while following the offline trajectory $\boldsymbol{x}_{\mathrm{TO}}(t)$ and $\boldsymbol{u}_{\mathrm{TO}}(t)$ of the terrain-aware TO in Figure 4.

5.1.2 Equations of motion. The system's dynamics in (8b) is based on a kinodynamic model of a (wheeled-)legged robot. In contrast to the EOM of the terrain-aware TO in (22), it defines the SRBD model along with the kinematics for each leg. It is given by

$$\dot{\boldsymbol{\theta}} = \boldsymbol{T}(\boldsymbol{\theta})\boldsymbol{\omega} \qquad (11a)$$

$$\dot{\boldsymbol{p}} = \boldsymbol{R}_{WB}(\boldsymbol{\theta})\boldsymbol{v} \qquad (11b)$$

$$\dot{\boldsymbol{\omega}} = \boldsymbol{I}_{\mathrm{nom}}^{-1}\left(-\boldsymbol{\omega} \times \boldsymbol{I}_{\mathrm{nom}}\boldsymbol{\omega} + \sum_{i=1}^{n_e} \boldsymbol{r}_{E_i}(\boldsymbol{q}_j) \times \boldsymbol{\lambda}_{E_i}\right) \qquad (11c)$$

$$\dot{\boldsymbol{v}} = \boldsymbol{g}(\boldsymbol{\theta}) + \frac{1}{m}\sum_{i=1}^{n_e} \boldsymbol{\lambda}_{E_i} \qquad (11d)$$

$$\dot{\boldsymbol{q}}_j = \boldsymbol{u}_j \qquad (11e)$$

where $\boldsymbol{R}_{WB}(\boldsymbol{\theta}) \in SO(3)$ represents the rotation matrix that projects the components of a vector from the torso frame B to the world frame W, $\boldsymbol{T}(\boldsymbol{\theta})$ is the transformation matrix from

angular velocities in the torso frame B to the Euler angles derivatives in the world frame W, $\boldsymbol{I}_{\mathrm{nom}}$ is the moment of inertia of the COM taken at the robot's nominal configuration $\boldsymbol{q}_{\mathrm{nom}}$, m is the total mass, $\boldsymbol{g}(\boldsymbol{\theta})$ is the gravitational acceleration in torso frame B. In contrast to the SRBD constraint of the terrain-aware TO in (22), the end-effector's contact position $\boldsymbol{r}_{E_i}(\boldsymbol{q}_j)$ of leg i w.r.t. the COM is retrieved through forward kinematics from the optimized joint position vector qj and (11e) needs to be added as an additional equality constraint of the joint velocity vector $\dot{\boldsymbol{q}}_j$.

5.1.3 Legs in contact. The MPC models some of the contact constraints in a similar fashion as the terrain-aware TO in (23), (24), and (25). The optimization variables of the MPC, however, include the kinematic information of each leg, which gives us

friction cone : $\qquad\qquad \boldsymbol{\lambda}_{E_i} \in \mathcal{C}(\boldsymbol{n},\mu_C) \qquad (12a)$

legged robots : $\qquad\qquad \boldsymbol{v}_{E_i}(\boldsymbol{x},\boldsymbol{u}) = \boldsymbol{0} \qquad (12b)$

wheeled $-$ legged robots : $\quad \pi_{E_i,\perp}(\boldsymbol{v}_{E_i}(\boldsymbol{x},\boldsymbol{u})) = 0 \quad (12c)$

$$\boldsymbol{v}_{E_i}(\boldsymbol{x},\boldsymbol{u}) \cdot \boldsymbol{n} = 0 \qquad (12d)$$

where $\mathcal{C}(\boldsymbol{n},\mu_C)$ implements the friction cone as an inequality constraint without the approximation by a friction pyramid, and $\boldsymbol{n}$ is the local surface normal in world frame W. The motion constraint of traditional legged robots in (12b) is modeled through the end-effectors' velocities. When in contact, the velocity $\boldsymbol{v}_{E_i}$ in world frame W of leg i is restricted to stay stationary. As given by (12c) and (12d), wheeled-legged robots can execute motions along the rolling direction, where $\pi_{E_i,\perp}(\cdot)$ is the projection of the end-effector velocity $\boldsymbol{v}_{E_i}$ onto the perpendicular direction of the rolling direction. In contrast to the missing leg kinematic of the terrain-aware TO in (25), the motion constraint of wheeled robots in (12c) and (12d), that is, the velocity along the rolling direction is left unconstrained or more precisely $\pi_{E_i,\parallel}(\boldsymbol{v}_{E_i}(\boldsymbol{x},\boldsymbol{u})) \in \mathbb{R}$, can be easily computed through forward kinematics. We would like to highlight that this motion constraint is the only part that differentiates legged and wheeled-legged locomotion.

The traversal of challenging terrain requires additional inequality constraints that respect safe terrain regions. To this end, we segment the height map $h_{\mathrm{terrain}}(x, y)$ into convex terrain regions given by a number n_h of half-space constraints $\boldsymbol{A}_i \in \mathbb{R}^{n_h \times 3}$ and $\boldsymbol{b}_i \in \mathbb{R}^{n_h}$ of leg i that ensure the leg lands within desired target regions. Also, wheeled-legged robots, as shown in Figure 5, need to respect the constraint while driving on these safe terrain patches. Each leg's convex terrain regions are selected through the state vector of the offline trajectory $\boldsymbol{x}_{\mathrm{TO}}$. The constraint can be formulated as

$$\boldsymbol{A}_i \cdot \boldsymbol{p}_{E_i}(\boldsymbol{x}) + \boldsymbol{b}_i \geq \boldsymbol{0} \qquad (13)$$

where $\boldsymbol{p}_{E_i}(\boldsymbol{x}) \in \mathbb{R}^3$ is the position of the end-effector in world frame W. Moreover, the safety constraint in (13) is

incorporated into the MPC as a control barrier function introduced by Grandia et al. (2021).

5.1.4 Legs in air. Legs in swing phase are not capable of generating ground reaction forces and need to follow collision-free trajectories. This restriction can be incorporated into the MPC as

$$\lambda_{E_i} = \mathbf{0} \tag{14a}$$

$$d_i(\mathbf{p}_{E_i}, t) > c(t) \tag{14b}$$

where the inequality in (14b) is added to the end-effector's cost function in (9) through the relaxed barrier function. The distance of the end-effector i to the closest obstacle is given by $d_i(\mathbf{p}_{E_i}, t)$ while $c(t)$ is a predefined lower bound on collision avoidance. In our work, we convert the height map $h_{\text{terrain}}(x, y)$ into a signed distance field (SDF) using the method of Fankhauser et al. (2018).

5.2 Torque generation

As visualized in Figure 3, we send torque commands to the embedded control loop of each motor. To this end, the optimized control input vector $\mathbf{u}^*$, including its contact forces and joint velocities, is translated through forward simulation into desired accelerations. Then inverse dynamics converts the desired accelerations from this rollout into torques.

6 Trajectory composer

Our final theoretical contribution proposes composing individual offline trajectories from the motion library into a single trajectory at run-time to produce longer and more complex maneuvers, as shown in Figure 12. For this reason, we leverage the MPC in Section 5. Before the motion is sent to the robot, we evaluate each transition to ensure the whole trajectory is feasible. The composed trajectory is then sent to the robot enacting the transitioning motions in real-time. It is important to note that no additional reference trajectories are added to the composition since the MPC can transition between (discontinuous) motions. If necessary, the motion composer adds additional time or a repositioning sequence to decrease the transitioning cost.

6.1 Trajectory composition

As shown in Figure 3, the operator can choose individual motions from the motion library, and the motion composer concatenates these motions into a single (composed) offline trajectory. To this end, an ordered list of motions, that is, individual offline trajectories $\mathbf{y}_k(t) = \begin{bmatrix} \mathbf{x}_{\text{TO},k}^T & \mathbf{u}_{\text{TO},k}^T \end{bmatrix}^T \in \mathcal{M}$ of trajectory k as defined in (2), is commanded by the operator, where $\mathcal{M}$ represents the motion library of individual offline trajectories. The composed offline trajectory generated by the motion composer is given by

$$\mathbf{m}(t) = (\mathbf{t}_1(t), \mathbf{y}_1(t), \mathbf{t}_2(t), \mathbf{y}_2(t), \ldots, \mathbf{t}_N(t), \mathbf{y}_N(t)) \tag{15}$$

where $\mathbf{t}_k(t)$ represents online generated transitions between individual offline trajectories $\mathbf{y}_k(t)$ and N is the number of composed trajectories. The final time horizon of $\mathbf{m}(t)$ is calculated through $T_{\text{TO}} = \sum_{n=1}^{N} (T_{\text{TD},n} + T_{\text{TO},n})$, with the transitioning duration of $\mathbf{t}_k(t)$ given by $T_{\text{TD},k}$.

Transition strategies. In this article, we present three different strategies for the transitions $\mathbf{t}_k(t)$ in (15), as given by:

1. No transition: The transitioning duration $T_{\text{TD},k}$ is set to zero, which invokes no transition between consecutive offline trajectories, that is, $\mathbf{y}_{k-1}(t) \rightarrow \mathbf{y}_k(t)$.

2. Time transition: We add a transitioning time between consecutive offline trajectories $\mathbf{y}_{k-1}(t) \rightarrow \mathbf{t}_k(t) \rightarrow \mathbf{y}_k(t)$ by setting the transition $\mathbf{t}_k(t)$ to be equal to the beginning of the upcoming offline trajectory, that is, $\mathbf{t}_k(t) = \mathbf{y}_k(t = \sum_{n=1}^{k} T_{\text{TD},n} + \sum_{n=1}^{k-1} T_{\text{TO},n})$, where the transitioning duration $T_{\text{TD},k}$ is set to 1 s in our work. The second strategy of only adding some transition time allows the robot to modify it's starting state and dampen any residual velocities from the previous trajectory.

3. Repositioning transition: We add a transitioning maneuver before scheduling the next trajectory. To this end, we choose a trotting gait that repositions the whole-body of the robot to the desired state of the following trajectory. Similar to the previous strategy, the transition $\mathbf{t}_k(t)$ is equal to the beginning of the upcoming offline trajectory, that is, $\mathbf{t}_k(t) = \mathbf{y}_k(t = \sum_{n=1}^{k} T_{\text{TD},n} + \sum_{n=1}^{k-1} T_{\text{TO},n})$, where the transitioning duration $T_{\text{TD},k}$ is set to 1 s in our work. The second strategy may not lead to optimal transitions if the following trajectory requires a substantially different starting state, for example, the legs' contact positions are too far apart, as shown in Figure 6. Hence, the third strategy places a repositioning maneuver before scheduling the following trajectory. By utilizing the MPC to transition using a known periodic gait, there is some guarantee that the online MPC will solve the following optimization problem with minimal latency in issuing the command instead of using the offline TO.

Transitioning cost evaluation. The three transitioning strategies are evaluated based on a transitioning cost, which is assessed online before the operator's motion command is sent to the real robot. To this end, we evaluate the MPC's solution of (8) over one iteration for each transition in (15) considering the previous and following trajectory, that is, the trajectory between consecutive trajectories $\mathbf{y}_{k-1}(t) \rightarrow \mathbf{t}_k(t) \rightarrow \mathbf{y}_k(t)$, and between the initial (measured) state $\mathbf{x}_0$ of the robot and the first trajectory $\mathbf{x}_0 \rightarrow \mathbf{t}_1(t) \rightarrow \mathbf{y}_1(t)$. The algorithm chooses the transition strategy that achieves the lowest cost given the previous and following trajectory.
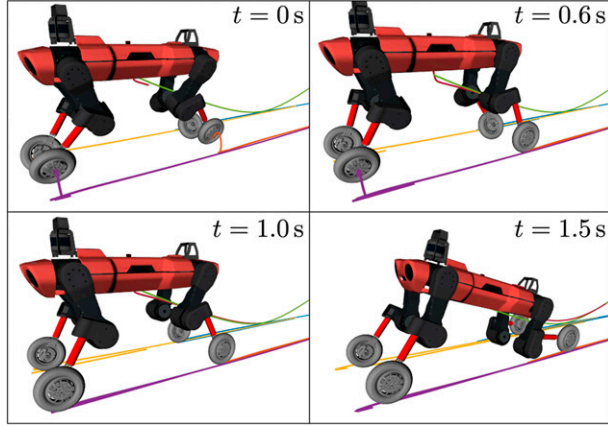
Figure 6. Ducking motion based on the interactive TO in Section 4.2. Here, the initial state of the end-effectors at $t = 0$ s is too far away from the offline trajectory. Thus, the robot performs a trotting gait to reposition its legs, that is, first the right-front and left-hind legs reposition at $t = 0.6$ s, and then the left-front and right-hind legs reposition at $t = 1$ s. The visualization shows the torso and end-effector trajectory of the offline trajectory and the MPC's solution.

6.2 Alignment with initial state

The composed trajectory $\boldsymbol{m}(t)$ in (15) is aligned with the robot's initial state $\boldsymbol{x}_0$. Accordingly, the alignment considers the torso's position offsets in the plane and it's heading orientation given by $p_{0,x}$, $p_{0,y}$ and $\theta_{0,\text{yaw}}$. To align the motion's height, we generate the average terrain height through the robot's feet positions, that is, $p_{0,z} = (1/n_e)\sum_{i=1}^{n_e} r_{E_i,z}$. Finally, the transformation matrix $\boldsymbol{T}_0$ aligning the final offline trajectory $\boldsymbol{m}(t)$ with the robot's initial state $\boldsymbol{x}_0$ is given by

$$\boldsymbol{T}_0 = \begin{bmatrix} \cos(\theta_{0,\text{yaw}}) & -\sin(\theta_{0,\text{yaw}}) & 0 & p_{0,x} \\ \sin(\theta_{0,\text{yaw}}) & \cos(\theta_{0,\text{yaw}}) & 0 & p_{0,y} \\ 0 & 0 & 1 & p_{0,z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (16)$$

7. Experiments

To evaluate our novel locomotion controller, we conducted a series of quantitative experiments with agile maneuvers at the robot's actuation limits. To the best of our knowledge, the roller-walking robot's hybrid motions have not been shown before in literature. All experiments were conducted with either the interactive, terrain-aware, or sampling-based TO in Section 4.2, 4.3, and 4.4, while the MPC in Section 5 performed all behaviors with the same parameter set. In the case of TO, the motion designers only varied task-specific parameters like goal position(s) and the task's time horizon TTO. Table 1 summarizes the four motion planners and lists their capabilities. The following sections report on experiments conducted with ANYmal (see also

Figures 1 and 2). A video available at https://youtu.be/ 39rRhTqcQc0 showing the results accompanies this article.

7.1 Experimental setup

Our trajectory composer, online MPC, inverse dynamics, and state estimator run in concurrent threads on a single PC (Intel i7-8850H, 2.6 GHz, Hexa-core, 64-bit). All offline trajectories are computed on a laptop (Intel E3-1505MV6, 3.0 GHz, Quad-core, 64-bit) and stored in a motion library on the robot. The robot is entirely self-contained in computation and sensing.

The estimation of the robot's state and the torque commands are computed together in a 400 Hz loop. The former requires the fusion of the inertial measurement unit (IMU) readings and the kinematic measurements from each actuator to acquire the robot's state, that is, the pose of the torso B w.r.t. the world frame W, as described by Bloesch et al. (2013). To this end, each leg's contact state is determined by estimating the contact force, which considers the measurements of the motor drives and the full-rigid body dynamics.

7.2 Results and discussion

In the following, we verify our six contributions (C1) to (C6) introduced in Section 1.2 by categorizing each result and discussion as part of its underlying contribution.

(R1) Whole-body coordination. The algorithms in Table 1 are mostly based on single optimization problems (except RRT & TO) optimizing over the whole-body trajectory, including the base pose, end-effector motion, contact force, and two of the algorithms also optimize the joint motion, while the terrain-aware TO computes gait timings. Thanks to this single optimization of the whole-body, the robot can coordinate complex and artistic motions, as shown in Figures 1 and 2, operating near robot limits and cumbersome to hand-craft through heuristics. The motion over the steps includes front-legs and hind-legs jumps at a speed of up to 1.5 m/s (see Figure 7). When ducking under a table, the optimization algorithm can discover specialized motions for our wheeled-legged robot. Here, the robot reaches its maximum torque limits of the hip motors. The turning motion includes a complex coordination of all joints, as shown in Figure 9. We further discuss the motion over the step and the turning motion in the following paragraphs.

The first maneuver in Figure 8 presents the motion of the base and end-effectors over a 0.2 m step. Here, it can be seen how the MPC's solution corrects the offline swing trajectory over the step preferring collision-free trajectories. The result shows that the MPC uses continuous optimization to correct mistakes from the offline trajectory.

The dynamic turning motion in Figure 1 represents a challenging motion and requires the coordination of all DOF. In particular, the robot needs to step and turn the torso

Table 1. Capabilities of our presented interactive TO, terrain-aware TO, combined sampling and optimization-based planner, and online MPC. Table design is adapted from Winkler (2018).

	Interactive TO	Terrain-aware TO	RRT & TO	Online MPC
Dynamic model (accuracy)	CD model	SRBD model	Kinematic model	Kinodynamic model
Number of optimizations	Single optimization	Single optimization	Two optimizations	Single optimization
Optimization time	>1 s	>1 s	1 s	20–50 ms
Time horizon	>1 s	>1 s	>1 s	1 s
Optimized components				
Base motion	6D	6D	6D	6D
End-effector motions	3D	3D	3D	3D
Contact forces	✓	✓	✗	✓
Joint motions	✓	✗	✓	✓
Step timing/sequence	✗	✓	✓	✗
Difficulty of shown task				
Line/point contacts	✓	✓	✗	✓
Flight phases	✓	✓	✗	✓
Non-flat terrain	✗	✓	✓	✓
Accurate rolling constraint	✓	✗	✓	✓
Adaptation of joint momentum	✓	✗	✗	✗
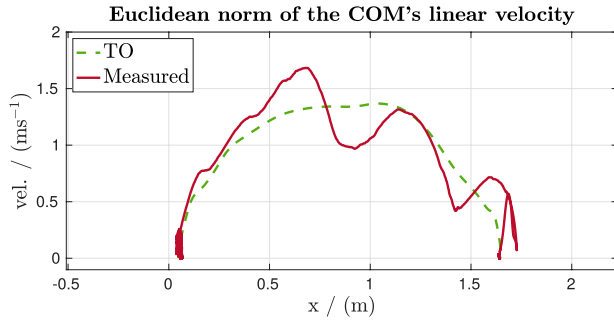
Figure 7. Results of the terrain-aware TO in combination with the MPC while executing the motion over the step in Figure 1. The plot shows the euclidean norm of the COM's linear velocity over the COM's forward position in world frame W. The MPC's solution equals the robot's measured state, that is, the measured velocity is equal to v ($t = 0$) due to the MPC's initialization after every iteration with the robot's measured state. The equivalent position profiles of the COM and end-effectors can be obtained in Figure 8.

at the same time while respecting all physical constraints. To this end, the TO discovers a hybrid locomotion strategy since the roller-walking robot ANYmal does not incorporate any steering mechanism for its wheels. To achieve the fast turning motion over almost 180°, the robot coordinates the legs in contact onto a common wheel axis line, enabling it to turn through actuating its wheels. Figure 9 shows the comparison of the offline trajectory and the MPC's solution over the task's time horizon represented by the robot's estimated state. The plot represents only a fraction of all whole-body states tracked through the MPC's cost function in Section 5.1.1. With a single parameter set Q, R, and Q_{final}, the robot can successfully execute all shown motions and
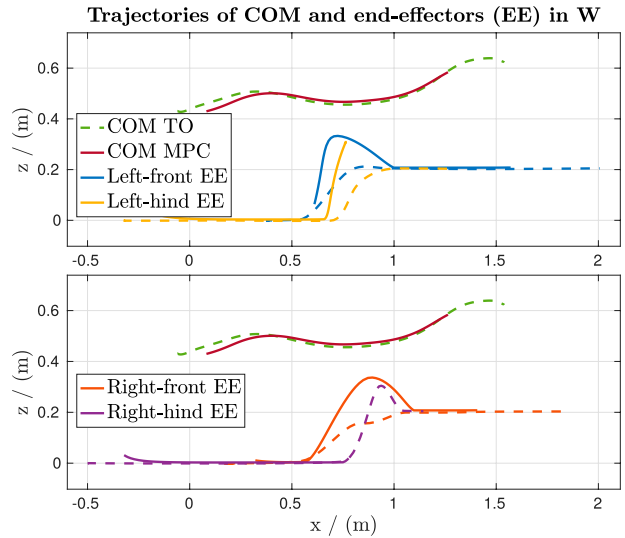
Figure 8. Results of the terrain-aware TO in combination with the MPC while executing the motion over the step in Figure 1. The plots show the two optimization problems' motion, that is, the torso and the four end-effector trajectories. The dotted lines are the offline generated trajectories, while the solid lines represent the MPC's solution over a one-second horizon at a time when the left-front end-effector lifts its leg. The equivalent COM's speed profile can be obtained in Figure 7.

weigh the high-dimensional tracking task, including positions, rotations, linear velocities, angular velocities, joint positions, joint velocities, and forces.

(R2) Offline-to-online gap. One of the main challenges of running offline trajectories is the online reaction to unforeseen conditions. The key to this challenge is our online MPC that robustifies the maneuver to a certain degree by recomputing solutions on the fly and adding feedback
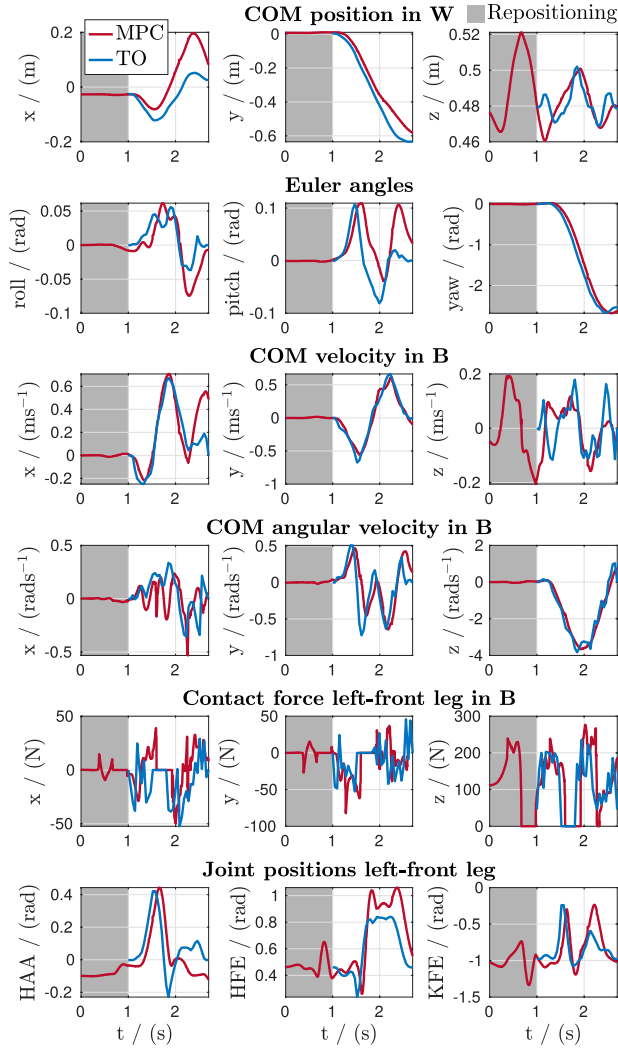
Figure 9. Results of the interactive TO in combination with the MPC while executing a repositioning motion (0–1 s) and the turning motion in Figure 1 (1–2.7 s). The plots compare the optimized whole-body trajectory of both algorithms. Here, the MPC solution is represented by the robot's measured state, which is the equivalent of the initial state vector x ($t = 0$) and initial control input vector u ($t = 0$). This equivalency is due to the MPC's fast update rate and the reinitialization of its optimization problem after every iteration with the robot's measured state. At $t = 1s$, the online motion planner in red smoothly blends between a discontinuity of the commanded offline trajectory in blue.

control. As shown in Table 1, the MPC recomputes optimal solutions on a 20–50 Hz loop while looking-ahead the offline trajectory over a one-second horizon. In the following, we present the results that show how the recomputation of optimal solutions can benefit the online execution of offline trajectories.

First, Figure 10 shows six scenarios where the robot recovers from unforeseen events. In all cases, the online MPC reacts to these events and successfully executes the offline trajectory. Such unexpected scenarios can either occur from mistakes that occur during offline motion

prototyping or an unpredictable disturbance throughout the online execution. For example, the former is shown in the top-left and middle-right image, which shows two scenarios of incorrect modeling of the obstacle and the robot's collision body, respectively. The latter is exemplified in the top-right, middle-left, lower-left, and lower-right images that exemplify the terrain's misalignment, slippage, and purposely misplaced obstacles. Our approach can react to such unforeseen disturbances while anticipating future events of the offline trajectory. With these anticipating skills and the feedback control, the MPC can rescue the robot from these unforeseen situations.

Second, the MPC's look-ahead also improves the tracking of offline trajectories in the presence of discontinuities, which might occur at the transitions to the offline trajectories. The MPC can blend between two motions and converge to the offline trajectory. For example, in Figure 9, the vertical movement along the z-axis of the COM's position and velocity experiences a discontinuity at $t = 1$ s, where the robot switches from a repositioning motion, as described in Section 6, to the offline trajectory. It can be seen how the red trajectory anticipates along its receding horizon the upcoming offline trajectory and finds a solution respecting the whole-body state.

The last experiment regarding the offline-to-online gap further discusses the importance of incorporating a planning horizon into the online optimization problem. In contrast to a naive trajectory tracking that only looks at one set point at a time, for example, Kim et al. (2019), our approach can react to unforeseen conditions while looking ahead at the offline trajectory. With these anticipating skills, the MPC adds feedback to the execution of offline trajectories, as shown in Figure 10. Before presenting quantitative results, we first discuss our qualitative findings comparing the set-point-only tracking approach of offline trajectories. When running on the real robot, a tracking controller optimizing over set points has two different strategies in terms of the offline trajectory's sampling frequency: (1) The offline trajectory's sampling frequency equals the tracking controller's update rate. (2) Sampling frequencies lower than the update rate require interpolations between the set points. The former strategy comes with longer optimization times of the offline TOs since the tracking controller's update rates are synchronized with the actuator's command loop (in our case, 400 Hz). Thus, this approach constrains the motion designer in terms of optimization speed and flexibility. The latter strategy comes with the caveat that the interpolated motion can not guarantee to be kinematically or dynamically feasible. In addition to the sampling frequency, Medeiros et al. (2020) experience that a naive trajectory tracking can not handle large tracking offsets. Therefore, the offline TO needs to be initialized with the robot's measured state, which ties the motion designer down to real experiments next to the robot. In contrast, the MPC is flexible w.r.t. the offline trajectories' sampling frequency and starting state. Figure 11 displays the tracking performance and disturbance rejection of a naive trajectory tracking

developed by Bjelonic et al. (2019) compared to our proposed MPC. The MPC can handle the disturbance without falling, and as can be seen in the figure's caption, the MPC offers flexibility for the motion designer.

(R3) Rapid motion prototyping. As discussed in the paragraph before, the MPC decouples the locomotion problem into an offline and online computation. With this decoupling, six motion designers throughout this work could iterate over behavior designs offline without tuning robot experiments, which sped up the creation of new motions rapidly. We developed 30 motions throughout this work using the terrain-aware TO, 53 maneuvers based on the interactive TO, and six based on the sampling and optimization-based planner that run successfully on the real robot. Motion designers can craft new trajectories in a matter of minutes. We also experienced 16 maneuvers that had no success on our ANYmal robot, which is mostly due to motions at the torque limits, for example, motions over high steps with a speed over 2 m/s using the terrain-aware TO and too fast turning motions developed by the interactive TO. Twelve of the unsuccessful maneuvers, however, can be identified in our two steps verification process before running it on the hardware. The first step checks the offline trajectory's feasibility in combination with the MPC using a visualization without any modeling errors and contact models. If feasible, the motion designers verify the maneuver in a simulation environment based on the Open Dynamics Engine (ODE) by Smith et al. (2005). This verification process helps the motion designer to iterate over behavior designs offline. None of the successful motions require retuning any of the MPC's parameters, which frees the motion designer from adjusting the real robot's performance.

(R4) Motion composition. Multiple offline trajectories can be composed into a single maneuver resulting in trajectories with longer time horizons. For example, Figure 12 shows the compositions of a ducking motion, a dynamic 90°turn, and another ducking motion. With a total duration of around 8 s, the motion composition's benefit is displayed in this obstacle curse consisting of two tables. By combining multiple offline trajectories, the robot can find optimal maneuvers in terms of the robot's physical constraints over a longer time horizon.

Transitioning between multiple trajectories requires appropriate transitions and the ability to blend between transitions and discontinuities smoothly. As already introduced before, Figure 9 shows a transition at $t = 1$ s between a repositioning gait and a dynamic turning motion. The whole-body trajectory shows that the MPC can anticipate future events and smoothly blend between both motions.

The video that accompanies this article shows a composed dancing trajectory with a total time horizon of 11 s. The motion is composed of eleven individual offline trajectories with a time horizon of 1 s per trajectory. Here, the motion composer did not add any transitions between the individual motions.

Figure 10. Recovery of the online MPC to unforeseen events. In all situations, the MPC could successfully recover and finish the executed offline trajectories. Top-left image: The motion designer did not correctly model the platform's width, and as such, the robot's front-left leg fell from the platform. Top-right image: While stepping up two steps, the end-effector moved the top platform since it was not properly secured. Due to this misalignment of the terrain, the front legs fell from the platform. Middle-left image: The front part of the torso hit the table while executing a dynamic 90°turn. Middle-right image: The hind legs' knee protectors collided with the ground during a ducking motion since the offline trajectory did not consider the protector's collision model. Lower-left image: The robot faced an unmodeled disturbance (wooden incline) on the ground while executing an offline trajectory assuming flat terrain. Lower-right image: The snapshot shows a time instance when the hind legs overcome the step. Here, the front legs slipped on the platform, and the front part of the torso collided with the platform.

One of the main challenges that occur during the online execution of composed offline trajectories is the state estimation's drift that accumulates along the offline trajectory. On flat terrain, we can execute dancing motions of up to 25 s without any noticeable drop in performance. The drift, however, becomes more crucial when executing motions over obstacles since the lift-off timings need to be synchronized with the terrain. In our case, the robot can successfully complete dynamic motions over obstacles with a time horizon of $T_{TO} = 5$ s, as shown in Figure 2, and the maximum distance or time of the offline trajectories depends on the state estimator's drift. For even longer offline trajectories, future work can study possible morphing strategies of the offline trajectories based on updated sensor data, as described by Kuindersma (2020).

(R5) Performance and generalization. With our novel framework, the robot can execute motions over challenging obstacles, unique motions through confined spaces, dynamic motions at the robot's limits, and artistic dance moves. Each of the offline trajectories out of the motion
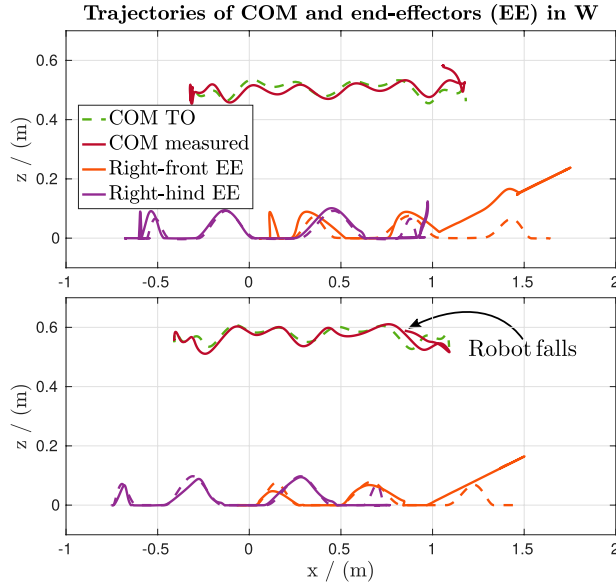
Figure 11. Comparison between a naive trajectory tracking (Bjelonic et al., 2019) over one set point at a time (lower plot) and the MPC over a receding horizon of 1 s (upper plot) while commanding a hybrid trotting gait through the terrain-aware TO. In this scenario, the robot needs to deal with an unmodeled inclination at $x = 1$ m, as shown in the lower-left image of Figure 10. The plots show the offline and measured trajectories, that is, the torso and two end-effector trajectories. The dotted lines are the offline generated trajectories, while the solid lines represent the online solutions. The MPC handles the disturbance and successfully finishes the motion, while the naive tracking controller makes the robot fall. Besides, the offline trajectories' sampling frequency of the naive tracking controller requires 400 Hz, and its starting state has to coincide with the robot's starting state. In contrast, the MPC is flexible w.r.t. the offline trajectories' sampling frequency and starting state.

library can be composed into a single maneuver. To the best of our knowledge, this work shows unprecedented motions on a roller-walking robot, as shown in Figures 1 and 2.

Our approach is not restricted to wheeled-legged robots, and, thus, in this experiment, we verify the generalizability of the approach to a traditional legged robot. Figure 13 shows the trajectory of ANYmal without wheels overcoming a step. The comparison to the wheeled-legged version in Figure 8 shows how the kinematic constraint influences the end-effector's optimal trajectories. The legged robot requires five steps per leg to overcome the obstacle, while the wheeled-legged robot can use its wheels and only step once per leg. We want to highlight that also, in the legged robot's case, the MPC can recover by optimizing the motion of the whole-body even in the presence of slippage, as presented in the figure's caption.

(R6) Evaluation of offline motion planners. The experiments show the execution of offline trajectories over longer time horizons generated through the interactive, terrain-aware TO, and sampling and optimization-based planner. The online MPC over shorter time horizons closes the offline-to-online gap. The following section discusses the

Figure 12. Motion composition of three offline trajectories based on the interactive TO in Section 4.2. The upper visualization shows the torso and end-effector trajectory of the composed trajectory $\boldsymbol{m}(t)$. Here, the robot executes two ducking motions with a duration of 2 s each, connected by a 90°turn with a duration of 1.7 s and two repositioning transitions. (see lower images).

performance on our robot ANYmal of each algorithm, while a summary of the algorithm's performance is presented in Table 1 with a quantitative study in (R1) to (R5).

The interactive TO considers the robot's CD model in (17) and finds motion plans that satisfy kinematic and dynamic constraints over flat terrain. With this higher-dimensional model compared to the SRBD, the offline trajectories incorporate more dynamic motions by respecting the inertia's change through the joint movement. For example, the dynamic turning motions in Figures 1 and 9 achieve a turning rate of 4 rad/s, which sets a new record on our roller-walking robot. The accompanying video's evaluation of the motion reveals that the robot coordinates a complex, whole-body trajectory. As described by Geilinger et al. (2018), the interactive TO comes with a suite of user-guided computational tools that support manual, semi-automatic, and fully automatic optimization of the robot's trajectories. We demonstrate the method's effectiveness by creating various unique motions for our robot ANYmal, for example, complex turning motions, unique motions through confined spaces (see Figure 12), and dance moves. Especially the generation of the dance motions reveals the fast and interactive capabilities of the approach. In minutes, the motion designer can generate dynamic motions, including a "moonwalk" that is synchronized with the song's pace. The optimization problem, however, does not consider non-flat terrain, and the motion designer needs to provide the gait sequences and timings.

In contrast, the terrain-aware TO does not provide user-guided computational tools that support manual or semi-automatic optimization. The algorithm operates fully automatic by giving a starting and a goal pose, optimizing motions over non-flat terrain (see Figures 2, 8, and 13). As shown in Figure 2, our roller-walking robot ANYmal achieves
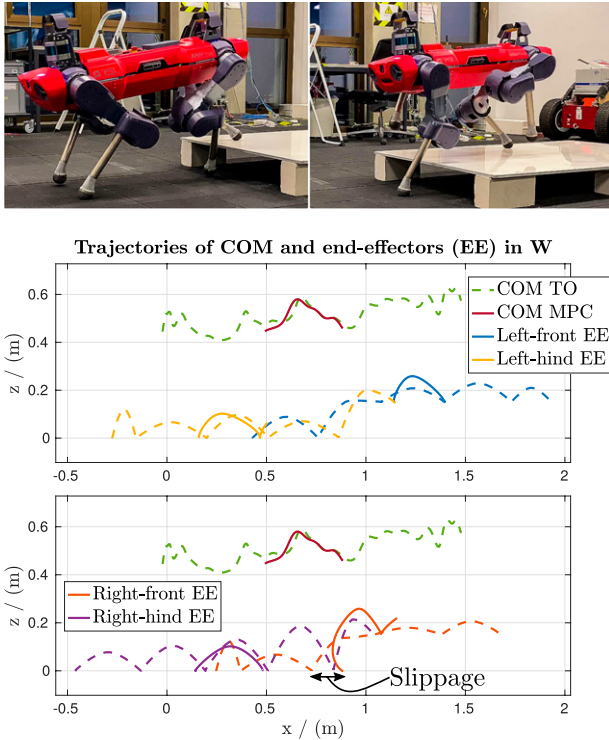
Figure 13. Results of the terrain-aware TO in combination with the MPC while executing the motion over the step with our traditional legged robot. The plots show the two optimization problems' motion, that is, the torso and the four end-effector trajectories. The dotted lines are the offline generated trajectories. In contrast, the solid lines represent the MPC solution over a one-second horizon at a time when the left-front end-effector touches down on the step, as shown in the top-left image. Here, the right-front end-effector slipped and landed too close to the step. The MPC can deal with these situations by optimizing the whole-body's motion, while considering the future trajectory of the offline TO.

Figure 14. Our quadrupedal robot ANYmal with wheels overcomes stepping stones (upper images) and steps up-and-down (lower images) through the sampling and optimization-based approach. All motions are statically stable, that is, the robot only lifts one leg at a time.

impressive results over a set of steps with a maximum speed of 1.5 m/s. Compared to the CD model of the interactive TO, this is achieved with a low-dimensional SRBD model. Still, the algorithm optimizes over the gait timings through a phase-based parametrization, as described in Appendix B and in more detail by Winkler et al. (2018). The optimization, however, is still sensitive to the initial gait schedule.

This sensitivity can be solved with our final offline motion generator. As shown in Figure 14, we verify for the first time the combination of sampling and optimization-based planning of Jelavic et al. (2021) on a real machine. With this combination, gait timings and sequences can be easily obtained through the RRT without any human supervision. The sampling-based TO can only generate statically stable motions, while the MPC refines these offline trajectories with a kinodynamic model. Furthermore, the results verify that our online motion planner is capable of handling offline trajectories that only contain kinematic information.

Our short-term motion planner is the online MPC that acts as a general-purpose algorithm integrating offline trajectories from any TO algorithm. Previous publications (Winkler et al., 2018; Medeiros et al., 2020) introducing the terrain-aware TO incorporate a naive tracking approach. With the incorporation of online MPC in our article, we finally manage to run these offline trajectories reliably on the robot even in the presence of disturbances and show agile motions on our robot ANYmal. In our previous work (Bjelonic et al., 2021), the MPC only considers future states up to a 1 s time horizon, which leads to unnecessary steps over obstacles. The video in https://youtu.be/_rPvKlvyw2w highlights this issue over stairs and steps. Combining offline and online optimization over the full-time horizon results in more optimal behavior in terms of speed and efficiency, as shown in Figure 2. Due to the task-generic properties of the MPC, any further development on the MPC benefits the execution of all offline trajectories.

8. Conclusions and future work

We present a method for the whole-body coordination of robotic platforms with legs and wheels, using TO to generate offline trajectories for complex motions and online MPC for continuous optimization along the offline trajectory. Our set of experiments involves complex locomotion maneuvers over challenging obstacles and at the robot's limits. These experiments verify that our method can execute dynamic offline trajectories on the real robot even in the absence of perfect knowledge of the environment and under unforeseen conditions, such as modeling errors and external disturbances. Unlike most offline-to-online optimization methods, our approach incorporates an MPC capable of reacting to these disturbances while anticipating future events of the offline trajectory. Our experimental

results demonstrate that our method can effectively execute a wide range of maneuvers, including fast motions over challenging obstacles, unique motions through confined spaces, dynamic motions at the robot's limits, and artistic dance moves. The results also show that our method uses continuous optimization through MPC to correct mistakes from the offline trajectory and unpredictable disturbances. The reliable execution of offline trajectories enables motion designers to iterate over behavior designs offline without tuning robot experiments, allowing them to author new behaviors rapidly. Also, individual offline maneuvers can be composed into a single long-time horizon maneuver.

At a more general level, our work explores the synchronization of long-time horizon maneuvers with short-time horizon plans. We have chosen optimization-based approaches that generate trajectories over both look-ahead horizons. A promising direction for future experiments is to analyze how humans and animals incorporate look-ahead planning stages to decide on the upcoming steps. Maneuvers over challenging obstacles and at the physical limits are especially demanding, requiring look-ahead planning to decide on the upcoming steps carefully. In this regard, a hierarchical learning-based approach can be compared with our proposed method.

As indicated throughout the article, a human operator chooses maneuvers out of a motion library, suggesting that the automation of this stage could quickly increase the robot's autonomy. A fascinating avenue for future work, therefore, could be the continuous computation of long-time horizon plans. Since our method can deal with unpredictable disturbances over the horizon of several offline trajectories, the update rate of the TO is not subjected to real-time constraints. With our framework, we can explore alternative algorithms, including mixed-integer optimizations, that simultaneously solve the whole-body trajectory, gait sequences, and timings, even if this complexity increases the solver time.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ORCID iDs

Marko Bjelonic   https://orcid.org/0000-0002-9123-3920
Oliver Harley   https://orcid.org/0000-0002-0476-1072

Notes

1. In this article, a whole-body approach or whole-body optimization refers to the simultaneous generation of the robot's contact forces, generalized coordinates and generalized velocities, including the torso's pose, linear and angular velocity, and the joint coordinates and velocities. The offline trajectories fed into the online MPC and the MPC's re-optimization consider the robot's whole-body.

2. We let the end-effector be fixed at a leg's endpoint, that is, the point on the wheel or foot in contact with the ground during stance, and define this point as a leg's end-effector.

3. The C++ implementation of the MPC's solver is publicly available through https://github.com/leggedrobotics/ocs2

References

Aceituno-Cabezas B, Mastalli C, Dai H, et al. (2017) Simultaneous contact, gait and motion planning for robust multi-legged locomotion via mixed-integer convex optimization. *IEEE Robotics and Automation Letters*.

Bellegarda G, van Teeffelen K and Byl K (2018) Design and evaluation of skating motions for a dexterous quadruped. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1703–1709.

Bellicoso CD, Jenelten F, Gehring C, et al. (2018) Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots. *IEEE Robotics and Automation Letters* 3(3): 2261–2268.

Bertsekas DP (2005) Dynamic programming and suboptimal control: a survey from adp to mpc. *European Journal of Control* 11(4–5): 310–334.

Betts JT (1998) Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics* 21(2): 193–207.

Bjelonic M, Bellicoso CD, de Viragh Y, et al. (2019) Keep rollin' - whole-body motion control and planning for wheeled quadrupedal robots. *IEEE Robotics and Automation Letters* 4(2): 2116–2123.

Bjelonic M, Grandia R, Harley O, et al. (2021) Whole-body mpc and online gait sequence generation for wheeled-legged robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Bjelonic M, Sankar PK, Bellicoso CD, et al. (2020) Rolling in the deep – hybrid locomotion for wheeled-legged robots using online trajectory optimization. *IEEE Robotics and Automation Letters* 5(2): 3626–3633.

Bledt G and Kim S (2019) Implementing regularized predictive control for simultaneous real-time footstep and ground reaction force optimization. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6316–6323.

Bledt G, Powell MJ, Katz B, et al. (2018) Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2245–2252.

Bloesch M, Hutter M, Hoepflinger MA, et al. (2013) State estimation for legged robots-consistent fusion of leg kinematics and imu. *Robotics* 17: 17–24.

Boston Dynamics (2019) More parkour atlas. https://youtu.be/_sBBaNYex3E

Buchanan R, Wellhausen L, Bjelonic M, et al. (2020) Perceptive whole-body planning for multilegged robots in confined spaces. *Journal of Field Robotics*.

Budhiraja R, Carpentier J and Mansard N (2019) Dynamics consensus between centroidal and whole-body models for locomotion of legged robots. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 6727–6733.

Carius J, Ranftl R, Koltun V, et al. (2019) Trajectory optimization for legged robots with slipping motions. *IEEE Robotics and Automation Letters* 4(3): 3013–3020.

Caron S and Pham Q (2017) When to make a step? tackling the timing problem in multi-contact locomotion by topp-mpc. In: IEEE-RAS International Conference on Humanoid Robotics, pp. 522–528.

Caron S, Pham QC and Nakamura Y (2017) Zmp support areas for multicontact mobility under frictional constraints. *IEEE Transactions on Robotics* 33(1): 67–80.

Dai H, Valenzuela A and Tedrake R (2014) Whole-body motion planning with centroidal dynamics and full kinematics. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids). pp. 295–302.

Dantec EL, Budhiraja R, Roig A, et al. (2020) Whole body model predictive control with a memory of motion: experiments on a torque-controlled talos. Working paper or preprint.

de Viragh Y, Bjelonic M, Bellicoso CD, et al. (2019) Trajectory optimization for wheeled-legged quadrupedal robots using linearized zmp constraints. *IEEE Robotics and Automation Letters* 4(2): 1633–1640.

Deits R and Tedrake R (2014) Footstep planning on uneven terrain with mixed-integer convex optimization. In: IEEE-RAS International Conference on humanoid robots (Humanoids), pp. 279–286.

Di Carlo J, Wensing PM, Katz B, et al. (2018) Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1–9.

Diehl M, Ferreau HJ and Haverbeke N (2009) Efficient numerical methods for nonlinear mpc and moving horizon estimation. In: *Nonlinear Model Predictive Control*. Berlin: Springer, 391–417.

Englsberger J, Werner A, Ott C, et al. (2014) Overview of the torque-controlled humanoid robot toro. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 916–923.

Erez T, Lowrey K, Tassa Y, et al. (2013) An integrated system for real-time model predictive control of humanoid robots. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 292–299.

Erez T, Tassa Y and Todorov E (2012) Infinite-horizon model predictive control for periodic tasks with contacts. *Robotics: Science and Systems* VII: 73.

Fankhauser P, Bjelonic M, Dario Bellicoso C, et al. (2018) Robust rough-terrain locomotion with a quadrupedal robot. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5761–5768.

Farshidian F, Neunert M, Winkler AW, et al. (2017) An efficient optimal planning and control framework for quadrupedal locomotion. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 93–100.

Focchi M, del Prete A, Havoutis I, et al. (2017) High-slope terrain locomotion for torque-controlled quadruped robots. *Autonomous Robots* 41(1): 259–272.

Focchi M, Orsolino R, Camurri M, et al. (2020) *Heuristic Planning for Rough Terrain Locomotion in Presence of External Disturbances and Variable Perception Quality*. Cham: Springer International Publishing, 165–209.

Geilinger M, Poranne R, Desai R, et al. (2018) Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels. *ACM Transactions on Graphics (TOG)* 37(4): 160.

Geilinger M, Winberg S and Coros S (2020) A computational framework for designing skilled legged-wheeled robots. *IEEE Robotics and Automation Letters* 5(2): 3674–3681.

Goldenberg A, Benhabib B and Fenton R (1985) A complete generalized solution to the inverse kinematics of robots. *IEEE Journal on Robotics and Automation*.

Grandia R, Farshidian F, Dosovitskiy A, et al. (2019a) Frequency-aware model predictive control. *IEEE Robotics and Automation Letters* 4(2): 1517–1524.

Grandia R, Farshidian F, Ranftl R, et al. (2019b) Feedback mpc for torque-controlled legged robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4730–4737.

Grandia R, Taylor AJ, Ames AD, et al. (2021) Multi-layered safety for legged robots via control barrier functions and model predictive control. In: Under Review for IEEE International Conference on Robotics and Automation (ICRA).

Griffin RJ, Wiedebach G, McCrory S, et al. (2019) Footstep planning for autonomous walking over rough terrain. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 9–16.

Hargraves C and Paris S (1987) Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics* 10(4): 338–342.

Henze B, Ott C and Roa MA (2014) Posture and balance control for humanoid robots in multi-contact scenarios based on model predictive control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3253–3258.

Herdt A, Diedam H, Wieber PB, et al. (2010) Online Walking Motion Generation with Automatic Foot Step Placement. *Advanced Robotics* 24(5–6): 719–737.

Herzog A, Rotella N, Schaal S, et al. (2015) Trajectory generation for multi-contact momentum control. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 874–880.

Hutter M, Gehring C, Lauber A, et al. (2017) Anymal-toward legged robots for harsh environments. *Advanced Robotics* 31(17): 918–931.

Jelavic E, Farshidian F and Hutter M (2021) Combined sampling and optimization based planning for legged-wheeled robots. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 8366–8372.

Jelavic E and Hutter M (2019) Whole-body motion planning for walking excavators. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2292–2299.

Jenelten F, Miki T, Vijayan AE, et al. (2020) Perceptive locomotion in rough terrain – online foothold optimization. *IEEE Robotics and Automation Letters* 5(4): 5370–5376.

Kajita S, Kanehiro F, Kaneko K, et al. (2001) The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In: Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180), pp. 239–246.

Kalakrishnan M, Buchli J, Pastor P, et al. (2010) Fast, robust quadruped locomotion over challenging terrain. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 2665–2670.

Kalakrishnan M, Buchli J, Pastor P, et al. (2011) Learning, planning, and control for quadruped locomotion over challenging terrain. *The International Journal of Robotics Research* 30(2): 236–258.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research* 30(7): 846–894.

Kim D, Di Carlo J, Katz B, et al. (2019) Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. arXiv preprint arXiv:1909.06586.

Klamt T and Behnke S (2018) Planning hybrid driving-stepping locomotion on multiple levels of abstraction. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1695–1702.

Koenemann J, Del Prete A, Tassa Y, et al. (2015) Whole-body model-predictive control applied to the hrp-2 humanoid. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3346–3351.

Krause M, Englsberger J, Wieber PB, et al. (2012) Stabilization of the capture point dynamics for bipedal walking based on model predictive control. *IFAC Proceedings Volumes* 45(22): 165–171.

Kudruss M, Naveau M, Stasse O, et al. (2015) Optimal control for whole-body motion generation using center-of-mass dynamics for predefined multi-contact configurations. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids). pp. 684–689.

Kuindersma S (2020) Recent progress on atlas, the world's most dynamic humanoid robot. https://youtu.be/EGABAx52GKI

Kuindersma S, Deits R, Fallon M, et al. (2016) Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots* 40(3): 429–455.

Lapp T and Singh L (2004) Model predictive control based trajectory optimization for nap-of-the-earth (noe) flight including obstacle avoidance. In: Proceedings of the 2004 American Control Conference, pp. 891–896 vol.1.

Laurenzi A, Hoffman EM and Tsagarakis NG (2018) Quadrupedal walking motion and footstep placement through linear model predictive control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2267–2273.

Li H, Frei RJ and Wensing PM (2020) Model hierarchy predictive control of robotic systems. arXiv preprint arXiv:2010.08881.

Magaña OAV, Barasuol V, Camurri M, et al. (2019) Fast and continuous foothold adaptation for dynamic locomotion through cnns. *IEEE Robotics and Automation Letters* 4(2): 2140–2147.

Mason S, Rotella N, Schaal S, et al. (2018) An mpc walking framework with external contact forces. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1785–1790.

Mastalli C, Focchi M, Havoutis I, et al. (2017) Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1096–1103.

Mastalli C, Havoutis I, Focchi M, et al. (2020) Motion planning for quadrupedal locomotion: coupled planning, terrain mapping, and whole-body control. *IEEE Transactions on Robotics* : 1–14.

Mayne D (1966) A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control* 3(1): 85–95.

Mayne DQ, Rawlings JB, Rao CV, et al. (2000) Constrained model predictive control: Stability and optimality. *Automatica* 36(6): 789–814.

Medeiros VS, Jelavic E, Bjelonic M, et al. (2020) Trajectory optimization for wheeled-legged quadrupedal robots driving in challenging terrain. *IEEE Robotics and Automation Letters* 5(3): 4172–4179.

Melon O, Geisert M, Surovik D, et al. (2020) Reliable trajectories for dynamic quadrupeds using analytical costs and learned initializations. *IEEE International Conference on Robotics and Automation (ICRA)*.

Morari M and Lee JH (1999) Model predictive control: past, present and future. *Computers & Chemical Engineering* 23(4–5): 667–682.

Mordatch I, Todorov E and Popović Z (2012) Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph* 31(4).

Naveau M, Kudruss M, Stasse O, et al. (2017) A reactive walking pattern generator based on nonlinear model predictive control. *IEEE Robotics and Automation Letters* 2(1): 10–17.

Neunert M, de Crousaz C, Furrer F, et al. (2016) Fast nonlinear model predictive control for unified trajectory optimization and tracking. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1398–1404.

Neunert M, Stäuble M, Giftthaler M, et al. (2018) Whole-body nonlinear model predictive control through contacts for

quadrupeds. *IEEE Robotics and Automation Letters* 3(3): 1458–1465.

Orin DE, Goswami A and Lee SH (2013) Centroidal dynamics of a humanoid robot. *Autonomous Robots* 35(2–3): 161–176.

Paparusso L, Kashiri N and Tsagarakis NG (2020) A disturbance-aware trajectory planning scheme based on model predictive control. *IEEE Robotics and Automation Letters* 5(4): 5779–5786.

Park C, Park JS, Tonneau S, et al. (2016) Dynamically balanced and plausible trajectory planning for human-like characters. In: ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. pp. 39–48.

Park HW, Wensing P and Kim S (2015) Online planning for autonomous running jumps over obstacles in high-speed quadrupeds. In: *Robotics: Science and Systems*.

Posa M, Cantu C and Tedrake R (2014) A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research* 33(1): 69–81.

Rao AV (2009) A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences* 135(1): 497–528.

Rebula JR, Neuhaus PD, Bonnlander BV, et al. (2007) A controller for the littledog quadruped walking on rough terrain. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1467–1473.

Sardain P and Bessonnet G (2004) Forces acting on a biped robot. center of pressure-zero moment point. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 34(5): 630–637.

Smith R, et al. (2005) Open dynamics engine.

Todorov E (2011) A convex, smooth and invertible contact model for trajectory optimization. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1071–1076.

Tsounis V, Alge M, Lee J, et al. (2020) Deepgait: planning and control of quadrupedal gaits using deep reinforcement learning. *IEEE Robotics and Automation Letters* 5(2): 3699–3706.

Vukobratović M and Borovac B (2004) Zero-moment point – thirty five years of its life. *International Journal of Humanoid Robotics* 1(01): 157–173.

Wieber P (2006) Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 137–142.

Wieber PB, Tedrake R and Kuindersma S (2016) *Modeling and Control of Legged Robots*. Cham: Springer International Publishing, 1203–1234.

Winkler AW (2018) *Optimization-Based Motion Planning for Legged Robots*. PhD Thesis, Zurich: ETH Zurich.

Winkler AW, Bellicoso CD, Hutter M, et al. (2018) Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. *IEEE Robotics and Automation Letters* 3(3): 1560–1567.

Winkler AW, Farshidian F, Pardo D, et al. (2017) Fast trajectory optimization for legged robots using vertex-based zmp constraints. *IEEE Robotics and Automation Letters* 2(4): 2201–2208.

Winkler AW, Mastalli C, Havoutis I, et al. (2015) Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5148–5154.

Yeganegi MH, Khadiv M, Moosavian SAA, et al. (2019) Robust humanoid locomotion using trajectory optimization and sample-efficient learning*. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 170–177.

Zimmermann D, Coros S, Ye Y, et al. (2015) Hierarchical planning and control for complex motor tasks. In: Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '15. New York, NY, USA: ACM, pp. 73–81.

Zucker M, Ratliff N, Stolle M, et al. (2011) Optimization and learning for rough terrain legged locomotion. *The International Journal of Robotics Research* 30(2): 175–191.

Appendix A

Interactive trajectory optimization

In the following, we describe each of the interactive TO's cost functions and constraints and drop the time dependency of the variables introduced in Section 4.2 to improve the equation's readability.

Cost function. At the core of this TO lies its interactive capabilities. The motion designer can formulate a set of objectives that give the user interactive control over the robot's motion. Moreover, the motion designer can add goals y_{goal}, for example, target pose of the robot's torso or any other whole-body motion goal, at a specific time t_{goal}, which creates an objective $(1/2)\left\|y(t = t_{\text{goal}}) - y_{\text{goal}}\right\|_2^2$.

Equations of motion. The global motion is described by the robot's state $x_{\text{cd}}(t)$ and governed by CD, which impose the Newton-Euler equations

$$\dot{\boldsymbol{\omega}}_{\text{cd}} = \boldsymbol{I}^{-1}(\boldsymbol{q}_j)\left(-\boldsymbol{\omega}_{\text{cd}} \times \boldsymbol{I}(\boldsymbol{q}_j)\boldsymbol{\omega}_{\text{cd}} + \sum_{i=1}^{n_e} \boldsymbol{r}_{E_i} \times \boldsymbol{\lambda}_{E_i}\right) \quad (17a)$$

$$\dot{\boldsymbol{v}}_{\text{cd}} = \boldsymbol{g} + \frac{1}{m}\sum_{i=1}^{n_e} \boldsymbol{\lambda}_{E_i} \quad (17b)$$

with $\boldsymbol{g}$ being the gravitational acceleration, $\boldsymbol{I}(\boldsymbol{q}_j)$ being the moment of inertia of the COM, m being the total mass, and $\boldsymbol{r}_{E_i} = \boldsymbol{R}^T(\boldsymbol{\theta})(\boldsymbol{p}_{E_i} - \boldsymbol{p})$ being the end-effector position relative to the torso's COM.

End-effector constraints. The ground reaction forces $\boldsymbol{\lambda}_{E_i}$ must be physically feasible. First, these forces are subject to the Coulomb friction model given by

$$\lambda_{E_i,n} \geq 0, \quad |\lambda_{E_i,t}| \leq \mu\lambda_{E_i,n} \quad (18a)$$

where $\lambda_{E_i,n}$ and $\lambda_{E_i,t} \in \mathbb{R}^2$ denote the normal and tangential component of $\boldsymbol{\lambda}_{E_i}$ w.r.t. to the terrain plane, and μ is the coefficient of friction. Next, end-effectors can only produce ground reaction forces when they are in contact with the environment. A footfall pattern defined by the motion designer specifies a binary contact flag c, where $c = 1$ indicates a stance phase and $c = 0$ a swing phase. Accordingly, the following constraints ensure that forces vanish in swing phases

$$(1 - c)\boldsymbol{\lambda}_{E_i} = \boldsymbol{0} \quad (19a)$$

Furthermore, no-slip and rolling constraints ensure that the end-effector position, orientation, and wheel speed are consistent with each other

$$\left(\dot{\boldsymbol{p}}_{E_i} + w_{E_i}\boldsymbol{a}(\boldsymbol{\alpha}_{E_i}) \times \boldsymbol{\rho}(\boldsymbol{\alpha}_{E_i})\right)c = \boldsymbol{0} \quad (20a)$$

Given the wheel's orientation $\boldsymbol{\alpha}_{E_i}$, $\boldsymbol{a}$ and $\boldsymbol{\rho}$ compute the wheel axis, and the vector connecting the wheel's center and its contact point with the ground.

Centroidal coordinate frame and hardware limits. In addition to the constraints described above, we instantiate a set of constraints to ensure the centroidal coordinate frame is consistent with the robot's kinematics and a set of auxiliary end-effector variables. We also allow the motion designer to specify constraints that enforce physical hardware limits, such as joint angle limits and boundaries to avoid end-effector collisions.

Parameterization of optimization variables. With this set of constraints in place, we discretize the motion plan $y(t)$ in time using direct transcription and solve the optimization problem (1) using Newton's method and a penalty method approach for the constraints.

Appendix B

Terrain-aware gait and trajectory optimization

In the following, we describe each of the terrain-aware TO's constraints and drop the time dependency of the variables introduced in Section 4.3 to improve the equation's readability.

Initial and final state. The initial and final state are fixed through the equality constraints given by

$$\begin{bmatrix} \boldsymbol{p}^T(0) & \boldsymbol{\theta}^T(0) \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{p}_0^T & \boldsymbol{\theta}_0^T \end{bmatrix}^T \quad (21a)$$

$$\begin{bmatrix} \boldsymbol{p}^T(T_{\text{TO}}) & \boldsymbol{\theta}^T(T_{\text{TO}}) \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{p}_g^T & \boldsymbol{\theta}_g^T \end{bmatrix}^T \quad (21b)$$

where the initial state $\begin{bmatrix} \boldsymbol{p}_0^T & \boldsymbol{\theta}_0^T \end{bmatrix}^T$ and goal state $\begin{bmatrix} \boldsymbol{p}_g^T & \boldsymbol{\theta}_g^T \end{bmatrix}^T$ are provided by the motion designer.

Equations of motion. The system's dynamics is based on a SRBD model of a (wheeled-)legged robot. As described in Section 1.2, SRBD assumes that the limb joints' momentum is negligible compared with the lumped COM inertia, and the inertia of the full-body system stays the same as to some nominal joint configuration. The EOM of the SRBD is given by

$$\dot{\boldsymbol{\theta}} = \boldsymbol{T}(\boldsymbol{\theta})\boldsymbol{\omega} \quad (22a)$$

$$\dot{\boldsymbol{p}} = \boldsymbol{R}_{WB}(\boldsymbol{\theta})\boldsymbol{v} \quad (22b)$$

$$\dot{\boldsymbol{\omega}} = \boldsymbol{I}_{\text{nom}}^{-1}\left(-\boldsymbol{\omega} \times \boldsymbol{I}_{\text{nom}}\boldsymbol{\omega} + \sum_{i=1}^{n_e} \boldsymbol{r}_{E_i} \times \boldsymbol{\lambda}_{E_i}\right) \quad (22c)$$

$$\dot{\boldsymbol{v}} = \boldsymbol{g}(\boldsymbol{\theta}) + \frac{1}{m}\sum_{i=1}^{n_e} \boldsymbol{\lambda}_{E_i} \quad (22d)$$

where $\boldsymbol{R}_{WB}(\boldsymbol{\theta}) \in SO(3)$ represents the rotation matrix that projects the components of a vector from the torso frame B to the world frame W, $\boldsymbol{T}(\boldsymbol{\theta})$ is the transformation matrix from angular velocities in the torso frame B to the Euler angles derivatives in the world frame W, $\boldsymbol{I}_{\text{nom}}$ is the moment of inertia of the COM taken at the robot's nominal configuration $\boldsymbol{q}_{\text{nom}}$, m is the total mass, $\boldsymbol{g}(\boldsymbol{\theta})$ is the gravitational acceleration in torso frame B.

Kinematic limits. A feasible workspace constrains the end-effector's position, that is, $\boldsymbol{r}_{E_i}(t) \in \mathcal{R}_i(\boldsymbol{\theta}, \boldsymbol{p})$, and is approximated by a cube with a fixed edge length, centered at the nominal position of each end-effector relative to the COM.

Legs in contact. During contact phases, the unilateral constraint and the friction cone are enforced given by

$$\boldsymbol{\lambda}_{E_i} \cdot \boldsymbol{n}(\boldsymbol{r}_{E_i}) \geq 0 \tag{23a}$$

$$\boldsymbol{\lambda}_{E_i} \in \mathcal{C}(\boldsymbol{n}, \mu_C) \tag{23b}$$

where $\boldsymbol{n} \in h_{\text{terrain}}(\boldsymbol{r}_{E_i})$ is the local surface normal in world frame W of the height map $h_{\text{terrain}}(x, y)$ evaluated at the contact position $\boldsymbol{r}_{E_i}$. The friction cone constraint (23b) implements an inequality constraint, which limits the ground reaction forces to remain inside the Coulomb friction cone defined by the friction coefficient μ_C. In the implementation of this TO, the constraint is approximated by a friction pyramid.

The motion constraint of traditional legged robots is modeled through the end-effectors' velocities, and when in contact, the velocity $\boldsymbol{v}_{E_i}$ in world frame W of leg i is restricted to

$$\boldsymbol{v}_{E_i} = \boldsymbol{0} \tag{24}$$

In contrast, wheeled-legged robots can execute motions along the rolling direction when in contact. Thus, the motion constraint in (24) changes to

$$\pi_{E_i, \perp}(\boldsymbol{v}_{E_i}) = 0 \tag{25a}$$

$$\boldsymbol{v}_{E_i} \cdot \boldsymbol{n} = 0 \tag{25b}$$

where $\pi_{E_i, \perp}(\cdot)$ in (25a) is the projection of the end-effector velocity $\boldsymbol{v}_{E_i}$ onto the perpendicular direction of the rolling direction. With this formulation and the constraint along the normal direction in (25b), the velocity along the rolling direction is left unconstrained, that is, $\pi_{E_i, \parallel}(\boldsymbol{v}_{E_i}) \in \mathbb{R}$. The

projection $\pi_{E_i, \perp}(\cdot)$, however, cannot be easily computed, due to the missing leg kinematics of the underlying model. In the terrain-aware TO, the rolling direction is approximated through the torso's orientation $\boldsymbol{\theta}$ and the height map's surface normal $\boldsymbol{n} \in h_{\text{terrain}}(\boldsymbol{r}_{E_i})$, that is, $\pi_{E_i, \perp}(\cdot)(\boldsymbol{\theta}, \boldsymbol{n})$. We would like to highlight that the motion constraint in (24) and (25) is the only part that differentiates legged and wheeled-legged locomotion.

Besides, the end-effector is enforced to stay in contact with the terrain, and this equality constraint is formulated by

$$r_{E_i,z}(t) = h_{\text{terrain}}(\boldsymbol{r}_{E_i}) \tag{26}$$

Legs in air. While leg i is in air, the ground reaction forces $\boldsymbol{\lambda}_{E_i}$ are set to zero, and the end-effector is not allowed to touch the terrain. The former defines an equality constraint, and the latter formulates an inequality constraint, that is

$$\boldsymbol{\lambda}_{E_i} = \boldsymbol{0} \tag{27a}$$

$$r_{E_i,z}(t) > h_{\text{terrain}}(\boldsymbol{r}_{E_i}) \tag{27b}$$

Parameterization of optimization variables. A direct collocation method (Hargraves and Paris 1987) transcribes the continuous problem in Section 4.3.1 into an NLP problem optimizing the decision variables in discrete times sampled along the trajectory. Sequences of third- and fourth-order polynomials then obtain the continuous motion, which ensures continuous derivatives at the polynomial junctions. The duration of each predefined phase j, and with that, the duration of each end-effector's polynomial, is changed based on the optimized phase duration $\Delta T_{i,j}$. Since these durations are continuous, the gait timings can be optimized without the need for mixed-integer programming. The optimization problem, however, becomes prone to local minima and, thus, sensitive to the motion designer's initial gait schedule.