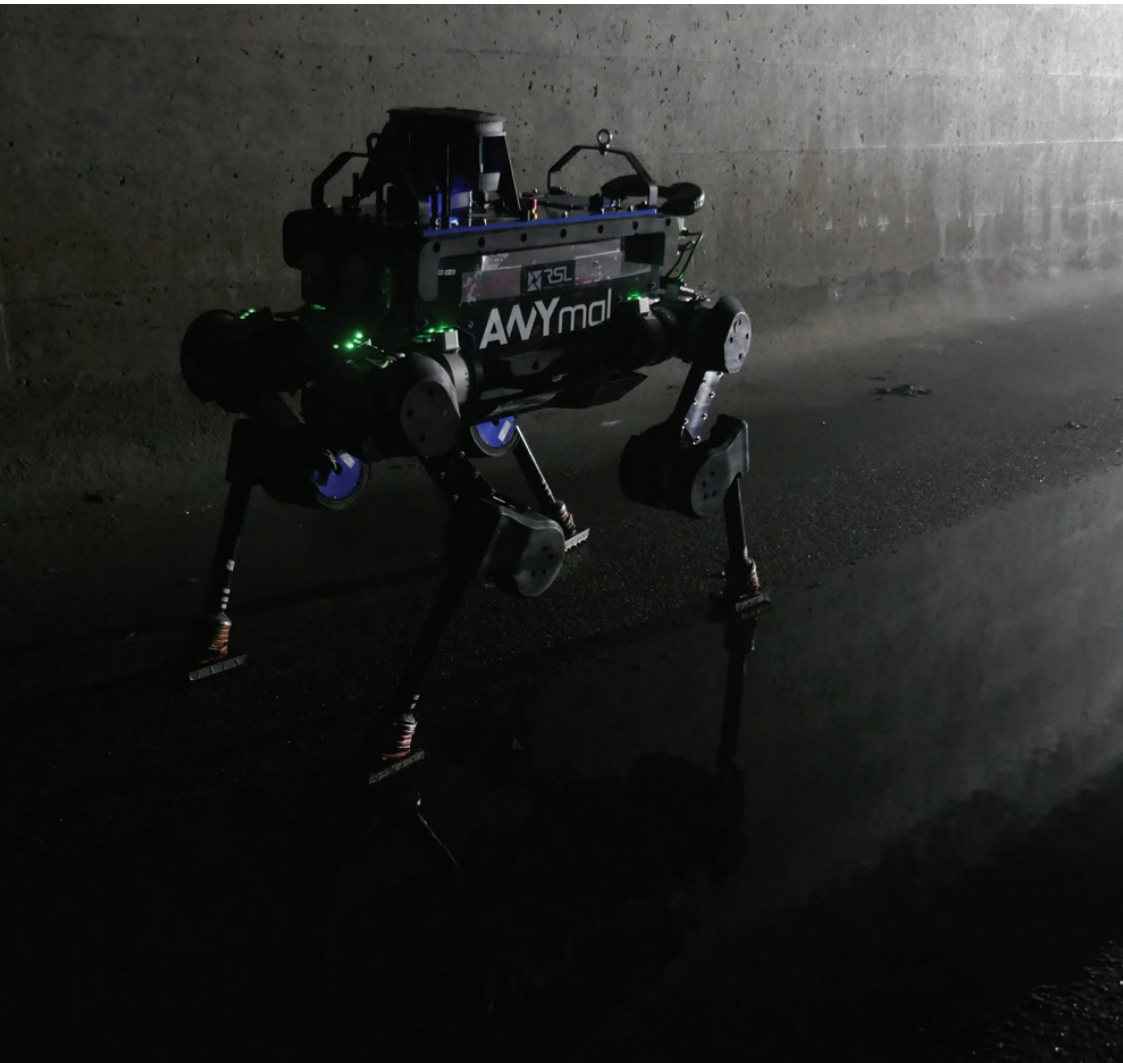
DISS. ETH. NO. 28542

Perceptive and Dynamic Locomotion through Nonlinear Model Predictive Control

Ruben Jelle Grandia



DISS. ETH NO. 28542

Perceptive and Dynamic Locomotion through Nonlinear Model Predictive Control

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES

(Dr. sc. ETH Zurich)

presented by

Ruben Jelle Grandia

M. Sc. in Robotics, Systems, and Control, ETH Zurich

born on 21.10.1992

accepted on the recommendation of

Prof. Dr. Marco Hutter
Prof. Dr. Aaron D. Ames
Prof. Dr. Michael Mistry

2022

Robotic Systems Lab
Institute for Robotics and Intelligent Systems
ETH Zurich
Switzerland

Abstract

Modern robots are slowly but surely finding their way from research laboratories to the real world. Legged robots, in particular, due to their bio-inspired morphology, hold the promise of walking around in complex environments, where they can assist humans in dangerous or repetitive tasks. This thesis focuses on the motion planning and control of complex robotic systems to give them the autonomy and versatility necessary for their broad adoption.

When designing motion optimization and control algorithms for legged systems, a key challenge lies in the fact that the underactuated dynamics of the system and the local terrain simultaneously impose constraints on contact timing, location, and force. Therefore, classical approaches to the locomotion problem have decomposed the control system into modules that separately deal with only footstep planning or with only contact force optimization. Coordination between such modules requires intricate engineering effort, and the exact setup is often specifically tuned for a particular scenario.

In this dissertation, we depart from such manual decomposition and take a holistic view of the locomotion problem to exploit the robot's full capabilities over challenging terrain. Throughout this work, optimal control plays a central role in formulating and solving locomotion as an optimization problem. We explore how reality can be transcribed into optimization-based controllers that remain real-time capable and have the desired closed-loop properties.

In particular, we propose using frequency-dependent cost functions to obtain solutions that are robust to unmodelled dynamics in the high-frequency spectrum. With this method, we demonstrate locomotion on highly compliant terrain and obtain a controller compatible with bandwidth limitations.

The computational challenge of executing the resulting high-dimensional Model Predictive Control (MPC) is addressed by using the feedback policy from a Differential Dynamic Programming (DDP)-based MPC algorithm, resulting in the first MPC-based controller that reasons about all degrees of freedom of the robot simultaneously and is executed on onboard hardware.

Additionally, we incorporate perceptive information into the formulation by extracting a sequence of convex inequality constraints as local approximations of foothold feasibility. We empirically demonstrate that such a strategy is an excellent trade-off between giving freedom to the optimization to discover complex motions and the reliability with which we can solve the formulated problem. We experimentally validate the proposed method in scenarios with gaps, slopes, and stepping stones with the ANYmal quadruped platform, resulting in state-of-the-art dynamic climbing motions across rough terrain.

Besides these contributions that target practical challenges in locomotion, we have studied the unification of MPC with nonlinear control theory to get theoretical guarantees for the closed-loop system and simplify the overall design process. First, we propose the embed Control Lyapunov Function (CLF) stability constraints inside nonlinear MPC to ensure closed-loop stability by design rather than through cost function or parameter tuning. The addition of a prediction horizon provides a performance advantage over CLF based controllers, which operate optimally point-wise in time. Second, safety guarantees can be obtained through the inclusion of Control Barrier Functions (CBFs). We propose a multi-layered locomotion framework that unifies CBFs with MPC to achieve both safe foot placement and dynamic balance along a prediction horizon. Our approach incorporates CBF based safety constraints both in a low-frequency kino-dynamic MPC formulation and a high-frequency inverse dynamics tracking controller. This ensures that safety-critical execution is considered across all hierarchies of the controller.

Keywords: Robotics, Legged Locomotion, Optimal-Control, Model Predictive Control, Terrain Perception, Whole-Body Control.

Zusammenfassung

Moderne Roboter finden langsam aber sicher ihren Weg aus den Forschungslabors in die reale Welt. Insbesondere Roboter mit Beinen versprechen aufgrund ihrer bioinspirierten Morphologie die Möglichkeit, sich in komplexen Umgebungen zu bewegen, wo sie den Menschen bei gefährlichen oder sich wiederholenden Aufgaben unterstützen können. Diese Arbeit befasst sich mit der Bewegungsplanung und Steuerung komplexer Robotersysteme, um ihnen die Autonomie und Vielseitigkeit zu verleihen, die für ihren breiten Einsatz erforderlich sind.

Beim Entwurf von Bewegungsoptimierungs- und Steuerungsalgorithmen für beingebundene Systeme besteht eine der größten Herausforderungen darin, dass die unteraktuierte Dynamik des Systems auf der einen Seite und das Gelände auf der anderen Seite gleichzeitig Einschränkungen in Bezug auf Kontaktzeitpunkt, -ort und -kraft mit sich bringen. Daher haben klassische Ansätze zur Lösung des Fortbewegungsproblems das Kontrollsystem in Module zerlegt, die sich separat nur mit der Schrittplanung oder nur mit der Optimierung der Kontaktkraft befassen. Die Koordination zwischen solchen Modulen erfordert einen hohen technischen Aufwand, und der genaue Aufbau ist oft speziell auf ein bestimmtes Szenario abgestimmt.

In dieser Dissertation wenden wir uns von einer solchen manuellen Zerlegung ab und betrachten das Fortbewegungsproblem aus ganzheitlicher Sicht, um die Fähigkeiten des Roboters in schwierigem Gelände voll auszuschöpfen. In allen Kapiteln dieser Arbeit spielt das Prinzip der optimale Regelung eine zentrale Rolle bei der Formulierung und Lösung des Fortbewegungsproblems als Optimierungsproblem. Wir untersuchen, wie die Realität in optimierungsbasierte Regler übertragen werden kann, die echtzeitfähig bleiben und die gewünschten Eigenschaften des geschlossenen Regelkreises aufweisen.

Insbesondere schlagen wir vor, frequenzabhängige Kostenfunktionen zu verwenden, um Lösungen zu erhalten, die robust gegenüber nicht modellierter Dynamik im Hochfrequenzbereich sind. Mit dieser Methode demonstrieren wir

die Fortbewegung auf stark nachgiebigem Terrain und erhalten einen Regler, der mit den Bandbreitenbeschränkungen der Antriebssysteme kompatibel ist.

Die rechnerische Herausforderung bei der Ausführung des resultierenden hochdimensionalen MPC wird durch die Verwendung des Feedbacks eines DDP-basierten MPC-Algorithmus angegangen, was zum ersten MPC-basierten Controller führt, der alle Freiheitsgrade des Roboters gleichzeitig berücksichtigt und auf Onboard-Hardware ausgeführt wird.

Zusätzlich beziehen wir Wahrnehmungsinformationen in die Formulierung ein, indem wir eine Reihe von konvexen Ungleichheitsbedingungen als lokale Approximationen des Geländes extrahieren. Wir zeigen empirisch, dass eine solche Strategie einen hervorragenden Kompromiss zwischen der Freiheit der Optimierung, komplexe Bewegungen zu entdecken, und der Zuverlässigkeit, mit der wir das formulierte Problem lösen können, darstellt. Wir validieren die vorgeschlagene Methode experimentell in Szenarien mit Lücken, Abhängen und Trittsteinen mit der vierbeinigen Plattform ANYmal, was zu hochmodernen dynamischen Kletterbewegungen in unwegsamem Gelände führt.

Neben diesen Beiträgen, die auf praktische Herausforderungen in der Fortbewegung abzielen, haben wir die Vereinheitlichung von MPC mit nichtlinearer Steuerungstheorie untersucht, um theoretische Garantien für das geschlossene System zu erhalten und den gesamten Entwurfsprozess zu vereinfachen. Zunächst schlagen wir vor, CLF-Stabilitätsbeschränkungen in nichtlineare MPC einzubetten, um die Stabilität des geschlossenen Regelkreises durch die Theorie und nicht durch Kostenfunktionen oder Parameterabstimmung zu gewährleisten. Die Hinzufügung eines Vorhersagehorizonts bietet einen Leistungsvorteil gegenüber CLF-basierten Reglern. Zweitens können Sicherheitsgarantien durch die Einbeziehung von CBFs erreicht werden. Wir schlagen ein mehrschichtiges Fortbewegungskonzept vor, das CBFs mit MPC vereint, um sowohl eine sichere Fußplatzierung als auch ein dynamisches Gleichgewicht entlang eines Vorhersagehorizonts zu erreichen. Unser Ansatz beinhaltet CBF-basierte Sicherheitsbedingungen sowohl in einer niederfrequenten kinodynamischen MPC-Formulierung als auch in einem hochfrequenten inversen Dynamik-Tracking-Controller. Dadurch wird sichergestellt, dass die sicherheitskritische Ausführung in allen Hierarchien des Controllers berücksichtigt wird.

Stichworte: Robotik, Lauffortbewegung, Optimale Reglung, Modellprädiktive Regelung, Geländewahrnehmung, Ganzkörperreglung.

Acknowledgments

Countless people have supported me while working on this doctoral thesis over the past years. First and foremost, I thank my advisor, Prof. Marco Hutter, for his guidance, trust, and continuous encouragement. His open-minded approach has created an inspiring research environment at the Robotic System Lab, where exploring one's own ideas is wholly supported.

I am grateful to have Prof. Aaron D. Ames on my committee. During my research visit to his lab, he made me appreciate control theory and showed me how theoretical work and experimental research can strengthen each other. I thank Prof. Michael Mistry for initiating the THING project and bringing together a great international consortium. I am honored to have him join the committee and review the work.

Furthermore, I would like to thank Diego Pardo, René Ranftl, and Farbod Farshidian for their role as mentors in my scientific journey. The frequent and in-depth discussions with Farbod were of invaluable importance throughout my work. Next, I thank my close scientific collaborators, Marko Bjelonic, Fabian Jenelten, Hendrik Kolvenbach, Maria V. Minniti, Andrew J. Taylor, and Giorgio Valsecchi. Each of you has taught me valuable lessons in both robotics and life.

I want to thank all my colleagues from the Robotic Systems Lab for the great work atmosphere, inspiring discussions, and friendships over the years. I feel privileged to have worked with such a fantastic group of people. A special thank you goes to the lab staff, particularly Maria Trodella, for ensuring that the lab runs smoothly. Many thanks also to all students I had the chance to supervise for exploring novel directions that ultimately shaped and contributed to this work.

Finally, I would like to thank my family, my valued friends worldwide, and especially my partner, Andrea. I could not have done this without your support.

Financial Support

This research was supported by the Swiss National Science Foundation (SNSF) as part of project No.188596 and by the Swiss National Science Foundation through the National Centre of Competence in Research Robotics (NCCR Robotics). This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780883. Additionally, it has been conducted as part of ANYmal Research, a community to advance legged robotics.

Preface

This thesis is a *cumulative* dissertation and, as such, includes a selection of the most relevant publications created by the author during his doctorate. Chapter 1 introduces the research topics, provides an overview of state of the art, and describes this thesis's overarching approach and contributions. Afterward, Chapters 2 to 5 contain the individual scientific articles published in peer-reviewed journals or conference proceedings. Chapter 6 contains recently submitted work, which is under review at the time of writing. Chapter 7 concludes the thesis and provides potential directions for future research.

Contents

Notation

Unless otherwise stated, we use the following convention for mathematical symbols: Scalars are denoted in light (s), vectors in bold ($\mathbf{v}$), and matrices in uppercase bold ($\mathbf{M}$). For functions we use a similar notation refering to the output of the function: Light for scalar functions ($f(\cdot)$), and bold for vector-valued functions ($\mathbf{f}(\cdot)$).

Acronyms

CBF	Control Barrier Function
CLF	Control Lyapunov Function
CoM	Center of Mass
CoP	Center of Pressure
CTLE	Continuous Time Lyapunov Equation
CWC	Contact Wrench Cone
DDP	Differential Dynamic Programming
DRC	DARPA Robotics Challenge
EE	End-Effector
EoM	Equations of Motion
HJB	Hamilton-Jacobi-Bellman
HZD	Hybrid Zero Dynamics
IMU	Inertial Measurement Unit
IP	Interior-Point
KKT	Karush-Kuhn-Tucker

LIPM	Linear Inverted Pendulum Model
LQ	Linear Quadratic
LQR	Linear Quadratic Regulator
MAE	Mean Absolute Error
MIMO	Multiple-Input Multiple-Output
MPC	Model Predictive Control
MSE	Mean Squared Error
NLP	Nonlinear Program
NMPC	Nonlinear Model Predictive Control
ODE	Open Dynamics Engine
p.s.d.	positive semi-definite
QP	Quadratic Program
QP	Quadratic Programming
RL	Reinforcement Learning
SDF	Signed Distance Field
SISO	Single-Input Single-Output
SLIP	Spring Loaded Inverted Pendulum
SLQ	Sequential Linear Quadratic
SLQ-MPC	Sequential Linear Quadratic Model Predictive Control
SQP	Sequential Quadratic Programming
WBC	Whole-Body Controller
ZMP	Zero Moment Point

1

Introduction

In order to effectively deploy robots in the real world, their capabilities have to be adjusted to the environments in which we wish to use them. In the context of the fabrication processes, heavy and rigid robotics arms or wheeled platforms have been successfully integrated into many production lines and distribution halls. However, when looking at applications beyond such controlled environments, e.g., search-and-rescue, outdoor exploration, or inspecting of complex industrial sites, robots are less capable and thus rarely used. Practically speaking, if we strive to automate dangerous or tedious tasks currently performed by humans, we need robots that can go where humans go.

About half a decade ago, the DARPA Robotics Challenge (DRC) sparked interest in legged machines with the vision to develop *"robots capable of executing complex tasks in dangerous, degraded, human-engineered environments"* (DARPA, 2015b; Krotkov et al., 2017). While great technical strides were made, the public perception was dominated by slow-moving robots and robots falling over (DARPA, 2015a). While many factors were at play, part of this result can be attributed to the common strategy of using a hierarchy of highly simplified planning models followed by a one-step-ahead optimization that uses full kinematics and dynamics. (Atkeson et al., 2018). Such a strategy prevents whole-body coordination on the planning level and leads to controllers that try to shoehorn the full dynamics into following the motion dictated by simple models. Additionally, the process of scanning for possible footholds in the terrain, fixing the footstep locations, and then moving the base through the support polygons, limited these systems to slow, quasi-static motions. In contrast, incorporating the full multi-domain system dynamics into gait optimization leads to more natural-looking and more efficient bipedal locomotion, as demonstrated by the humanoid robot DURUS on a treadmill at the sideline of that same DRC competition. There, Hereid et al. (2016) and Reher et

al. (2016) showed how large-scale nonlinear optimization in combination with the Hybrid Zero Dynamics (HZD) framework (Grizzle et al., 2014; Westervelt et al., 2007) can be leveraged to automatically discover stable whole-body locomotion controllers.

Meanwhile, the field of quadrupedal robots saw rapid development on the hardware side by embracing electric motors. This technology change significantly reduced complexity over the hydraulicly driven predecessors (Hutter et al., 2016; Kenneally, De, and Koditschek, 2016; Seok et al., 2013). Today, in the year 2022, quadrupedal robots are widely available and are finding their way into industrial applications (Gehring et al., 2021). ANYmal (ANYbotics, 2022), Spot (Boston Dynamics, 2022), Vision 60 (Ghost Robotics, 2022), and Unitree B1 (Unitree Robotics, 2022) are commercially available platforms and are primarily used for (semi-)autonomous inspection tasks.

Nevertheless, on the control side, the dominant strategy is still to use a slow, static gait in complex environment where visual terrain perception is required. Dynamic running motions are typically performed by a separate, dedicated controller limited to relatively flat terrain where good footholds are abundant. Here, there is a big contrast between the complex dynamic motions we see in nature and those that we are able to replicate with our legged machines. Tightly integrating perception into underactuated dynamic motions and reasoning about closed-loop stability and safety in that context is still an active area of research. To make legged robots useful beyond their current industrial inspection tasks and to allow them to explore any terrain like humans and animals would, there is a need for a control approach that scales to a broader range of scenarios and can unlock the potential of the full dynamics of the system on complex terrain.

A fundamental difficulty in scaling classical locomotion controllers to a broad range of scenarios lies in the chosen decomposition of the problem, as illustrated in Fig. 1.1a. Footstep planning and optimization of the torso motion are solved independently and require hand-crafted heuristics to coordinate the two. These heuristics and the design choices within each module are specialized to either slow and perceptive gaits or fast but blind locomotion. In this thesis, we propose to move away from this classic decomposition and pursue an approach where the full motion of the robot is treated as a single unit, Fig. 1.1b. Moreover, when both the planning and execution layer are reasoning about the complete system, we can start to blur the boundaries also in the hierarchical dimension and move towards an approach where planning and control are unified, Fig. 1.1.

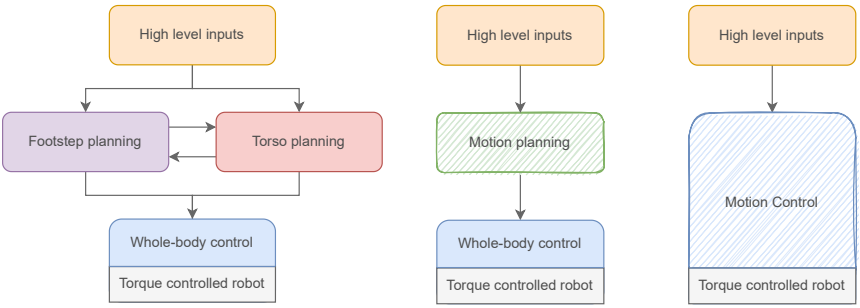


Figure 1.1: Schematic overview of possible approaches to robotic locomotion **(a)** a decomposed motion planning approach, **(b)** a single-task motion planning approach, and **(c)** integrated motion planning and execution.

1.1 Related Work

Successful deployment of a robot requires complex software frameworks governing perception, state estimation, goal extraction, footstep planning, motion planning, and motion stabilization. The complexity of these systems can be appreciated in papers that describe what it takes to deploy a robot in the real world, e.g., for ANYmal (Bellicoso et al., 2018b), Atlas (Kuindersma et al., 2016), or HRP-2 (Nishiwaki, Chestnutt, and Kagami, 2012). This section aims to provide a broad overview of the field of model-based control for legged robots and the many works related to this thesis. A more detailed technical discussion and the relation to individual contributions can be found in the dedicated chapters. The first section reviews the choices one can make to decompose the locomotion problem into subproblems. Afterward, modeling of the system dynamics and modeling of the terrain are discussed. These first three sections define the locomotion problem. In the subsequent section, we review the optimization methods available to solve the posed problems. Finally, we introduce recent work in nonlinear control theory that has been applied to robotics systems. Because many works span several aspects, the same work might be discussed from different viewpoints in the individual sections.

1.1.1 Decomposing Locomotion

The considered decompositions of the locomotion problem are visualized in Fig. 1.2. The fully decomposed, gait and motion decomposition, and no decomposition formulations are each discussed in a separate section. We define a

gait as the sequence of contact configurations and the timing of the transitions between them.

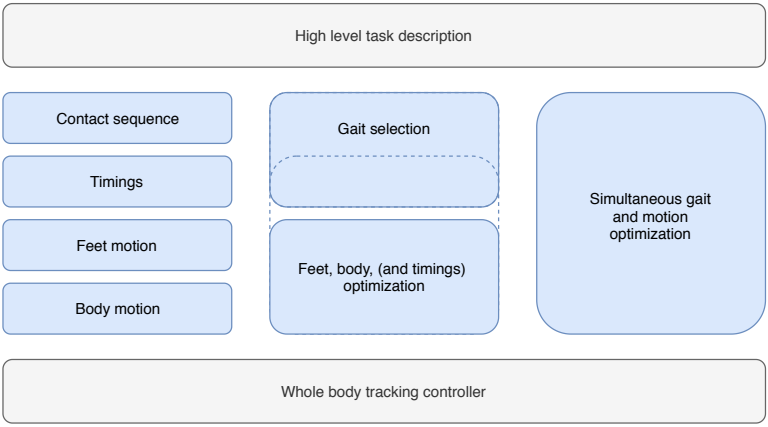


Figure 1.2: Three categories of methods to convert high-level task descriptions into motion plans. Left: Decompose the problem into smaller modules and solve each one sequentially. Middle: Fix the gait before jointly optimizing feet and body motions. Depending on the approach, contact timings can be added to the optimization. Right: Solve the entire locomotion problem in one go.

Full Decomposition

In the fully decomposed approach, one splits the locomotion problem into many different submodules that contain heuristics, closed-form solutions, or small optimization problems. The exact decomposition or the method used within a submodule often depends on the application, robot morphology, and the considered terrain. Since we can draw inspiration from nature on what good locomotion would look like, many heuristics are available from biomechanical work or intuition.

The locomotion problem can be simplified into individual contact transitions when assuming a quasi-static gait with a predetermined stepping sequence. Such methods were demonstrated in the early work on *LittleDog* (Kalakrishnan et al., 2010; Kolter, Rodgers, and Ng, 2008). In a one-step-ahead fashion, these methods check the next foothold for kinematic feasibility, feasibility w.r.t. the terrain, and the existence of a statically stable transition (Tonneau et al., 2018). Belter, Labecki, and Skrzypczynski (2016), Fankhauser et al. (2018), and Griffin et al. (2019) achieved onboard perception and control with such

an approach. Sampling over templated foothold transitions achieves similar motions (Mastalli et al., 2020b). Swing leg motions are often optimized one at a time, given the motion of the base and the target foothold.

In methods for dynamic locomotion, foothold locations are commonly selected through a Raibert heuristic, Linear Inverted Pendulum Model (LIPM), or variants thereof (Gehring et al., 2015; Pratt et al., 2006; Raibert, 1986; Wensing and Orin, 2013). Given a sequence of contact locations, the base motion is designed afterward with simplified stability criteria such as Zero Moment Point (ZMP) (Bellicoso et al., 2018a; Kajita et al., 2003; Vukobratović and Borovac, 2004), a Contact Wrench Cone (CWC) (Caron, Pham, and Nakamura, 2015; Carpentier and Mansard, 2018), or more generally, with an MPC approach that considers contact forces and a single rigid body (Di Carlo et al., 2018).

These methods, initially designed for flat terrain, have been adapted to traverse rough terrain (Bajracharya et al., 2013; Bazeille et al., 2014), or specialized to jump over 2D obstacles (Park, Wensing, and Kim, 2015). When perceptive information is available, the foot placement is often adapted based on a foothold score (Magana et al., 2019; Wermelinger et al., 2016). After that, the same torso optimization module used for blind locomotion can be applied, as demonstrated in the works of Jenelten et al. (2020), Kim et al. (2020), and Villarreal et al. (2020). Recently, Belter et al. (2019), Gangapurwala et al. (2022), and Yu et al. (2021) replaced heuristic foothold selection with a learning-based policy. Still, the hierarchical decomposition of foothold selection and torso motion remains.

Decoupled Gait and Motion Planning

When the contact sequence and timings are given a priori, standard motion optimization can be applied with explicit transitions dynamics at the predefined times. In contrast to the previous section, the torso and leg motions of the robot are now optimized in a single task. One challenge of this line of work is the computational complexity when using high-dimensional models. The methods in his thesis fall in this category, and a key consideration when formulating the optimization problems has thus been to keep computation within real-time constraints.

One of the first demonstrations of simultaneous optimization of foot placement and a ZMP trajectory was achieved by adding 2D foot locations as decision variables to an MPC algorithm (Herdt et al., 2010). Afterward, Kinodynamic (Farshidian et al., 2017a), Centroidal (Orin, Goswami, and Lee,

2013; Sleiman et al., 2021), or even full dynamics models (Budhiraja et al., 2018; Herzog, Schaal, and Righetti, 2016; Koenemann et al., 2015; Pardo et al., 2017) have been used to optimize 3D foot locations and body motion simultaneously.

A few real-time capable realizations have been proposed and experimentally demonstrated on hardware (Bledt, Wensing, and Kim, 2017; Farshidian et al., 2017b; Neunert et al., 2018). To aid the convergence of the optimization, a library of offline motions can be generated to warm-start the online optimization process (Dantec et al., 2021).

As an extension, the duration of each contact phase can be included as a decision variable (Hereid et al., 2016; Ponton et al., 2018) or found through bi-level optimization (Farshidian et al., 2017c; Li and Wensing, 2020; Seyde et al., 2019). This allows the robot to shorten or extend its steps in case of a large disturbance. Furthermore, it can be helpful in rough terrain where different distances between footholds or the presence of obstacles might require a nonstandard swing duration. The phase-based parametrization used in the HZD framework similarly results in online adaptation of stride duration (Grizzle et al., 2014; Westervelt et al., 2007). Interestingly, in this case, the adaptation naturally arises from the nonlinear feedback controller and is not explicitly optimized for during execution. Note, however, that none of these approaches are enable to alter the order of the discrete contact modes.

No Decomposition

Several methods exist that also optimize the contact sequence together with the whole-body motion. In the following paragraphs, we will review such methods based on complementarity constraints, mixed-integer programming, or explicitly integrating contact models into the optimization. While impressive results have been shown in simulation, the optimization landscape of these problems contains many local minima and necessitates case-specific initialization or hyperparameter tuning. Furthermore, demonstrations on hardware are generally lacking.

The fact that contact forces can only be created while in contact with the environment can be translated into complementarity constraints (Dai, Valenzuela, and Tedrake, 2014; Posa, Cantu, and Tedrake, 2014; Sleiman et al., 2019). The constraint encodes that the distance to the environment must be zero before non-zero contact forces are allowed. The method was extended to higher-order integration schemes (Manchester and Kuindersma, 2017; Patel et al., 2019),

and to (partially) elastic problems in (Shield, Johnson, and Patel, 2022). With similar intuition, Mordatch, Todorov, and Popović (2012) augment the search space with indicator variables for contact activation and penalize a mismatch between contact activation, environment distance, and active contact forces.

Using discrete decision variables in a mixed-integer approach naturally encodes the complete hybrid optimization problem (Aceituno-Cabezas et al., 2018; Pajarinen et al., 2017). Marcucci et al. (2017) decompose the hybrid-MPC problem into two stages for increased computational efficiency: A first module proposes a feasible mode sequence given the current state. The second module can then solve a conventional MPC problem with the mode sequence given.

Alternatively, a differentiable contact model can be embedded as part of the system dynamics (Tassa, Erez, and Todorov, 2012). Here, the trajectory optimization does not have to reason about contact activation and forces explicitly, and the physics are enforced during forward simulation. Neunert et al. (2018) use a smooth contact model together with the full rigid body dynamics. However, additional cost function shaping with periodic foot lifting costs was necessary to achieve a satisfying motion. Furthermore, the approach is sensitive to the contact model parameters. A penalty-based contact model was proposed in (Marcucci, Gabiccini, and Artoni, 2017) with similar properties as a soft contact model. Finally, truncated differentiable iterations of a hard contact model are used in (Carius et al., 2018; Carius et al., 2019). The authors showed that also in this case, the optimization landscape contains many local minima. Follow-up work showed that sampling-based methods are more suited for contact discovery than gradient-based methods (Carius et al., 2022). Novel combinations of gradient and sampling-based methods could be a promising direction for these type of problems (Layeghi, Tonneau, and Mistry, 2021).

Finally, Winkler et al. (2018) parameterized the duration of stance and swing phases with independent variables per leg. Different gaits can emerge when contact transitions pass each other along the time horizon.

1.1.2 Robot Dynamics

The dynamics of legged robots can be approximated at different levels of complexity. The approaches found in the literature can be roughly categorized into the following three levels of approximation: inverted pendulum and other template models, kinodynamic and centroidal models, and full rigid body dynamics. These models have already been briefly touched upon in the previous sections since the choice of the model often goes hand in hand with the chosen control decomposition.

Inverted Pendulum Models

The interaction between stance legs and the floating base can be abstracted as an inverted pendulum. With a single contact point and horizontal moving point mass, this model captures the fundamental underactuation of bipedal walking and dynamic gaits for quadrupeds. The simplicity of the model permits closed-form solutions (Englsberger, Ott, and Albu-Schäffer, 2015) and detailed characterization of stability (Xiong and Ames, 2022). When extending to a finite support area, the dynamic feasibility of the motion can be approximated by the ZMP (Vukobratović and Borovac, 2004), which needs to lie inside the support polygon. This method does not reason about contact forces explicitly but ensures that there exists a contact force distribution that satisfies the unilateral constraints.

The Spring Loaded Inverted Pendulum (SLIP) model has been widely studied as well and shown to better capture the periodic vertical torso motion during dynamic human walking and running (Blickhan, 1989; Geyer, Seyfarth, and Blickhan, 2006). This correspondence sparked a trend of designing compliant robots to specifically fit this model (Grimes and Hurst, 2012; Hubicki et al., 2016a; Hutter et al., 2012; Rezazadeh et al., 2015).

Other extensions have included additional terms that account for the absence of angular momentum, step height variation, etc. Still, the reduced number of state variables specifically chosen to fit periodic walking makes reasoning over general aperiodic whole-body motions challenging with this class of models.

Kinodynamic and Centroidal Models

A kinodynamic model considers the full kinematics of the robot and a reduced representation of the dynamics. This can, for example, be achieved by considering mass-less legs together with a free-floating base with constant inertia (Farshidian et al., 2017a). We used this formulation in Chapters 2, 3, and 5 and recently relaxed the fixed torso inertia assumption in Chapter 6. Instead of parameterizing the legs through the joint coordinates, some works choose to parameterize only the Cartesian foot position and constrain leg length (Bledt, Wensing, and Kim, 2017; Winkler et al., 2018). An extension was proposed in (Arreguit, Faraji, and Ijspeert, 2018), where an additional rigid body is used to represent each limb.

These kinodynamic models should not be confused with the centroidal dynamics (Orin, Goswami, and Lee, 2013), which compresses the full rigid body dynamics into six momentum equations subject to external forces. The catch is

that these six equations by themselves cannot be forward integrated to obtain a meaningful body pose. Wieber (2006) demonstrated that any body orientation can be achieved with an articulated system while keeping the angular moment zero along the trajectory (e.g., a cat reorienting itself during a fall). Still, in a collocation approach, this model can be effectively used to reduce the number of equality constraints that represent the dynamics (Dai, Valenzuela, and Tedrake, 2014). When joint velocities are added as control variables, the model can be forward integrated under the assumption of sufficient torque and instantaneous velocity control (Sleiman et al., 2021).

Full Rigid Body Dynamics

The full rigid body dynamics describe the dynamics of a set of rigid bodies in 3D space connected through (actuated) joints and under the influence of external forces. When the kinematic structure of the robot forms a directed graph, i.e., there are no kinematic loops, efficient recursive algorithms exist (Carpentier et al., 2019; Featherstone, 2014). This model has been extensively used for instantaneous torque control in the Whole-Body Controller (WBC) framework (Mistry, Buchli, and Schaal, 2010; Nakanishi, Mistry, and Schaal, 2007; Sentis and Khatib, 2006), where it allows direct incorporation of friction cone constraints and torque limits. A projected formulation eliminates contact forces and allows planning in the constraint consistent subspace (Pardo et al., 2017). This type of model was long thought to be too complex for online motion optimization for full-size humanoids or quadrupeds. However, recent work shows promising results (Mastalli et al., 2020a, 2022), and deployment on onboard hardware does not seem to be too far away anymore.

Contact Dynamics

The vast majority of approaches consider the environment to be rigid together with a Coulomb friction model and unilateral constraints. These constraints are typically modeled with a friction cone or a polytopic approximation when linear constraints are preferred. By constraining the solution space to remain below the friction limit, one prevents the need to describe sliding contacts, which are significantly harder to model and optimize over (Carius et al., 2019).

In the context of contact invariant optimization, as discussed in Section 1.1.1, soft contact models (Neunert et al., 2018) or contact smoothing (Mordatch, Todorov, and Popović, 2012) are used inside trajectory optimization. However, these models are selected and tuned for their numerical properties rather than physical accuracy. The models need to be smooth since the highly coupled

interaction between stiff contact dynamics and slow dynamics of the robot leads to poor convergence of the algorithms. In the worst case, this numeric model tuning can lead to highly undesired effects when the optimizer starts exploiting dynamic properties of the terrain that are entirely wrong.

Some work has been done to try to learn a contact model from data, especially for granular media (Hubicki et al., 2016b). A combination of learning a terrain model and trajectory optimization has been proposed by Chang et al. (2017). We have explored this direction in the beginning of this thesis as an alternative to the method proposed in Chapter 2. However, similar to what was discussed in the previous paragraph, we often found that the optimizer starts exploiting artifacts of the contact model. More work is needed to ensure that these learned contact models are well behaved across the entire optimization domain.

Actuator Dynamics

Most model-based approaches for torque-controlled robots consider the actuators a perfect torque source. However, in the case of series elastic actuators, or when operating the motors close to their limits, the low-level dynamics can significantly affect the system dynamics. Moreover, results in successful sim-to-real transfer in Reinforcement Learning (RL) methods underline the importance of considering imperfect actuator dynamics (Hwangbo et al., 2019). Several proposals have been made to incorporate bandwidth, torque, and joint limits within model-based control (Braun et al., 2013; Nakanishi and Vijayakumar, 2012; Schlossman et al., 2018). Unfortunately, since parts of the underlying actuator dynamics have very different time constants than the robot itself, much smaller time steps are required to simulate the coupled dynamics accurately. This leads to slower update rates, preventing such models from being used in MPC for high-dimensional systems. Additionally, numerical instability can arise when a limb with stiffly modeled actuators impacts the environment (Koenemann et al., 2015). In Chapter 2, we avoid such issues by promoting the optimal solution to lie within the space for which the perfect torque-source assumption is still valid, instead of explicitly modeling the actuators.

1.1.3 Terrain Representation

The geometry of the terrain affects both foothold selection and collision avoidance during the swing phase. For surprisingly many scenarios, irregularities in the terrain can be accepted as disturbances and adapted to by an appropriate feedback controller (Dario Bellicoso et al., 2016; Di Carlo et al., 2018;

Lee et al., 2020). However, for more complex terrain, especially those with negative obstacles, this strategy is not sufficient, and visual information needs to be incorporated.

The use of a 2.5D elevation map as a local, high-resolution representation of the terrain has a long-standing history in the field of legged robotics (Herbert et al., 1989). Till today, this representation, together with an elevation mapping framework (Fankhauser, Bloesch, and Hutter, 2018), is the central intermediate representation of visual information for perceptive locomotion controllers. Approaches where footholds are selected based on a search or a sampling-based algorithm can directly operate on such a structure. When incorporating this information into gradient-based optimization, however, more work is needed to ensure good convergence in the face of discontinuities and non-linearities of the terrain. This aspect will be discussed in detail in Chapter 6.

In the long term, the fact that only a single surface can be represented per location in an elevation map leads to practical limitations when navigating confined spaces or walking underneath obstacles. As a first step, Buchanan et al. (2021) propose to use an elevation layer for both the floor and the ceiling. Gaertner et al. (2021) propose to use a Euclidean Signed Distance Field for torso collision avoidance. However, the practical resolution of these methods is only suitable for collision avoidance and not for precise foot placement. Alternatively, Bertrand et al. (2020) use an octree representation to fuse 3D pointcloud data over time, and extract a convex plane decomposition as representation for control. It remains an open question what the right representation is for legged locomotion to fuse visual information over time.

1.1.4 Motion Optimization

For trajectory optimization, large-scale optimization software like SNOPT (Gill, Murray, and Saunders, 2005) and IPOPT (Wächter and Biegler, 2006) are popular and form the basis of many works in offline trajectory optimization (Mordatch, Todorov, and Popović, 2012; Nguyen et al., 2016; Posa, Cantu, and Tedrake, 2014; Winkler et al., 2018). These mature software packages contain years of development and are therefore extremely robust. However, they are general-purpose, and despite the use of sparse linear algebra routines, optimization times remain in the order of seconds at best for realistic locomotion problems.

In the field of nonlinear MPC, specialized solvers were developed that explicitly exploit the structure of optimal control problems to enable iterations at a

much higher rate. See (Kouzoupis et al., 2018) for a comparison of state-of-the-art Quadratic Program (QP) solvers that form the core of second-order optimization approaches to nonlinear problems. In the context of robotic motion optimization, several variants of Differential Dynamic Programming (DDP) (Jacobson and Mayne, 1970), e.g., iLQR (Howell, Jackson, and Manchester, 2019; Tassa, Erez, and Todorov, 2012), SLQ (Farshidian et al., 2017a), and FDDP (Mastalli et al., 2020a). These approaches also exploit sparsity through the recursive Riccati backward pass. Throughout this thesis, we have used and contributed to these structure-exploiting algorithms to realize real-time capable implementations.

1.1.5 Nonlinear Control

The field of nonlinear control theory studies the closed-loop control of systems governed by nonlinear differential equations. Rigorous mathematical tools establish properties of the controlled system without resorting to linearization or numerical discretizations (Khalil, 2002). From this class of methods, Lyapunov-based analysis is a powerful tool for certifying stability properties. The existence of a CLF for the controlled system implies the existence of a continuous state-feedback controller that renders the origin globally asymptotically stable (Artstein, 1983; Sontag, 1989b). In extension, the CLF definition implies the existence of a point-wise set of stabilizing control inputs.

Due to this, the CLF itself may then be used to synthesize an optimization-based controller with more desirable properties using quadratic programming (Ames et al., 2014; Ames and Powell, 2013). The use of CLFs to synthesize stabilizing controllers for robotic platforms has become increasingly popular (Galloway et al., 2015; Ma et al., 2017; Nguyen and Sreenath, 2015). Hybrid Zero Dynamics (HZD) is a framework to synthesize the necessary Lyapunov function around virtual constraints and ensure hybrid invariance for the periodic motion, including discrete jumps (Westervelt et al., 2007).

Geometric constraints like collision avoidance or stepping on a predefined surface can be formulated as safety constraints on the configuration of the robot. Control barrier functions are a formal framework to design such a constraint with closed-loop guarantees for the full nonlinear system. This approach was used to achieve walking on stepping stones (Nguyen et al., 2016) and generalize collision avoidance (Singletary et al., 2021). Moreover, they can be embedded into optimization problems in the same way as CLFs. In Chapters 4 and 5, we embed these safety and stability constraints inside an MPC formulation to achieve long-term optimality with closed-loop guarantees.

1.2 Approach and Contributions

The goal of this research can be summarized as follows:

Automatically adapt locomotion behavior to the environment to increase the autonomy and robustness of legged robots.

We first close the gaps between the current state-of-the-art in dynamic motions and the conservative approaches used to traverse rough terrain. This is achieved by jointly considering all degrees of freedom of the robot instead of decoupling base and foot motion in a hierarchical approach. Second, we exploit the unified capabilities by demonstrating dynamic locomotion in challenging terrain where the robot is pushed to its limits, and nontrivial motions are required. By doing so, we increase the range of environments where the robot can autonomously locomote, thus extending the applicability of legged robots.

One difficulty in designing controllers for such complex systems is the need to meet a large set of design requirements simultaneously. Achieving stable and safe behavior is often in conflict with performance objectives, and finding the right balance between these requirements can be challenging. The central idea of this thesis is to coordinate these tasks not through laborious engineering but instead to synthesize a solution through an online optimization process. The focus is on online algorithms such that the robot can react to changes in the environment, disturbances, and use the latest information coming from its sensors.

Nonlinear optimal control provides a rich framework to formulate the envisioned tasks. Control objectives are set through the cost function together with a specification of the system model and other imposed constraints, i.e.,

$$\begin{aligned} \text{minimize} \quad & \text{Control objectives}, && \text{(1.1a)} \\ \text{subject to:} \quad & \text{System dynamics}, && \text{(1.1b)} \\ & \text{Stability constraints}, && \text{(1.1c)} \\ & \text{Safety constraints}, && \text{(1.1d)} \\ & \dots && \text{(1.1e)} \end{aligned}$$

In this way, the burden of trading off multiple objectives under the constraints set by physics is transferred from the engineer to the optimization algorithm. Continuously solving such an open-loop optimal control problem in a receding horizon fashion results in feedback control. This process is often referred to

as MPC (Mayne et al., 2000; Rawlings, Mayne, and Diehl, 2017). The resulting requirement that each optimal control problem must be solved within a reasonable time compared to the dynamics of the problem raises interesting problems and underlies many discussions in this thesis.

Given the abstract optimal control problem in (1.1), several research questions naturally arise:

Q1 (formulation): *How can we transcribe the locomotion problem and the robot's environment into a tractable optimization problem?*

Q2 (execution): *How do we execute the optimized solution? Specifically, how do we bridge the frequency difference between motion optimization and execution?*

Q3 (guarantees): *What guarantees can we give on the resulting closed-loop feedback control?*

Each publication within this thesis speaks to one or more of these fundamental questions. The following paragraphs state for each paper the scientific contributions made in this context and summarize the lessons learned.

Paper I

Grandia, R., Farshidian, F., Dosovitskiy, A., Ranftl, R., and Hutter, M. (2019a). "Frequency-aware model predictive control". *IEEE Robotics and Automation Letters* 4.2, pp. 1517–1524

The presence of unmodeled dynamics can make an optimized motion infeasible for the real system. In particular, compliant contacts and actuator dynamics lead to bandwidth limitations and are the source of many failed sim-to-real transfers. In classical locomotion controllers, these problems are often mitigated by adding ad-hoc low-pass filters or rate limiters between individual control modules. In the context of Q1 (formulation), we investigate how these bandwidth limitations can be encoded into the optimization problem in a more principled way. We introduce frequency-dependent cost functions, which allow us to impose additional costs on solutions that excite these unmodelled dynamics. With this method, we improved the robustness of ANYmal while locomoting on unmodeled soft ground. This technique proved to be fundamental for successful sim-to-real deployment, and we have used this technique in every locomotion paper following this work.

Paper II

Grandia, R., Farshidian, F., Ranftl, R., and Hutter, M. (2019b). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4730–4737

In this work, we focussed on Q2 (execution) and proposed a method to bridge the gap between a low update-rate MPC and the high rate execution of torque commands. So far, related work has introduced additional feedback gains within a whole-body control framework to track the feedforward component of the MPC solution. Instead of these manually tuned gains, we proposed to use the feedback policy derived from a DDP-based algorithm at the rate of the tracking controller to provide a fast, linear approximation to the optimal policy. Furthermore, we show that our feedback MPC algorithm directly designs constraint-satisfactory gains without additional computational cost. To the best of our knowledge, this was the first time that an MPC algorithm that considers all degrees of freedom of the robot was deployed using only onboard computation.

Second, the Sequential Linear Quadratic (SLQ) algorithm was extended to include inequality constraints through a barrier function method, which allows us to formulate friction cone constraints. This contributed to a more complete description of the locomotion problem in view of Q1 (formulation).

Paper III

Grandia, R., Taylor, A. J., Singletary, A., Hutter, M., and Ames, A. D. (2020). "Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions". In: *Proceedings of Robotics: Science and Systems*. Corvalis, Oregon, USA

In discrete-time nonlinear MPC, stability is guaranteed by an appropriately designed terminal cost and terminal constraint (Grüne and Pannek, 2017; H. Chen and F. Allgöwer, 1998; Mayne et al., 2000). However, these terminal components are challenging to design in general and can conflict with performance objectives. This work provides an alternative answer to Q3 (guarantees) and presents a novel set of approaches for unifying CLFs and nonlinear MPC on robotic platforms. The unified methods all guarantee stability by design, whereas the baseline nonlinear MPC methods were sensitive to cost function parameters. We thus find that the pairing of these control methodologies significantly reduced the tuning of prediction horizon length and terminal con-

ditions. Moreover, compared to baseline CLF methods, the addition of the prediction horizon improves the performance of the system.

Limited computational resources and fast system dynamics challenge the deployment of these unified methods on modern robotic systems. While existing work has analyzed the stability and optimality properties obtained through the unification of CLFs and nonlinear MPC, there has been little attention to the practical and computational aspects of the resulting nonlinear optimization problem. Indeed, to the best of our knowledge, such a control scheme was not yet experimentally realized on robotic hardware before this work.

Paper IV

Grandia, R., Taylor, A. J., Ames, A. D., and Hutter, M. (2021). "Multi-layered safety for legged robots via control barrier functions and model predictive control". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8352–8358

In continuation of the work in Paper IV towards Q3 (guarantees), we similarly addressed safety constraints by combining methods from nonlinear control theory and MPC. Second, we study how such guarantees can be provided throughout the hierarchy of motion planning and execution, thus addressing Q2 (execution) in the context of formal guarantees. We build upon the controller proposed in Paper II and add CBF safety constraints both in the low-frequency MPC layer as well as the higher rate WBC tracking controller. Compared to standard CBF approaches, this adds a horizon when determining safe control inputs. Compared to MPC approaches, the safety-critical constraint is enforced at a higher rate and incorporates the full rigid body dynamics model.

Paper V

Grandia, R., Jenelten, F., Yang, S., Farshidian, F., and Hutter, M. (2022). "Perceptive Locomotion through Nonlinear Model Predictive Control". *(submitted to) IEEE Transactions on Robotics*

In the last paper of this dissertation, we turn our focus back to Q1 (formulation) and include perceptive terrain information in the optimal control problem. To keep the optimization problem tractable, we propose to use a sequence of geometric primitives, in this case, convex polygons, to encode local foothold constraints. Compared to existing approaches that include the full map in the

optimization, this approach mitigates the numerical challenges posed by the nonlinearity and discontinuity of the terrain. Additionally, we add collision avoidance constraints between the knees and the terrain based on a Signed Distance Field (SDF). Because the approach jointly optimizes all degrees of freedom of the robot, we find that the robot coordinates both the main body and the leg to satisfy this collision avoidance constraint in complex terrain. Such emerging coordination was hard to achieve in the classical decomposed locomotion controllers.

Besides the contributions to advancing state-of-the-art in perceptive locomotion, this work resulted in several software contributions. We implemented a multiple-shooting algorithm to provide fast and reliable online solutions to the nonlinear optimal control problem. This algorithm proved to be significantly more robust than the SLQ method used in the preceding work. The implementation is publicly available[1] as part of the MPC toolbox OCS2 (*OCS2: An open source library for Optimal Control of Switched Systems*). The segmentation of the terrain into steppable planes and convex local approximations is publicly available as well[2]. Finally, the implemented algorithm to convert an elevation map into an SDF was contributed to the open-source gridmap library[3].

1.3 Robotic Systems

The methods developed during this thesis are formulated and implemented in a general way, such that systems in related domains can benefit. The collage in Fig. 1.3 gives an overview of all the robots that the methods described in this thesis have been applied to, either directly as part of this thesis or during a collaboration.

The quadrupedal robot ANYmal B was used in Chapters 2 and 3, and its successor, ANYmal C, was used in Chapters 5 and 6. The cat-like robot Dyana was developed during the 2021 ETH Focus project (Dyana, 2022). In that project, the MPC formulation was modified to incorporate the closed-chain kinematics that couple knee and ankle joints in the hind legs. For the wheeled version of ANYmal C, the MPC was adapted by simply removing the contact constraint in the driving direction of the end-effector (Bjelonic et al., 2021). Other than that, the basic formulation and even cost function

[1] https://github.com/leggedrobotics/ocs2/tree/main/ocs2_sqp
[2] https://github.com/leggedrobotics/elevation_mapping_cupy
[3] https://github.com/ANYbotics/grid_map

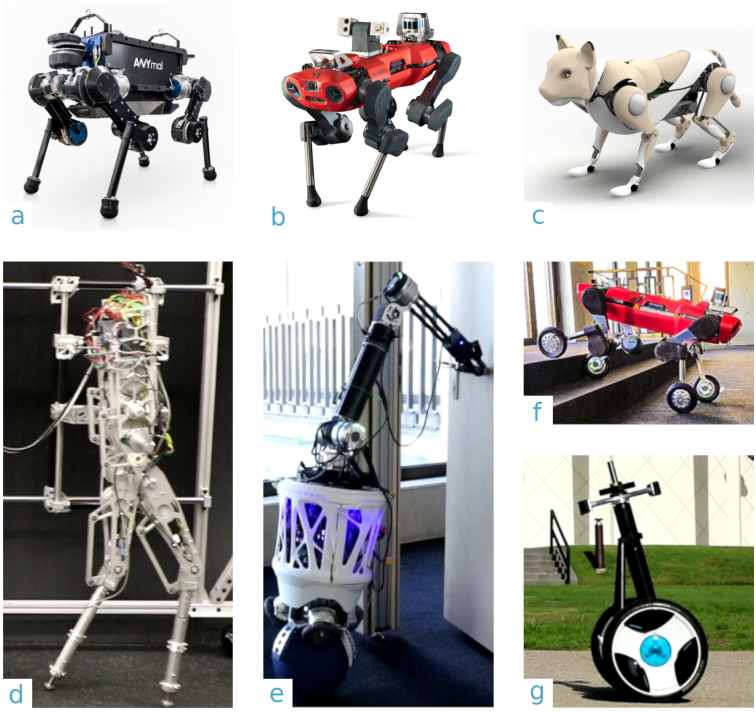


Figure 1.3: Robots that the methods described in this thesis have been applied to: a) ANYmal B, b) ANYmal C, c) Dyana, d) AMBER-3M, e) Ballbot, f) ANYmal on wheels, g) Ninebot E+ Segway.

parameters for the point-foot robot could be reused, proving the generality of the approach.

Besides quadrupedal robots, applications to different morphologies have been studied as well. In Galliker et al. (2022), we deployed our multiple-shooting MPC described in Chapter 6 on the bipedal robot AMBER (Ambrose et al., 2017). The ball balancing robot with an arm was used during two collaborations (Minniti et al., 2019, 2021). For this robot, the underactuation of the base and the contact interaction of the arm share many similarities with legged robots. Finally, the Segway is the only robot that does not change its contact configuration during deployment. The simpler, single-domain dynamics made it an excellent test case for the more theory focussed work in Chapter 4.

2

Frequency-Aware Model Predictive Control

Grandia, R., Farshidian, F., Dosovitskiy, A., Ranftl, R., and Hutter, M. (2019a). "Frequency-aware model predictive control". *IEEE Robotics and Automation Letters* 4.2, pp. 1517–1524

DOI: 10.1109/LRA.2018.2806566

Video: https://youtu.be/RSJgqkk2VRI

Transferring solutions found by trajectory optimization to robotic hardware remains a challenging task. When the optimization fully exploits the provided model to perform dynamic tasks, the presence of unmodeled dynamics renders the motion infeasible on the real system. Model errors can be a result of model simplifications, but also naturally arise when deploying the robot in unstructured and nondeterministic environments. Predominantly, compliant contacts and actuator dynamics lead to bandwidth limitations. While classical control methods provide tools to synthesize controllers that are robust to a class of model errors, such a notion is missing in modern trajectory optimization, which is solved in the time domain. We propose frequency-shaped cost functions to achieve robust solutions in the context of optimal control for legged robots. Through simulation and hardware experiments we show that motion plans can be made compatible with bandwidth limits set by actuators and contact dynamics. The smoothness of the Model Predictive Control (MPC) solutions can be continuously tuned without compromising the feasibility of the problem. Experiments with the quadrupedal robot ANYmal, which is driven by highly-compliant series elastic actuators, showed significantly improved track-

ing performance of the planned motion, torque, and force trajectories and enabled the machine to walk robustly on terrain with unmodeled compliance.

2.1 Introduction

Trajectory optimization based on the full dynamics of a robotic system provides a flexible tool to generate complex motion plans. It enables the system to exploit the dynamic capabilities of the robot to achieve a task. State-of-the-art approaches are able to rapidly find solutions while incorporating increasingly complex model descriptions, which allows using trajectory optimization in a MPC fashion. However, relying on the specific structure of the model makes implementation of the synthesized motion plans prone to modeling errors. Executing motion plans on hardware has therefore proven to be nontrivial and often requires manual, task-dependent tuning of cost functions and constraints to achieve feasible motions.

A major source of modeling error is the treatment of actuators as perfect torque sources. Any real system is subject to bandwidth limits and as such is not an ideal torque source. A similar modeling error occurs when assuming a rigid contact with the ground. The rigid contact essentially provides the optimizer with infinite bandwidth control over the contact forces. This assumption generally does not hold during locomotion in outdoor environments or on compliant surfaces as shown in Fig. 2.1. As a result, motion plans generated assuming idealized contact and actuator dynamics cannot be tracked by the hardware, leading to poor tracking performance or failure of the locomotion controller.

In this paper, we extend MPC methods for legged locomotion to situations where the assumptions of rigid ground and perfect actuators are invalid. The selected baseline model describes the 6 degrees of freedom Center of Mass (CoM) dynamics and motion of each leg. Simultaneous optimization of footstep location and contact interaction is achieved by having both contact forces and joint velocities as control inputs. We address the issues of inherent bandwidth limits in real robots by adapting the cost function to be frequency-dependent, making it possible to penalize high frequencies in the motion plans. The solver, therefore, does not have to reason about the exact details of terrain and actuator dynamics but will produce solutions that are achievable under the bandwidth limits. We show that motion plans generated with our frequency-aware trajectory optimization can be followed by the hardware more

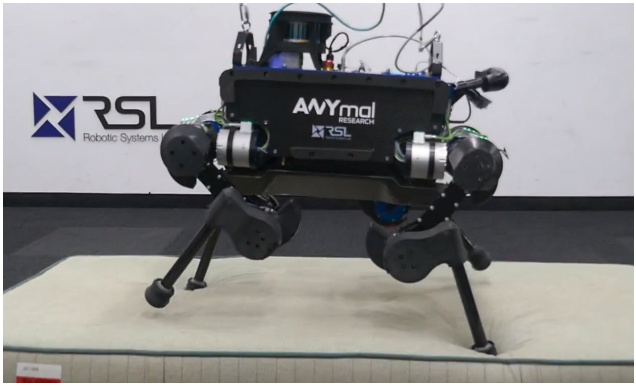


Figure 2.1: The quadruped robot ANYmal (Hutter et al., 2016) trotting in place on non-rigid terrain. This experimental setup is used to test the controller's robustness against unmodeled contact dynamics.

accurately than those generated with a standard baseline and enables locomotion on compliant terrain.

2.1.1 Related Work

Local feedback stabilization around a planned motion is a well-known technique to mitigate modeling errors (Englsberger and Ott, 2012; Sygulla et al., 2017; Takenaka et al., 2009; Zhou et al., 2016) and has led to successful soft ground walking for a bipedal robot (Hashimoto et al., 2012). However, especially for dynamic motions, performance can be increased by providing a high-quality feedforward term. The effort in this work to improve the feasibility of the feedforward term and state reference can be seen as complementary to local stabilization strategies. Moreover, disturbances can be rejected by adjusting the motion plan through fast replanning, reducing the burden on the feedback controller.

In the case of a series elastic actuator, the dynamics can be approximated and added to the model. The optimization algorithm is in those cases able to exploit the properties of the specific actuator and adding spring-damper elements to the joints is even known to result in motions that resemble those found in nature (Schultz and Mombaur, 2010). For series elastic actuators, methods have been proposed to incorporate bandwidth, torque, and joint limits in a computationally efficient way (Braun et al., 2013; Nakanishi and Vijayakumar, 2012; Schlossman et al., 2018). However, very often we do not have

exact details of actuators and modeling them would lead to high engineering effort. Moreover, since parts of the underlying actuator dynamics have very different time constants, smaller timesteps are required. This leads to slower update rates, preventing such models from use in MPC with complex systems. Additionally, stability problems can arise when a limb with stiffly modeled actuators makes contact with the environment (Koenemann et al., 2015). We avoid such issues by not explicitly modeling the actuators, but by incorporating well-known bandwidth limitations up to which the perfect tracking assumption is valid.

While model parameters for actuators can be obtained from first principles or through repeated experiments, contact dynamics are considerably harder to model or predict. A combination of learning a terrain model and trajectory optimization has been proposed in (Chang et al., 2017). However, such methods have not yet reached real-time capabilities.

In the context of contact invariant optimization, soft contact models (Neunert et al., 2017) or contact smoothing (Mordatch, Todorov, and Popović, 2012) are used inside trajectory optimization. These models are selected and tuned for their numerical properties rather than physical accuracy. The models need to be smooth since the highly coupled interaction between stiff contact dynamics and slow dynamics of the robot lead to poor convergence of the algorithms. In the worst case, this numeric model tuning can lead to highly undesired effects when the optimizer starts exploiting dynamic properties of the terrain which are entirely wrong.

Reasoning over higher order terms results in solutions with a higher degree of continuity, which improves performance on hardware considerably. This can be achieved by selecting a smooth parameterization of the solution space as done in spline based optimization (Bellicoso et al., 2018a; Kalakrishnan et al., 2010; Werner, Turlej, and Ott, 2017), collocation methods (Hereid et al., 2016; Pardo et al., 2017), or when using dynamic motion primitives (Werner et al., 2017). This, however, limits the motions that can be expressed and can often require a problem-specific, manual tuning procedure. Alternatively, higher derivatives can be selected as the control inputs in MPC (Kajita et al., 2003). In Sect. 2.2.4 we show that this formulation can be interpreted as a special case of the presented frequency weighting method.

As an alternative to higher order formulations or explicitly modeling actuator dynamics, we propose to encode bandwidth limits through the cost function. A trivial way to do so is to put extra costs on input signals, but this results

in slower response overall and goes against the desire to perform highly dynamic motions at the limits of the system. Instead, we intend to explore a different approach and propose to use a frequency-dependent cost function (Gupta, 1980). We penalize control actions only in the high frequency range and combine this idea with a modern optimal control framework (Farshidian et al., 2017a; Farshidian et al., 2017b).

Frequency-based approaches have been proposed with several applications in both the MPC and legged robotic communities. In (Hours et al., 2015) constraints on the output spectrum are formed in the frequency domain. However, the proposed window-based constraints approach requires previous and future decision variables, which negatively affects the Riccati-sparsity pattern exploited in optimal control methods. In (Hashimoto et al., 2015), a Fourier transform has been used to find a closed form solution for momentum compensation of the lower body with the upper body. In contrast to our setting where the unmodeled dynamics are unknown, the forces that are to be compensated for are assumed to be known based on a full rigid body dynamics model.

2.1.2 Contributions

We introduce frequency-dependent cost functions integrated into modern MPC strategies for legged locomotion. Through simulation experiments, we study the effect of such a cost function on the resulting solutions. The proposed method provides the user with an intuitive way to achieve robustness against unmodeled phenomena like actuator bandwidth limits and non-rigid contact dynamics. These findings were successfully validated in hardware experiments on different grounds. Using frequency-shaped cost functions, we could improve the robustness of ANYmal while locomoting under substantial external disturbances coming from external pushes or unmodeled soft ground.

2.2 Method

First, we discuss uncertainty in the dynamics from a robustness point of view and motivate the particular choice of cost functions. Afterward, the integration with a Sequential Linear Quadratic Model Predictive Control (SLQ-MPC) method (Farshidian et al., 2017b) is presented. This method, based on Differential Dynamic Programming, relies on a linear approximation of the dynamics and a quadratic approximation of the cost function around the latest trajectory.

For brevity of notation in the current section, and without loss of generality, we consider the following quadratic cost function without linear and mixed state-input costs,

$$J = \frac{1}{2}\int_0^\infty \left(\mathbf{x}(t)^\top \mathbf{Q}\mathbf{x}(t) + \mathbf{u}(t)^\top \mathbf{R}\mathbf{u}(t)\right)\, \mathrm{d}t, \tag{2.1}$$

where $\mathbf{Q}$ is the positive semi-definite state cost Hessian and $\mathbf{R}$ is the positive definite input cost Hessian.

2.2.1 High Frequency Robustness

Consider a linear plant $G(j\omega)$ with unstructured multiplicative uncertainty model $L(j\omega)$,

$$\tilde{G}(j\omega) = [I + L(j\omega)]G(j\omega), \tag{2.2}$$

$$\bar{\sigma}[L(j\omega)] < l_m(\omega), \quad \forall\omega \geq 0, \tag{2.3}$$

where $\bar{\sigma}$ is the maximum singular value of the disturbance model and $l_m(\omega)$ is a frequency-dependent upper bound. The closed loop stability condition for these models is (Doyle and Stein, 1981),

$$l_m(\omega) < \underline{\sigma}[I + GK(j\omega)^{-1}], \tag{2.4}$$

where $\underline{\sigma}$ is the minimum singular value, and $GK(j\omega)$ is the transfer function of plant and controller together.

To be robust against large uncertainties at high frequencies, according to (2.4), the loop gain, $GK(j\omega)$, should be kept low. Intuitively, penalizing inputs at the high frequencies reduces the feedback gain at those frequencies, which allows for larger uncertainty magnitude, $l_m(\omega)$. We therefore propose to use the following frequency-dependent input weighting

$$\tilde{R}(\omega) = \left|\frac{1+\beta j\omega}{1+\alpha j\omega}\right|^2 R, \qquad \text{with} \quad \beta > \alpha, \tag{2.5}$$

where R is the original input cost, and $-\beta^{-1}$ and $-\alpha^{-1}$ are the zero and pole of the loopshaping transfer function. A visualization of such cost function is provided in Fig. 2.2.

Indeed, for Single-Input Single-Output (SISO) systems Anderson *et al.* (Anderson and Mingori, 1985) established that the open loop gain at high frequency under the frequency-shaped cost function (2.5) is reduced, *i.e.*,

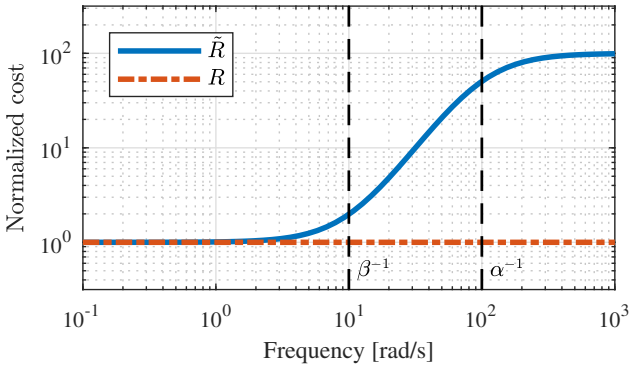


Figure 2.2: Example of the frequency-shaped cost function for $\alpha = 0.01$, $\beta = 0.1$, and the standard input costs in the frequency domain. Costs are normalized by R.

$|GK_{\tilde{R}}(j\omega)| < |GK_R(j\omega)|$ for large ω, where $K_{\tilde{R}}$ and K_R are the Linear Quadratic Regulator gains obtained under the frequency-shaped cost and baseline cost respectively. According to (2.4), the resulting increase in $\underline{\sigma}[I + GK(j\omega)^{-1}]$ permits a higher model uncertainty in the stopband.

Unfortunately, to the best of our knowledge, a robustness proof for Multiple-Input Multiple-Output (MIMO) systems is not available. Despite that, the intuition that penalizing high frequency input increases compatibility with actuator bandwidth limits remains. In this paper, we aim to empirically validate the effect of using such a cost function.

2.2.2 Frequency-shaped Cost Functions

MPC plans over a receding horizon. The cost function in (2.5) therefore needs to be expressed in the time domain as well. This can be achieved by a state augmentation as described in (Gupta, 1980).

The standard quadratic cost function for the time domain (2.1) can be converted to the frequency domain (2.6) according to Parseval's theorem:

$$J = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(\hat{\mathbf{x}}(\omega)^H \mathbf{Q}\, \hat{\mathbf{x}}(\omega) + \hat{\mathbf{u}}(\omega)^H \mathbf{R}\, \hat{\mathbf{u}}(\omega) \right) \mathrm{d}\omega, \tag{2.6}$$

where $\hat{\mathbf{x}}(\omega)$, and $\hat{\mathbf{u}}(\omega)$ are the Fourier transform of $\mathbf{x}(t)$ and $\mathbf{u}(t)$, and $(\cdot)^H$ is the Hermitian transpose of the vector. Here, it becomes apparent that the standard costs over states and inputs are constant for all frequencies. To

leave the possibility of having different loopshaping per input dimension, the frequency-dependent weight matrix in (2.5) is generalized to

$$\tilde{\mathbf{R}}(\omega) = \begin{bmatrix} r_1^*(\omega) & & \\ & \ddots & \\ & & r_m^*(\omega) \end{bmatrix} \mathbf{R} \begin{bmatrix} r_1(\omega) & & \\ & \ddots & \\ & & r_m(\omega) \end{bmatrix},$$

$$r_i(\omega) = \frac{1 + \beta_i j\omega}{1 + \alpha_i j\omega}, \qquad \beta_i > \alpha_i, \tag{2.7}$$

where $r_i^*(\omega)$ is the complex conjugate of $r_i(\omega)$. Every input direction can now have its own shaping function r_i. In order to transfer this new cost function back into the time domain, a change of variables is required. The filtered inputs, $\hat{\boldsymbol{\nu}}(\omega)$, are defined elementwise as

$$\hat{\nu}_i(\omega) = r_i(\omega)\hat{u}_i(\omega). \tag{2.8}$$

After substitution, we arrive back at a frequency-independent cost function over the filtered variables in (2.10). This cost is then converted back to the time domain in (2.11),

$$J = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(\hat{\mathbf{x}}(\omega)^H \mathbf{Q}\, \hat{\mathbf{x}}(\omega) + \hat{\mathbf{u}}(\omega)^H \tilde{\mathbf{R}}(\omega)\, \hat{\mathbf{u}}(\omega) \right) \mathrm{d}\omega \tag{2.9}$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(\hat{\mathbf{x}}(\omega)^H \mathbf{Q}\, \hat{\mathbf{x}}(\omega) + \hat{\boldsymbol{\nu}}(\omega)^H \mathbf{R}\, \hat{\boldsymbol{\nu}}(\omega) \right) \mathrm{d}\omega \tag{2.10}$$

$$= \frac{1}{2} \int_{0}^{\infty} \left(\mathbf{x}(t)^\top \mathbf{Q}\, \mathbf{x}(t) + \boldsymbol{\nu}(t)^\top \mathbf{R}\, \boldsymbol{\nu}(t) \right) \mathrm{d}t. \tag{2.11}$$

2.2.3 Implementation

The presence of the filtered inputs $\boldsymbol{\nu}(t)$ in the cost function (2.11) requires the augmentation of the original problem. Considering the original system dynamics, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, and state input constraint, $\mathbf{g}(\mathbf{x}, \mathbf{u}) \leq \mathbf{0}$, this can be achieved in several ways. If $\mathbf{r}(\omega)$ consists of proper rationals, its state space realization[1], ($\mathbf{A}_r$, $\mathbf{B}_r$, $\mathbf{C}_r$, $\mathbf{D}_r$), can be used to substitute for $\boldsymbol{\nu}$ in the cost function. The filter's internal dynamics are appended to the system.

$$J = \frac{1}{2} \int_{0}^{\infty} \left(\mathbf{x}^\top \mathbf{Q}\, \mathbf{x} + (\mathbf{C}_r \mathbf{x}_r + \mathbf{D}_r \mathbf{u})^\top \mathbf{R}\, (\mathbf{C}_r \mathbf{x}_r + \mathbf{D}_r \mathbf{u}) \right) dt \tag{2.12}$$

$$\begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\mathbf{x}}_r\end{bmatrix} = \begin{bmatrix}\mathbf{f}(\mathbf{x},\mathbf{u})\\ \mathbf{A}_r\mathbf{x}_r + \mathbf{B}_r\mathbf{u}\end{bmatrix}, \quad \mathbf{g}(\mathbf{x},\mathbf{u}) \leq \mathbf{0}, \tag{2.13}$$

If $\mathbf{r}(\omega)$ consists of improper rationals, the transfer function $\mathbf{s}(\omega) = \mathbf{r}^{-1}(\omega)$ is defined such that $\hat{u}_i(\omega) = s_i(\omega)\hat{\nu}_i(\omega)$. The state space realization[1] of $\mathbf{s}(\omega)$, ($\mathbf{A}_s$, $\mathbf{B}_s$, $\mathbf{C}_s$, $\mathbf{D}_s$), is used to substitute for $\mathbf{u}$ in both system dynamics and constraints. Again, the filter's internal dynamics are added to the system.

$$J = \frac{1}{2}\int_0^\infty \left(\mathbf{x}^\top \mathbf{Q}\,\mathbf{x} + \boldsymbol{\nu}^\top \mathbf{R}\,\boldsymbol{\nu}\right) \mathrm{d}t, \tag{2.14}$$

$$\begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\mathbf{x}}_s\end{bmatrix} = \begin{bmatrix}\mathbf{f}(\mathbf{x}, \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu})\\ \mathbf{A}_s\mathbf{x}_s + \mathbf{B}_s\boldsymbol{\nu}\end{bmatrix}, \ \mathbf{g}(\mathbf{x}, \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}) \leq \mathbf{0}, \tag{2.15}$$

The system inputs, $\mathbf{u} = \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}$, are then retrieved after optimization.

Because the selected class of shaping functions are of relative degree zero, choosing between these equivalent options is a numerical consideration. Since the poles of each $r_i(\omega)$ are in our case higher than its zeros, the dynamics of $\mathbf{A}_r$ are faster than those of the realization of its inverse, $\mathbf{A}_s$, we therefore use the latter formulation.

2.2.4 Related Methods

The relation with higher order formulations can be understood by considering a shaping function of $r_i(\omega) = j\omega$ for each input in (2.7) instead of $\frac{1+\beta_i j\omega}{1+\alpha_i j\omega}$. The resulting state space realization is

$$\dot{\mathbf{x}}_s = \boldsymbol{\nu}, \qquad \mathbf{u} = \mathbf{x}_s, \tag{2.16}$$

which is equivalent to $\dot{\mathbf{u}} = \boldsymbol{\nu}$, *i.e.*, the auxiliary input is the derivative of the original input. By selecting $r_i(\omega) = (j\omega)^n$, general n-th order formulations can be retrieved. Higher order methods can thus be seen as a special case of frequency shaping. The proposed method considers more general transfer functions, allowing the user with more flexibility to target a specific frequency range.

There is, however, a numerical consideration when using frequency shaping functions that go to infinity as ω goes to infinity. For such an transfer function $\mathbf{s}(\omega)$ will be strictly proper and have a realization with $\mathbf{D}_s = 0$. As can be seen

[1] State space realizations are computed according to the balanced realization described in (Moore, 1981)

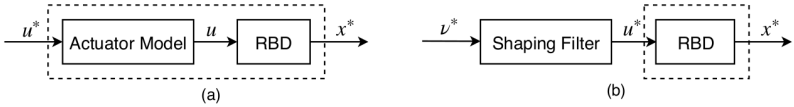


Figure 2.3: Block diagrams for actuator modeling (a) and frequency shaping (b). RBD denotes the rigid body dynamics. $\mathbf{u}^*$ and $\mathbf{x}^*$ are the optimized system input and state references to be tracked by the robot.

from the constraints in (2.15), this results in pure-state constraints. This is not a problem in theory, but such constraints are computationally more expensive to handle than state-inputs constraints, so we avoid them in practice to achieve fast replanning.

When using a higher order formulation, inequality constraints can be used to put hard limits on the smoothness of the trajectories. Similar constraints can be placed in the frequency based method as done in (Hours et al., 2015). Such a discussion is thus rather a preference for designing behavior through costs or constraints.

The difference between the proposed method and embedding an actuator model can be understood from Fig. 2.3. When embedding an actuator model in the system dynamics, one would interpret the input to that model as the command to be sent to the robot. However, in the proposed method, the command sent to the robot is the original input, leaving the assumed relation between $\mathbf{x}$ and $\mathbf{u}$ unchanged. In the former, the optimized state input trajectories, $\{\mathbf{x}^*, \mathbf{u}^*\}$ relies on the accuracy of the actuator model, while in the latter, the filter is used to restrict input frequency content to a feasible range.

2.3 Experimental Setup

2.3.1 Problem Formulation

The proposed method is applied to the kinodynamic model of a quadruped robot, which describes the dynamics of a single free-floating body along with the kinematics for each leg. Equations of Motion (EoM) are given by

$$\begin{cases} \dot{\boldsymbol{\theta}} = \mathbf{T}(\boldsymbol{\theta})\boldsymbol{\omega}, \\ \dot{\mathbf{p}} = {}_{\mathrm{W}}\mathbf{R}_{\mathrm{B}}(\boldsymbol{\theta})\,\mathbf{v}, \\ \dot{\boldsymbol{\omega}} = \mathbf{I}^{-1}\left(-\boldsymbol{\omega}\times\mathbf{I}\boldsymbol{\omega} + \sum_{i=1}^{4}\mathbf{r}_{\mathrm{EE}i}(\mathbf{q})\times\boldsymbol{\lambda}_{\mathrm{EE}i}\right), \\ \dot{\mathbf{v}} = \mathbf{g}(\boldsymbol{\theta}) + \frac{1}{m}\sum_{i=1}^{4}\boldsymbol{\lambda}_{\mathrm{EE}i}, \\ \dot{\mathbf{q}} = \mathbf{u}_J, \end{cases}$$

where ${}_{\mathrm{W}}\mathbf{R}_{\mathrm{B}}$ and $\mathbf{T}$ are the rotation matrix of the base with respect the global frame and the transformation matrix from angular velocities in the base frame to the Euler angles derivatives in the global frame. $\mathbf{g}$ is the gravitational acceleration in body frame, $\mathbf{I}$ and m are the moment of inertia about the CoM and the total mass respectively. The inertia is assumed to be constant and taken at the default configuration of the robot. $\mathbf{r}_{\mathrm{EE}i}$ is the position of the foot i with respect to CoM. $\boldsymbol{\theta}$ is the orientation of the base in Euler angles, $\mathbf{p}$ is the position of the CoM in world frame, $\boldsymbol{\omega}$ is the angular rate, and $\mathbf{v}$ is the linear velocity of the CoM. $\mathbf{q}$ is the vector of twelve joint positions. The inputs of the model are the joint velocity commands $\mathbf{u}_J$ and end effector contact forces $\boldsymbol{\lambda}_{\mathrm{EE}i}$.

The constraints depending on the mode of a leg at that point in time are formulated as

$$\begin{cases} \mathbf{v}_{\mathrm{EE}i} = \mathbf{0}, \quad \boldsymbol{\lambda}_{\mathrm{EE}i} \in \mathcal{C}(\hat{\mathbf{n}}, \mu), & \text{if } i \text{ is a stance leg} \\ \mathbf{v}_{\mathrm{EE}i}\cdot\hat{\mathbf{n}} = c(t), \quad \boldsymbol{\lambda}_{\mathrm{EE}i} = \mathbf{0}, & \text{if } i \text{ is a swing leg} \end{cases}$$

where $\mathbf{v}_{\mathrm{EE}i}$ is the end effector velocity in world frame, which constrains a stance leg to remain on the ground and a swing leg to follow the predefined curve $c(t)$ in the direction of the local surface normal, $\hat{\mathbf{n}}$, to avoid foot scuffing with zero contact force inputs $\boldsymbol{\lambda}_{\mathrm{EE}i}$. This curve ends with a negative velocity of $0.75\,\mathrm{m/s}$, which is maintained until contact is detected. The friction cone, $\mathcal{C}(\hat{\mathbf{n}}, \mu)$, is defined by the surface normal and friction coefficient, $\mu = 0.7$. This constraint is enforced by projecting the inputs onto the feasible set in the forward rollout of the SLQ-MPC algorithm. Limitations of such a clamping strategy are discussed in (Tassa, Mansard, and Todorov, 2014). In practice, we

find that these constraints are rarely active and that the projection is sufficient for the motions in this work.

The baseline cost is formulated as a quadratic function

$$
\begin{aligned}
J &= \Phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t, \\
L &= \frac{1}{2} \left(\mathbf{x} - \mathbf{x}_d\right)^\top \mathbf{Q} \left(\mathbf{x} - \mathbf{x}_d\right) + \frac{1}{2} (\mathbf{u} - \mathbf{u}_0)^\top \mathbf{R} (\mathbf{u} - \mathbf{u}_0), \\
\Phi &= \frac{1}{2} \left(\mathbf{x} - \mathbf{x}_d\right)^\top \mathbf{Q}_T \left(\mathbf{x} - \mathbf{x}_d\right),
\end{aligned} \tag{2.17}
$$

where $\mathbf{x}_d = [\boldsymbol{\theta}_{\text{des}}^\top, \mathbf{p}_{\text{des}}^\top, \boldsymbol{\omega}_{\text{des}}^\top, \mathbf{v}_{\text{des}}^\top, \mathbf{q}_0^\top]^\top$ is a desired state consisting of commanded base pose and twist by the user and a default configuration for the joints. Inputs are defined as $\mathbf{u} = [\boldsymbol{\lambda}_{\text{EE}_1}^\top, \ldots, \boldsymbol{\lambda}_{\text{EE}_4}^\top, \mathbf{u}_j^\top]^\top$, and $\mathbf{u}_0$ is the equilibrium input for standing in the default configuration. $\Phi(\cdot)$ is the final state cost, which is a heuristic to approximate the truncated infinite horizon, and is implemented as a diagonal cost on the base pose and velocities. $L(\cdot, \cdot)$ is the intermediate cost where we use a simple diagonal cost on all state variables and contact force inputs. For the costs on the joint velocities, a diagonal matrix is pre- and post-multiplied by the end-effector Jacobians to define costs over the task space.

2.3.2 System Integration

In the model described in the previous section, the control inputs are end-effector forces and joint velocities. To translate the solution to torque commands, we extract a full position, velocity, and acceleration plan for CoM and end-effector trajectories, in addition to the planned contact forces. This plan is tracked by the hierarchical Whole-Body Controller (WBC) architecture described in (Dario Bellicoso et al., 2016). The tasks in decreasing priority are (1) satisfying the equation of motions and zero acceleration for contact feet (2) tracking CoM and swing leg trajectories, and (3) tracking the planned contact forces. The desired contact forces from the MPC are thus communicated in two ways: The CoM trajectory dictates the net acting forces, and force tracking task regulates the internal forces. Without the latter, contact forces would be redistributed among the contacts, which would override the planned smoothness of the trajectories. Additionally, on all priorities, torque limits and friction cone constraints are imposed as inequality constraints.

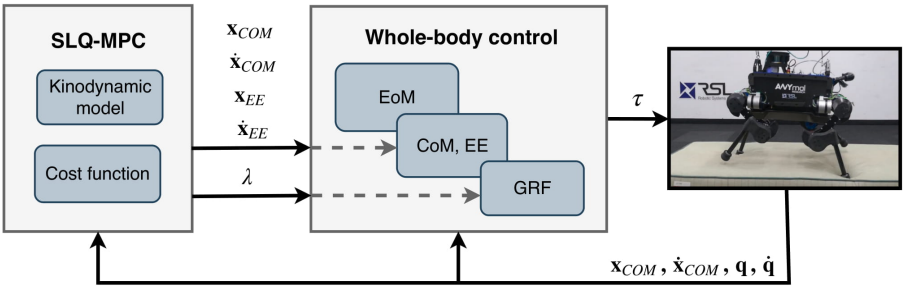


Figure 2.4: MPC and whole-body control structure overview. The SLQ-MPC algorithm running on a separate desktop PC sends CoM and End-Effector (EE) reference to the onboard whole-body control structure. This hierarchical controller computes torque commands based on the listed priorities.

The optimal control problem in (2.17) is solved for a user-defined gait with the continuous time SLQ-MPC algorithm described in (Farshidian et al., 2017a). We use a receding horizon length of 1.0 s, which results in an MPC update rate of 70 Hz for the baseline method and around 40 Hz for the frequency-shaped method.

The MPC runs on a desktop PC with an Intel Core i7-8700K CPU@3.70 GHz hexacore processor and continuously computes a motion plan from the latest known state through a real-time-iteration scheme. The WBC runs on the dedicated onboard PC and tracks the most recent plan. Here, the augmented filter state is propagated as well with the currently available augmented input plan $\boldsymbol{\nu}(t)$. Both nodes communicate over a local network. An overview of the experimental setup is provided in Fig. 2.4.

2.4 Results

We study the effects of adopting the cost function in (2.11) for the previously described setup under various locomotion tasks. To see the results at different levels of model errors, we conduct perfect model simulations, rigid-body simulations, and hardware experiments. When selecting different values for β, α is selected such that the frequencies in the stopband incur a cost of 100 times the steady state cost, *i.e.*, $\alpha = 0.1\beta$.

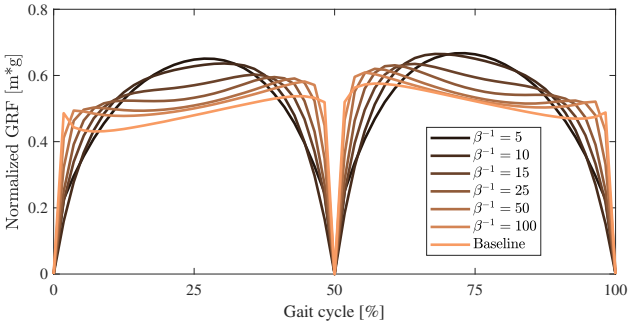


Figure 2.5: Ground reaction forces in z-direction for a trotting gait at 0.5 m/s with a period of 0.7 s. The first half of the gait cycle corresponds to the left front foot and the second half to the left hind foot. As the frequency limit decreases from infinity (*i.e.*, baseline) to 5, the smoothness of the planned contact forces increases.

2.4.1 Perfect Model

First, we investigate the effect of the loopshaping on the contact forces in a simulation that uses the same model as the MPC. This shows how the resulting trajectories are different already in the case of no modeling errors. The analyzed gait is a trot with a duty factor of 0.5, *i.e.*, with no overlap in stance phases of the diagonal feet, while a forward velocity of 0.5 m/s is commanded. Fig. 2.5 shows the ground reaction forces when selecting different values for β. As seen from the plot, the baseline method instantaneously applies contact forces once a foot is in contact. As expected, lowering the frequency at which costs start to increase, *i.e.*, lowering β^{-1}, results in increasingly smooth trajectories. The frequency-shaped method approaches the baseline as β^{-1} goes to infinity. The corresponding base height trajectories are plotted in Fig. 2.6. Smoother contact force trajectories require more vertical displacement of the base, while the baseline produces the exact amount of force to keep the base level.

2.4.2 Physics Simulation

The combination of tracking controller and MPC is evaluated in the Open Dynamics Engine (ODE) (Smith et al., 2005) rigid-body simulation, where we can vary ground properties in a controlled way. The model errors, in this case, come from the difference between rigid-body dynamics and the kinodynamic model, as well as the assumption of a rigid ground contact when the terrain

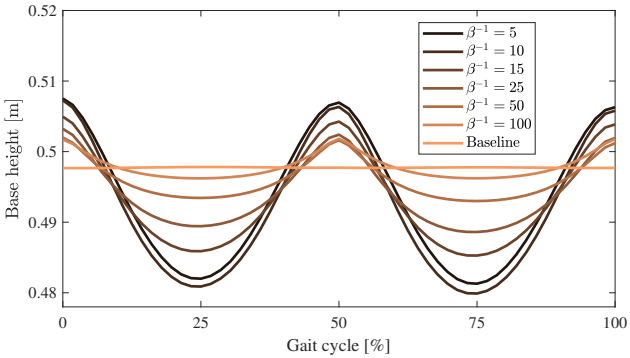


Figure 2.6: Base height for a trotting gait at 0.5 m/s with a period of 0.7 s under different smoothing parameters. In general, the vertical displacement of the base increases as the controller becomes smoother since it cannot abruptly increase or decrease the ground reaction forces at the stance feet.

is made compliant. ODE allows for simulation of soft contacts by modeling[2] the ground contact forces as a spring-damper system. Three different sets of spring-damper parameters k_p and k_d are selected to simulate hard, intermediate, and soft ground, respectively. For each terrain, three different cost functions are evaluated: the baseline without frequency shaping as well as frequency-dependent cost functions with $\beta^{-1} = 50$ and $\beta^{-1} = 10$. These values were selected based on Fig. 2.5 to represent three levels of smoothness in the continuum of available cost functions. The resulting contact force profiles for all combinations are shown for a single stance phase in Fig. 2.7. Desired and measured contact forces are shown for a single leg during an in-place trotting motion with a stance duration of 0.35 seconds.

As the compliance of the terrain increases, the difference between desired forces generated from the WBC and resulting forces grows. The WBC uses rigid-body dynamics with a hard contact assumption to compute desired contact forces. The difference between desired and measured forces is therefore a measure of disturbances inserted by additional unmodeled dynamics, which in general includes the bandwidth limits of actuators and contact dynamics that we aim to avoid. Table 2.1 shows the force tracking performance averaged

[2]ODE relaxes the rigid-contact solver such that it implicitly resembles a spring-damper interaction.

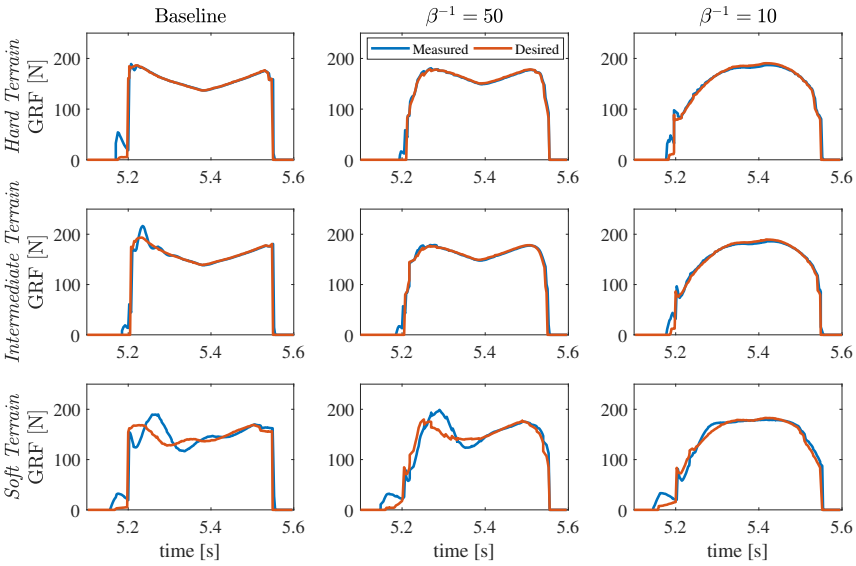


Figure 2.7: Measured and desired contact forces in z-direction during a trot in-place for various ground properties and cost functions. The columns from left to right correspond to the baseline, $\beta^{-1} = 50$, and $\beta^{-1} = 10$ cost functions. The rows, from top to bottom corresponds to ground properties with k_p and k_d of $\{1e6, 100\}$, $\{1e5, 50\}$, and $\{1e4, 30\}$. As the compliance of the terrain increases, the difference between desired forces generated from the whole-body controller and resulting forces grows.

over six gait cycles in Mean Absolute Error (MAE) and Mean Squared Error (MSE) defined as

$$\text{MAE} = \frac{1}{T}\int_0^T |\boldsymbol{\lambda} - \boldsymbol{\lambda}_{\text{des}}|\,\mathrm{d}t, \quad \text{MSE} = \frac{1}{T}\int_0^T (\boldsymbol{\lambda} - \boldsymbol{\lambda}_{\text{des}})^2\,\mathrm{d}t.$$

For all cost functions, tracking performance degrades as the model error increases. Qualitatively, the baseline controller suffers from a larger error at the beginning and end of the contact phase, due to its step-like change in the desired forces. Even on hard terrain, there is an apparent benefit of loopshaping. The smoother transition between contact phases mitigates the disturbance generated by contact timing mismatch. Differences become most apparent for the soft terrain, where the smoother trajectory has better performance especially in MSE.

Table 2.1: Force tracking performance MAE (MSE) [N (N^2)] for different terrain and cost functions in simulation.

	Cost function		
Terrain $\{k_p, k_d\}$	Baseline	$\beta^{-1} = 50$	$\beta^{-1} = 10$
Hard $\{1e6, 100\}$	4.8 (303.5)	**3.6** (58.5)	5.0 (104.2)
Medium $\{1e5, 50\}$	5.5 (316.2)	**4.6** (110.1)	5.0 (100.4)
Soft $\{1e4, 30\}$	13.5 (525.5)	11.6 (286.9)	**7.4** (146.5)

Furthermore, we examine the locomotion strategy under two extrema of cost functions. The commanded forward velocity during a trotting motion is gradually increased until failure occurs. The foot placement strategies are visualized in Fig. 2.8. The plots show footstep locations from a top-view with the robot starting at the origin. As the velocity increases towards the right side of the plot, the footstep locations start to differ. Interestingly, with the smoother cost function of $\beta^{-1} = 10$, the foot placement strategy is significantly altered and becomes velocity dependent. Where under the baseline costs the controller chooses a fixed width foot placement, the frequency-shaped solution places the feet increasingly inwards for higher velocities. This can be explained by realizing that horizontal forces, like the normal forces in Fig. 2.5, smoothly start from and end at zero. During a switch from one contact pair to the next, a lateral force is required to make the CoM velocity change in the direction of the next support line. Under the frequency-dependent cost function, high lateral forces around the contact switch are expensive, and a solution where the supports are more aligned is thus preferred. As the forward velocity increases, so does the required change in lateral velocity for a given width. The alignment of support lines therefore gradually becomes more pronounced.

Remarkably, this results in a significantly higher maximum velocity of 0.9 m/s versus 0.6 m/s.

2.4.3 Hardware

On hardware, we aim to validate the simulation results for contact force tracking performance on different terrains; The floor of the lab, a 3.5 cm foam block, and a mattress are selected to test a rigid, an intermediate and a very compliant terrain respectively. A force-torque sensor is mounted on the right front leg to obtain direct measurements of the ground reaction forces. The resulting measured and desired forces for different cost function and terrain combina-

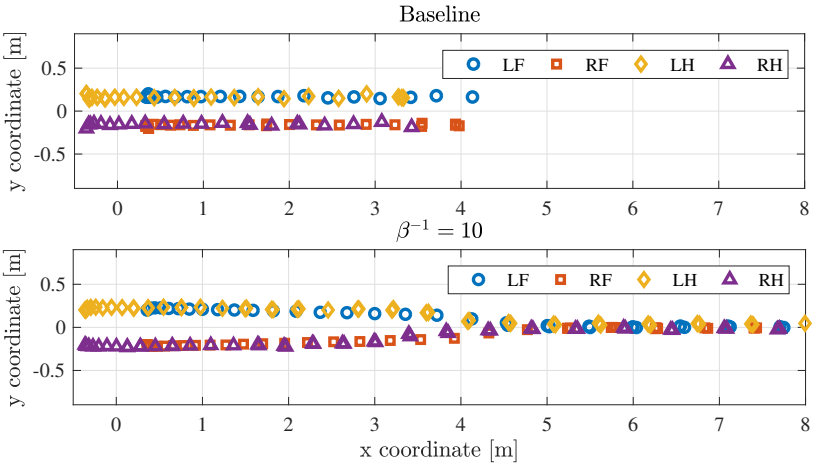


Figure 2.8: Foot placement strategy for the baseline (top) and the $\beta^{-1} = 10$ (bottom) cost functions during a trot with increasing velocity commands in the positive x-direction. Commands start at $v = 0\,\mathrm{m/s}$ and accelerate with $0.05\,\mathrm{m/s^2}$. Failure occurs at $0.6\,\mathrm{m/s}$ and $0.9\,\mathrm{m/s}$ respectively. The footstep strategy of the baseline method is different from the frequency-aware solution. While the former minimizes the lateral motion of the Center of Pressure (CoP) by aligning the support polygons (lines), the baseline method does not adapt its footstep planning based on the velocity.

tions are shown in Fig. 2.9. The plots show the difference between measured and desired contact forces of the right-front leg during the first three steps. The MAE and MSE averaged over those first three gait cycles are given in Table 2.2. On hard terrain, all methods perform well, and slight differences in tracking performance occur at the beginning and end of the contact phase. In this area the $\beta^{-1} = 10$ controller provides the smoothest transition, resulting in the best MAE and MSE on all terrains. Where in the simulation experiments we see that a medium amount of smoothing is best for hard and medium stiff terrain, we do not see this in the hardware experiments. The difference could be caused by the series elastic actuators of ANYmal, causing model errors and bandwidth limits even on hard terrain. For a step input to the torque level reached during trot, a 90% rise-time up to 35 ms, equivalent to a bandwidth of around 60 rad/s can be expected (Hutter et al., 2016).

When further reducing the compliance by trotting on the mattress, the baseline and $\beta^{-1} = 50$ controllers suffer a substantial decrease in performance. The base height during the first part of the experiment is shown in Fig. 2.10. Due

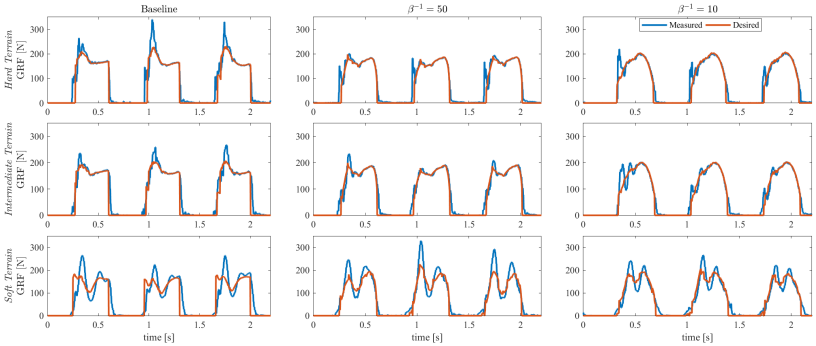


Figure 2.9: Measured and desired contact forces for the right front foot during the first three trotting gait cycles on the lab floor (top row), foam (middle row) and a mattress (bottom row) with the baseline (left column), $\beta^{-1} = 50$ (middle column), and $\beta^{-1} = 10$ (right column) cost functions. A smoother transition between swing and stance phase results in better tracking performance.

Table 2.2: Force tracking performance MAE (MSE) [N (N^2)] for different terrain and cost functions on hardware.

	Cost function		
Terrain	Baseline	$\beta^{-1} = 50$	$\beta^{-1} = 10$
Hard	27.1 (1465.7)	14.7 (785.5)	**12.7** (641.4)
Medium	21.4 (1040.1)	19.7 (741.3)	**16.0** (555.4)
Soft	30.9 (2308.1)	26.0 (1237.5)	**22.0** (824.3)

to the significant mismatch between the planned and resulting contact force with each footstep, the baseline controller loses base height in a few steps, causing it to fail. The $\beta^{-1} = 50$ cost function does achieve a trot, as shown in the video[3], but strong oscillations are present between the terrain and the feet. The $\beta^{-1} = 10$ cost function, finally, results in both a stable trot and a smooth contact interaction.

In the accompanying video, we additionally show the behavior under disturbances. The robot trots in place and has costs on base deviation from the initial position. We disturb the robot in the horizontal plane. Qualitatively, the reactive stepping and push-back behavior differ. Under the baseline method, the

[3] `https://youtu.be/RSJgqkk2VRI`

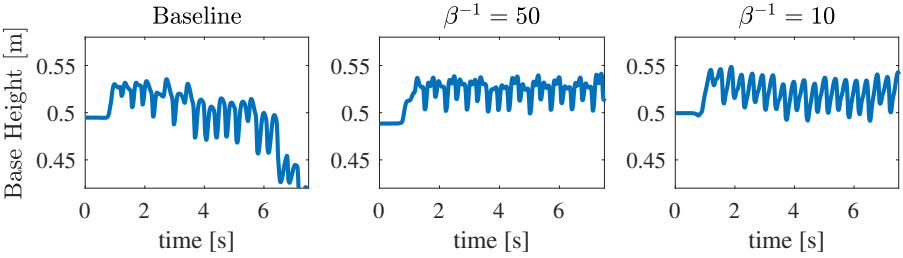


Figure 2.10: Measured base height during the first seconds of trotting motion on the mattress with the baseline (left), $\beta^{-1} = 50$ (middle), and $\beta^{-1} = 10$ (right) cost functions. Where the baseline method fails to maintain its height, the frequency-aware controllers remain stable.

robot reacts to a push by generating lateral forces and the user experiences instant resistance. The frequency-aware implementation with $\beta^{-1} = 10$ instead accepts deviation of the base trajectory and adapts future step location and force profile to smoothly return to the origin.

2.5 Discussion

We have shown that a single parameter of a frequency-dependent cost function provides a handle on a rich variety of solutions. While the smoothness at the beginning of the stance phase is not surprising due to the filter, the anticipatory decrease in force before lifting the foot, as seen in Fig. 2.5, shows that the filter and planning are tightly working together. The additional filter states allow the Riccati-type algorithm to reason about future state-input constraints, in this case, zero contact forces during the swing phase, and adapts the strategy to approach them smoothly. This is remarkable because the backward pass is projected on those constraints only at the point in time that they are active. Interestingly we also find that the foot placement strategy changes significantly. These observations highlight the fact that embedding frequency awareness of the MPC is richer than simply filtering the obtained inputs. As a future work, we will explore ways to change to cost function online and in this way adapt the locomotion strategy to the terrain.

For the legs in swing phase, optimizing over joint torque, which is effectively one derivative higher than optimizing over joint velocities, can also induce a smoother swing trajectory. However, obtaining real time performance with

the extra nonlinear effects is challenging. We believe that keeping the model as simple as possible helps to get a robust solution working on hardware.

2.6 Conclusion

We introduced frequency-aware MPC by combining frequency-dependent cost functions with modern MPC methods. With simulation experiments, we show that the resulting smoother force profiles improve tracking performance when the rigid terrain assumption is relaxed, without the need to explicitly model it. We validated these results on hardware and see a similar trend when comparing performance on various terrains. The method is shown to provide robustness against unmodeled dynamics of series elastic actuators and compliant terrains. We demonstrated that with this approach ANYmal is now able to execute dynamic motions even on highly compliant terrains.

3

Feedback MPC for Torque-Controlled Legged Robots

Grandia, R., Farshidian, F., Ranftl, R., and Hutter, M. (2019b). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4730–4737

DOI: 10.1109/IROS40897.2019.8968251

Video: https://youtu.be/KrTrLGDA6FQ

The computational power of mobile robots is currently insufficient to achieve torque level whole-body Model Predictive Control (MPC) at the update rates required for complex dynamic systems such as legged robots. This problem is commonly circumvented by using a fast tracking controller to compensate for model errors between updates. In this work, we show that the feedback policy from a Differential Dynamic Programming (DDP) based MPC algorithm is a viable alternative to bridge the gap between the low MPC update rate and the actuation command rate. We propose to augment the DDP approach with a relaxed barrier function to address inequality constraints arising from the friction cone. A frequency-dependent cost function is used to reduce the sensitivity to high-frequency model errors and actuator bandwidth limits. We demonstrate that our approach can find stable locomotion policies for the torque-controlled quadruped, ANYmal, both in simulation and on hardware.

3.1 Introduction

MPC has gained broad interest in the robotics community as a tool for motion control of complex and dynamic systems. The ability to deal with nonlinearities and constraints has popularized the technique for many robotic applications, such as quadrotor control (Alexis et al., 2011), autonomous racing (Liniger, Domahidi, and Morari, 2015), and legged locomotion (Farshidian et al., 2017b; Koenemann et al., 2015; Tassa, Erez, and Todorov, 2012).

MPC strategies typically optimize an open-loop control sequence for a given cost function over a fixed time horizon. The control sequence is then executed until a new control update is calculated based on the current state estimate. While this strategy assumes that the model is exact and that there are no external disturbances, the repeated optimization provides a feedback mechanism that can correct for modeling errors provided that the control loop can be executed at a sufficiently high rate. However, for high dimensional systems such as legged robots and due to the computational restrictions of mobile platforms, the achievable update rate of the MPC loop is insufficient to effectively deal with model uncertainty and external disturbances.

As a remedy, a separately designed, light-weight motion tracker is often used in practice (Murray et al., 2009). The motion tracker runs at a higher rate than the MPC loop and provides feedback correction to the control sequence that was designed by the MPC. For complex systems such as legged robots it is a challenging task to design a controller that tracks arbitrary motions while satisfying the many constraints arising from the locomotion task. The fundamental problem is that such motion trackers do not look ahead in the horizon and therefore cannot anticipate changes in contact configuration. As an alternative, projected (time-varying) Linear Quadratic Regulator (LQR) have been proposed as a framework to automatically design feedback controllers around a given reference trajectory (Mason et al., 2016; Posa, Kuindersma, and Tedrake, 2016). However, the stabilizing feedback policy is always designed in a secondary stage and often with a different objective function than the one used for computing the optimal trajectories, which leads to inconsistency between the feedback policy and the MPC trajectories.

In this work, we propose a feedback MPC approach for motion control of a legged system and show that the optimized feedback policy can directly be deployed on hardware. We achieve stable locomotion under a very low update rate (15 Hz), and the optimized feedback policy removes the need for a separate motion controller. Furthermore, the modification of the control

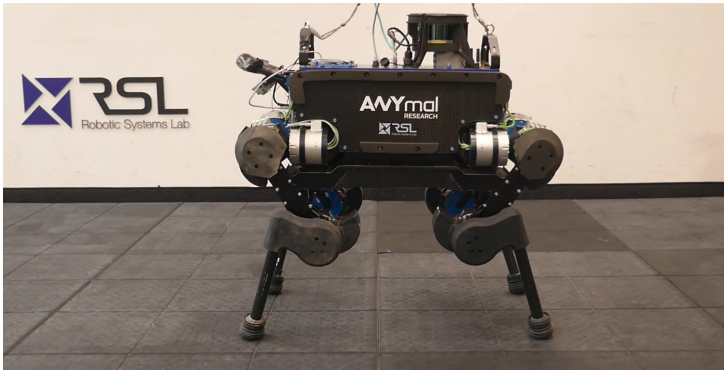


Figure 3.1: ANYmal, the torque controlled-legged robot used in this work.

inputs is consistent with the MPC and thus produces a continuous signal across update instances.

To be able to directly apply the feedback strategy on a legged system, the optimized policy needs to respect all the constraints of the locomotion task such as friction and unilateral constraints of contact forces. To achieve this, we propose to extend the Sequential Linear Quadratic (SLQ) (Sequential Linear Quadratic) algorithm (Farshidian et al., 2017a). We use SLQ in a real-time iteration MPC scheme (Diehl, Bock, and Schlöder, 2005) where the algorithm optimizes a constrained feedback policy, $\boldsymbol{\pi}(\mathbf{x}, t) : \mathbb{X} \times \mathbb{R}^+ \rightarrow \mathbb{U}$,

$$\boldsymbol{\pi}(\mathbf{x}, t) = \mathbf{u}^*(t) + \mathbf{K}(t) \left(\mathbf{x} - \mathbf{x}^*(t)\right), \tag{3.1}$$

where $\mathbf{u}^*(t) \in \mathbb{U}$ and $\mathbf{x}^*(t) \in \mathbb{X}$ are locally optimal input and state trajectories. $\mathbf{K}(t)$ is a time-varying LQR gain matrix which maps the state deviation from $\mathbf{x}^*$ to an admissible control correction. We extend the algorithm to problems with inequality constraints using a barrier function method to accurately handle the constraints arising from the friction cone. We further use the frequency-aware MPC approach introduced in (Grandia et al., 2019a) to render the resulting feedback policy robust to the bandwidth limitations imposed by real actuators.

We perform experiments in simulation and on a real legged system (Fig. 3.1) and demonstrate that our approach is able to find robust and stable locomotion polices at MPC update rates as low as 15 Hz, which facilitates onboard execution on mobile platforms with limited computational power.

3.1.1 Related Work

Methods to incorporate robustness explicitly into the MPC methodology have been previously studied in the literature (Mayne et al., 2000). Min-max MPC (Bemporad and Morari, 1999), for example, optimizes an open-loop control sequence for the worst-case disturbance inside a predefined set. While this formulation appears attractive, it can be overly conservative due to its inability to include the notion of feedback that is inherently present in the receding-horizon implementation of the control (Lee and Yu, 1997).

Min-max Feedback MPC was proposed to address this shortcoming by planning over a state-dependent control policy instead of an open-loop feedforward sequence (Scokaert and Mayne, 1998). Unfortunately optimizing the feedback policy for all possible disturbance realizations does not yet scale to the problem dimensions encountered in legged robotics. However, even without considering disturbances, optimizing over the feedback policy has an additional advantage. When the update rate of the MPC loop is low, the feedback policy can provide local correction to the deviation of the real platform from the optimal trajectories. Exploiting this additional aspect of feedback MPC has not yet been fully explored in robotic applications that are subject to path constraints.

The feedback policy that minimizes a cost function for a given dynamical system and path constraints can be computed using the Hamilton-Jacobi-Bellman (HJB) equation (Bertsekas, 1995). While directly solving this equation for high dimensional systems is prohibitively complex, a variant of the dynamic programming approach known as DDP (Mayne, 1966) has proven to be a powerful tool in many practical applications. The SLQ method that we use in this work is a DDP-based approach which uses a Gauss-Newton approximation. Consequently, it only considers the linearized dynamics instead of a second-order approximation.

Although using the LQR gains derived from a DDP-based approach directly for motion tracking generates promising results in simulation, it dramatically fails on real hardware. This phenomenon has been reported before in other real-world applications of LQR on torque-controlled robots (Mason, Righetti, and Schaal, 2014; Mason et al., 2016). Focchi et al. (Focchi et al., 2016) have shown that instability can occur if the limitations of the low-level torque controller are neglected in the high-level control design. They have argued that the bandwidth of the low-level controller inversely relates to the achievable impedance of the high-level controller. To this end, to apply the SLQ feedback policy on hardware, we need to encode these bandwidth limitations in

our optimization problem. In this work, we use the frequency-aware MPC approach introduced in (Grandia et al., 2019a). This MPC formulation penalizes control actions in the frequency domain and automatically finds a trade-off between the bandwidth limitation of actuators and the stiffness of the high-level feedback policy.

3.1.2 Contributions

We propose a whole-body MPC approach for legged robots, where the actuation commands are computed directly based on the MPC feedback policy. Specifically we present the following contributions which we empirically validate on the ANYmal platform (Fig. 3.1) in simulation and on real hardware:

- We propose to apply feedback MPC for whole-body control of a legged system. To the best of our knowledge, this is the first time that such a control scheme is applied on hardware for motion control of legged robots.
- The SLQ algorithm is extended to include inequality constraints through a barrier function method, which allows us to formulate friction cone constraints.
- We show that our feedback MPC algorithm directly designs constraint-satisfactory LQR gains without additional computational cost.
- A frequency domain design approach is used to incorporate actuation bandwidth limits in the MPC formulation to avoid rendering stiff gains. Thus, the feedback gains can be directly applied to the robot.
- We show that the feedback MPC algorithm is capable of bridging the gap between low update-rate MPC and high rate execution of torque commands using only an onboard computer with moderate computational power.

3.2 Method

3.2.1 Problem Definition

Consider the following nonlinear optimal control problem with cost functional

$$\min_{u(\cdot)} \Phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t), t)\, \mathrm{d}t, \tag{3.2}$$

where $\mathbf{x}(t)$ is the state and $\mathbf{u}(t)$ is the input at time t. $L(\cdot)$ is a time-varying running cost, and $\Phi(\cdot)$ is the cost at the terminal state $\mathbf{x}(T)$. Our goal is to find an input trajectory $\mathbf{u}(\cdot)$ that minimizes this cost subject to the following system dynamics, initial condition, and general equality and inequality constraints:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t) \tag{3.3}$$
$$\mathbf{x}(0) = \mathbf{x}_0 \tag{3.4}$$
$$\mathbf{g}_1(\mathbf{x}, \mathbf{u}, t) = \mathbf{0} \tag{3.5}$$
$$\mathbf{g}_2(\mathbf{x}, t) = \mathbf{0} \tag{3.6}$$
$$\mathbf{h}(\mathbf{x}, \mathbf{u}, t) \geq \mathbf{0}. \tag{3.7}$$

The feedback policy which minimizes this problem can be calculated using a DDP-based method. A variant of this method for continuous-time systems known as SLQ is introduced in (Farshidian et al., 2017a), where it solves the above optimization problem in the absence of the inequality constraints in equation (3.7). This method computes a time-varying, state-affine control policy based on a quadratic approximation of the optimal value function in an iterative process. The SLQ approach uses a Lagrangian method to enforce the state-input equality constraints in (3.5). The pure state constraints in (3.6) are handled by a penalty method.

In SLQ, the simulation (forward pass) and the optimization (backward pass) iterations alternate. Once the backward pass is completed, a forward pass computes a new trajectory based on the improved feedback policy. The local, Linear Quadratic (LQ), approximation of the nonlinear optimal control problem is constructed after each forward pass. The LQ model permits an efficient solution of the approximate problem by solving the Riccati differential equation. The feedback policy is then updated with an appropriate linesearch procedure in the direction of the LQ problem's solution.

We follow the same SLQ approach and extend the method with inequality constraints through a relaxed barrier function approach.

3.2.2 Relaxed Barrier Functions

Using a barrier function is a well know technique to absorb inequality constraint into the cost function. For each constraint in a given set of N_{in} inequality constraints, a barrier term $B(h)$ is added to the cost

$$\hat{L}(\mathbf{x}, \mathbf{u}, t) = L(\mathbf{x}, \mathbf{u}, t) + \mu \sum_{i=1}^{N_{in}} B(h_i(\mathbf{x}, \mathbf{u}, t)). \tag{3.8}$$

A widely used barrier function is the logarithmic barrier used in interior-point methods. The optimal solution is approached by letting $\mu \to 0$ over successive iterations. However, a downside of the log-barrier is that it is only defined over the feasible space, and evaluates to infinity outside. Due to the rollout mechanism in the SLQ approach, one cannot ensure that successive iterations remain inside the feasible region at all time. Furthermore, the Hessian of the log-barrier goes to infinity as one approaches the constraint boundary, which results in an ill-conditioned LQ approximation.

The relaxed barrier functions previously proposed for MPC problems addresses both these issues (Feller and Ebenbauer, 2017) and is therefore particularly suitable for the SLQ approach. This barrier function is defined as a log-barrier function on the interior of the feasible space, and switched to a different function at a distance δ from the constraint boundary.

$$B(h) = \begin{cases} -\ln(h), & h \geq \delta, \\ \beta(h; \delta), & h < \delta. \end{cases} \tag{3.9}$$

We use the quadratic extension proposed in (Hauser and Saccon, 2006):

$$\beta(h; \delta) = \frac{1}{2} \left(\left(\frac{h - 2\delta}{\delta} \right)^2 - 1 \right) - \ln(\delta). \tag{3.10}$$

The relaxed barrier function, which is continuous and twice differentiable, is plotted as a function of the constraint value in Fig. 3.2. The quadratic extension puts an upper bound to the curvature of the barrier function, which prevents ill-conditioning of the LQ approximation. Note that by letting $\delta \to 0$, the standard logarithmic barrier is retrieved. Furthermore, it has been shown that the optimal solution can be obtained for a nonzero value of δ (Aguiar et al., 2017), when the gradient of the penalty term is larger than the Lagrange

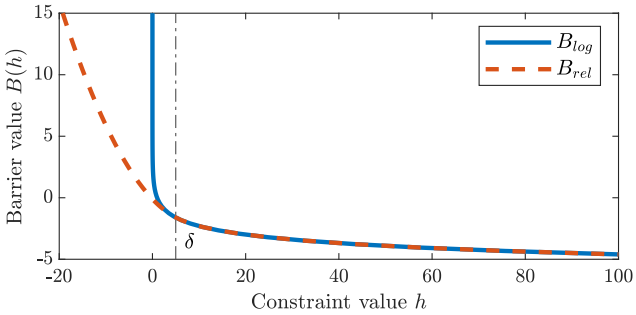


Figure 3.2: Comparison of the log-barrier function $B_{log} = -\ln(h)$ and relaxed barrier function B_{rel} as defined in (3.9) with $\delta = 5$.

multiplier of the associated constraint. Optimization with the relaxed barrier function can thus be interpreted as an augmented Lagrangian approach when $h < \delta$ and as a log-barrier method for $h \geq \delta$.

3.2.3 LQ Approximation

With the inequality constraints embedded in the cost function, we obtain the following linearization of the system dynamics in (3.3) and state-inputs constraints in (3.5) for a given state trajectory $\mathbf{x}_{k-1}(t)$ and input trajectory $\mathbf{u}_{k-1}(t)$:

$$\delta\dot{\mathbf{x}} = \mathbf{A}\delta\mathbf{x} + \mathbf{B}\delta\mathbf{u}, \tag{3.11}$$

$$\mathbf{C}\delta\mathbf{x} + \mathbf{D}\delta\mathbf{u} + \mathbf{e} = \mathbf{0}, \tag{3.12}$$

where $\delta\mathbf{x} = \mathbf{x}(t) - \mathbf{x}_{k-1}(t)$, and $\delta\mathbf{u} = \mathbf{u}(t) - \mathbf{u}_{k-1}(t)$ are deviations from the previous iteration, around which the LQ approximation is made. Note that the time-dependency of the matrices was dropped to shorten the notation. The quadratic approximation of the cost in (3.8) is given by

$$\Phi(\mathbf{x}(T)) \approx q_f + \mathbf{q}_f^\top \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^\top \mathbf{Q}_f^\top \delta\mathbf{x}, \tag{3.13}$$

$$\begin{aligned}\hat{L}(\mathbf{x}, \mathbf{u}, t) \approx\ & q_L(t) + \mathbf{q}_L^\top \delta\mathbf{x} + \mathbf{r}^\top \delta\mathbf{u} + \\ & \frac{1}{2}\delta\mathbf{x}^\top \mathbf{Q}_L \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{u}^\top \mathbf{R}\delta\mathbf{u} + \delta\mathbf{u}^\top \mathbf{P}\delta\mathbf{x},\end{aligned} \tag{3.14}$$

which requires access to the second-order approximation of the barrier term and inequality constraints.

With the optimal control problem reduced to an equality constrained LQ approximation, the constrained Riccati backward pass in (Farshidian et al., 2017a) yields the quadratic value function $V(\mathbf{x}, t) = \frac{1}{2}\mathbf{x}^\top \mathbf{S}(t)\mathbf{x} + \mathbf{s}(t)^\top \mathbf{x} + s(t)$. This value function induces the optimal feedback policy in equation (3.1) with feedback gains computed as

$$\mathbf{K}(t) = \left(\mathbf{I} - \mathbf{D}^\dagger \mathbf{D}\right) \mathbf{R}^{-1} \left(\mathbf{P}^\top + \mathbf{B}^\top \mathbf{S}\right) + \mathbf{D}^\dagger \mathbf{C}, \tag{3.15}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{D}^\dagger$ is the right pseudo-inverse of the full row rank matrix $\mathbf{D}$. Notice how the feedback gains ensure that the equality constraints are satisfied by projecting the first term to the nullspace of the constraints, and by adding the term $\mathbf{D}^\dagger \mathbf{C}$ to satisfy the constraint when the state deviates from the plan.

3.2.4 Frequency Shaping

As discussed in Section 3.1.1, it has been proven difficult to use feedback gains from an LQR design on a torque-controlled robot. We propose to use the frequency-dependent cost function introduced in our previous work (Grandia et al., 2019a), which was used to render the feedforward solution robust to high frequency disturbances. In this work, we show that it has a similar effect on the feedback structure. We briefly summarize how the problem is adapted and refer to (Grandia et al., 2019a) for further details.

A frequency-dependent cost on the inputs can be introduced by evaluating the cost function on auxiliary inputs $\boldsymbol{\nu}$. The auxiliary inputs $\boldsymbol{\nu}$ are defined by frequency-dependent shaping functions $r_i(\omega)$ applied to the system inputs such that

$$\hat{\nu}_i(\omega) = r_i(\omega)\hat{u}_i(\omega), \tag{3.16}$$

where i denotes elements associated to individual inputs, ω is the signal frequency in rad/s, and $\hat{\nu}_i(\omega)$ and $\hat{u}_i(\omega)$ are the Fourier transform of the auxiliary input and system input respectively. Following our previous work, we use high pass filters to achieve increased costs at higher input frequencies:

$$r_i(\omega) = \frac{1 + \beta_i j\omega}{1 + \alpha_i j\omega}, \qquad \beta_i > \alpha_i. \tag{3.17}$$

The transfer function $\mathbf{s}(\omega) = \mathbf{r}^{-1}(\omega)$, with state space realization $(\mathbf{A}_s, \mathbf{B}_s, \mathbf{C}_s, \mathbf{D}_s)$, is constructed such that $\hat{u}_i(\omega) = s_i(\omega)\hat{\nu}_i(\omega)$. The original system is augmented with an additional filter state, $\mathbf{x}_s$, such that $\tilde{\mathbf{x}} =$

$[\mathbf{x}^\top, \mathbf{x}_s^\top]^\top$, and optimization is performed w.r.t. the auxiliary inputs $\boldsymbol{\nu}$. The augmented system dynamics and state-input constraints are defined as

$$\begin{bmatrix} \mathbf{f}(\mathbf{x}, \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}) \\ \mathbf{A}_s\mathbf{x}_s + \mathbf{B}_s\boldsymbol{\nu} \end{bmatrix}, \quad \begin{matrix} \mathbf{g}_1(\mathbf{x}, \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}, t) = \mathbf{0}, \\ \mathbf{h}(\mathbf{x}, \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}, t) \geq \mathbf{0}. \end{matrix} \tag{3.18}$$

The feedback policy obtained from this augmented system is of the form

$$\boldsymbol{\nu}(\mathbf{x}, t) = \boldsymbol{\nu}^*(t) + \begin{bmatrix} \mathbf{K}_{\boldsymbol{\nu},\mathbf{x}}(t) & \mathbf{K}_{\boldsymbol{\nu},\mathbf{x}_s}(t) \end{bmatrix} \begin{bmatrix} \mathbf{x} - \mathbf{x}^*(t) \\ \mathbf{x}_s - \mathbf{x}_s^*(t) \end{bmatrix}.$$

After optimization, the original input is retrieved by substituting this policy into the output function of the filter, $\mathbf{u} = \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}$, resulting in the complete feedback policy

$$\begin{aligned} \begin{bmatrix} \mathbf{u}(\tilde{\mathbf{x}}, t) \\ \boldsymbol{\nu}(\tilde{\mathbf{x}}, t) \end{bmatrix} = &\begin{bmatrix} \mathbf{C}_s\mathbf{x}_s^*(t) + \mathbf{D}_s\boldsymbol{\nu}^*(t) \\ \boldsymbol{\nu}^*(t) \end{bmatrix} + \\ &\begin{bmatrix} \mathbf{D}_s\mathbf{K}_{\boldsymbol{\nu},\mathbf{x}}(t) & \mathbf{D}_s\mathbf{K}_{\boldsymbol{\nu},\mathbf{x}_s}(t) + \mathbf{C}_s \\ \mathbf{K}_{\boldsymbol{\nu},\mathbf{x}}(t) & \mathbf{K}_{\boldsymbol{\nu},\mathbf{x}_s}(t) \end{bmatrix} \begin{bmatrix} \mathbf{x} - \mathbf{x}^*(t) \\ \mathbf{x}_s - \mathbf{x}_s^*(t) \end{bmatrix}. \end{aligned} \tag{3.19}$$

3.3 Implementation

We apply our approach to the kinodynamic model of a quadruped robot, which describes the dynamics of a single free-floating body along with the kinematics for each leg. The Equations of Motion (EoM) are given by

$$\begin{cases} \dot{\boldsymbol{\theta}} = \mathbf{T}(\boldsymbol{\theta})\boldsymbol{\omega}, \\ \dot{\mathbf{p}} = {}_{\mathrm{W}}\mathbf{R}_{\mathrm{B}}(\boldsymbol{\theta})\,\mathbf{v}, \\ \dot{\boldsymbol{\omega}} = \mathbf{I}^{-1}\left(-\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} + \sum_{i=1}^{4} \mathbf{r}_{\mathrm{EE}j}(\mathbf{q}) \times \boldsymbol{\lambda}_{\mathrm{EE}j}\right), \\ \dot{\mathbf{v}} = \mathbf{g}(\boldsymbol{\theta}) + \frac{1}{m}\sum_{i=1}^{4} \boldsymbol{\lambda}_{\mathrm{EE}j}, \\ \dot{\mathbf{q}} = \mathbf{u}_J, \end{cases}$$

where ${}_{\mathrm{W}}\mathbf{R}_{\mathrm{B}}$ and $\mathbf{T}$ are the rotation matrix of the base with respect the global frame and the transformation matrix from angular velocities in the base frame to the Euler angles derivatives in the global frame. The term $\mathbf{g}$ is the gravitational acceleration in body frame, $\mathbf{I}$ and m are the moment of inertia about the Center of Mass (CoM) and the total mass respectively. The inertia is assumed to be constant and taken at the default configuration of the robot. $\mathbf{r}_{\mathrm{EE}j}$ is the position of the foot j with respect to CoM. $\boldsymbol{\theta}$ is the orientation of the base in

Euler angles, $\mathbf{p}$ is the position of the CoM in world frame, $\boldsymbol{\omega}$ is the angular rate, and $\mathbf{v}$ is the linear velocity of the CoM. $\mathbf{q}$ is the vector of twelve joint positions. The inputs of the model are the joint velocity commands $\mathbf{u}_J$ and end-effector contact forces $\boldsymbol{\lambda}_{\mathrm{EE}j}$ in the body frame.

3.3.1 Equality Constraints

The equality constraints depend on the mode of each leg at a certain point in time. We assume that the mode sequence is a predefined function of time. The resulting mode-depended constraints are

$$\begin{cases} \mathbf{v}_{\mathrm{EE}j} = \mathbf{0}, & \text{if } i \text{ is a stance leg,} \\ \mathbf{v}_{\mathrm{EE}j} \cdot \hat{\mathbf{n}} = c(t), \quad \boldsymbol{\lambda}_{\mathrm{EE}j} = \mathbf{0}, & \text{if } i \text{ is a swing leg,} \end{cases}$$

where $\mathbf{v}_{\mathrm{EE}j}$ is the end-effector velocity in world frame. These constraints ensure that a stance leg remains on the ground and a swing leg follows the predefined curve $c(t)$ in the direction of the local surface normal $\hat{\mathbf{n}}$ to avoid foot scuffing. Furthermore, the constraints enforce zero contact force at swing legs.

3.3.2 Inequality Constraints

Our proposed relaxed barrier method allows to model the friction cone without the commonly used polytope approximation. The cone constraint for each end-effector,

$$\boldsymbol{\lambda}_{\mathrm{EE}j} \in \mathcal{C}(\hat{\mathbf{n}}, \mu_c), \tag{3.20}$$

is defined by the surface normal and friction coefficient $\mu_c = 0.7$. After projecting the contact forces to the local frame of the surface, a canonical second-order cone constraint is found in terms of local contact forces $\mathbf{F} = [F_x, F_y, F_z]$. An effective cone constraint used in conjunction with barrier methods (Lobo et al., 1998) is given by

$$h_{\mathrm{cone}} = \mu_c F_z - \sqrt{F_x^2 + F_y^2} \geq 0. \tag{3.21}$$

However, the gradient of this constraint is not defined at $\mathbf{F} = 0$, which causes numerical issues close to the origin. While for interior point methods this problem can be solved by using the squared constraint, $\mu_c^2 F_z^2 - F_x^2 + F_y^2 \geq 0$, this strategy does not work well together with the relaxed barrier function due to the saddle point it introduces at the origin. Since the relaxed barrier function allows infeasible iterates, the solutions can cross the origin and end up

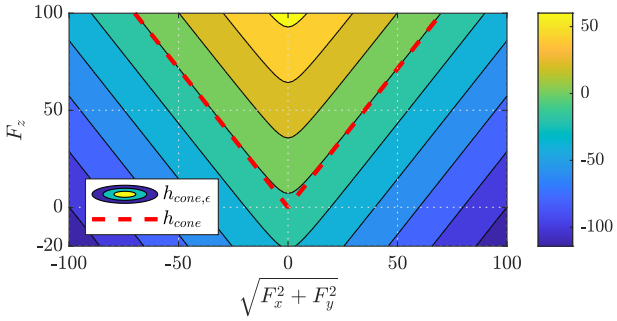


Figure 3.3: Comparison of the friction cone constraint h_{cone} and the perturbed cone $h_{\text{cone},\epsilon}$ for $\epsilon = 5$, $\mu_c = 0.7$. The contour of $h_{\text{cone},\epsilon}$ is shown together with the zero crossing of h_{cone}.

in the negative reflection of the cone, which became a feasible region through the squaring operation. We, therefore, use the perturbed cone

$$h_{\text{cone},\epsilon} = \mu_c F_z - \sqrt{F_x^2 + F_y^2 + \epsilon^2} \geq 0, \tag{3.22}$$

which is differentiable at the origin, remains infeasible for any negative F_z, and is a conservative lower bound for the original cone (3.21). It therefore holds that

$$h_{\text{cone},\epsilon} \geq 0 \implies h_{\text{cone}} \geq 0. \tag{3.23}$$

In Fig. 3.3 the level sets of this constraint are compared to the original cone. It can be seen that the constraint is convex and the zero crossing of $h_{\text{cone},\epsilon}$ is strictly inside the feasible region.

3.3.3 Torque Computation

The control inputs $\mathbf{u}$ consist of contact forces and joint velocities. These commands have to be translated to torques. When using only the feedforward trajectories, desired accelerations and contact forces are extracted and tracked by a hierarchical inverse dynamics controller (Dario Bellicoso et al., 2016). When using the feedback policy, we forward simulate the system under the feedback policy for a short time and extract desired accelerations from this rollout. The inverse dynamics is then only used to convert the desired accelerations into torques, without adding additional feedback. This inverse dynamics controller is evaluated at 400 Hz, while the Sequential Linear Quadratic Model Predic-

tive Control (SLQ-MPC) algorithm runs asynchronously on a second onboard Intel i7-4600U@2.1 GHz dual core processor.

Finally, each individual motor has a local, embedded, control loop. For the stance legs a torque controller is used, and for the swing legs the motors take the commanded torque as a feedforward term and close the loop over the desired position and joint velocities.

3.4 Results

We first show the qualitative differences between a feedback policy and a feedforward policy when operating under low update rates and disturbances. Then, the influence of the relaxed barrier function cone constraint on the planned feedback gains is shown. We also examine the structure in the obtained feedback matrices and compare the difference between those obtained with frequency shaping and those without. Finally, we show that the proposed elements, when taken together, lead to a method that can be successfully executed on the onboard hardware of a torque-controlled robot.

We use a diagonal cost on the state and control inputs for all experiments. When frequency shaping is used, we set $\alpha_i = 0.01$, $\beta_i = 0.2$ for the contact force inputs and $\alpha_i = 0.01$, $\beta_i = 0.1$ for the joint velocity inputs in (3.17).

3.4.1 Feedback MPC

We first investigate the effect of low update frequencies and the use of feedback policies from the SLQ-MPC in simulation. We introduce model errors to show the performance of the different strategies. The mass of the control model is increased by 10 % with respect to the simulation model. Each MPC controller is brought to a stable trot gait and commanded to move 1 m forward.

First, feedforward MPC is used with an update rate equal to the control frequency of 400 Hz. Every control loop thus has access to the optimal solution from the current state. The resulting desired linear accelerations are shown in the top of Fig. 3.4. Discontinuities only arise around a contact switch; the desired accelerations are continuous otherwise. In the middle plot, the desired accelerations are shown when the update rate is restricted to an update frequency of 20 Hz. The updates are clearly visible because every time the feedforward trajectory is updated, the accumulated deviation from the feedforward plan is reset and a new open loop trajectory is tracked. In the bottom plot, we show the performance when the SLQ feedback policy is used. The

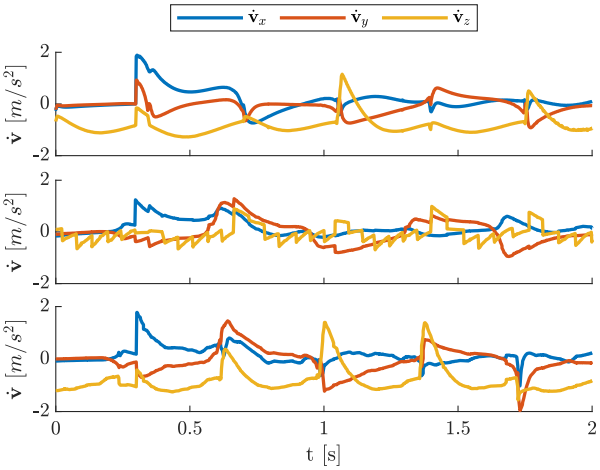


Figure 3.4: Desired linear acceleration for the center of mass, as commanded by the MPC. In the top two plots a feedforward MPC strategy is used with update rates of 400 Hz and 20 Hz respectively. In the bottom plot, feedback MPC is used with an update rate of 20 Hz. The trajectories are for a trotting gait with a command to move 1 m ahead send at 0.3 s

discontinuities at updates are significantly reduced and smooth trajectories comparable to MPC with high update rate are retrieved.

These experiments show that using the feedback from the MPC can recover some of the performance lost due to lower update rates. By using a policy that is consistent with future MPC updates, the discontinuities at the updates are reduced up to the validity of the linear quadratic approximation.

3.4.2 Feedback Gains Near Inequality Constraints

In the following experiment we prescribe a task where we require to lift the left front leg and simultaneously set the desired body location towards the front left, outside of the support polygon. This task requires the algorithm to coordinate the step with the body movement. The experiment is performed without the frequency shaped cost to allow us to focus on the effect of using the inequality constraints. In the left side of Fig. 3.5 we show the optimized solution without inequality constraints. Without the cone constraint, the optimal strategy is to produce negative contact forces in the right hind leg, such that the desired body position can be reached as early as possible. Furthermore,

we plot the maximum gain in the row associated with each vertical contact force, which shows that the zero force feedback terms for the swing leg are compliant with the corresponding equality constraint.

The resulting solution when adding inequality constraints is shown on the right of Fig. 3.5. Here, we used fixed barrier parameters $\mu = 0.5$, $\delta = 0.1$, under which all constraints are strictly satisfied. With the inequality constraints, the contact forces remain positive on the right hind leg. Additionally, where before the feedback gains are about equal for the three remaining stance legs, now, the feedback gains for the right hind leg are significantly reduced. When the contact force approaches zero, the feedback gains go to zero as well, which ensures that the inequality constraint is not violated after applying the feedback policy at a disturbed state. This interaction between the inequality constraints and the feedback gains is a result of the barrier function. As the constraint boundary is approached, the Hessian of the barrier function w.r.t. the contact forces increases. This increases the input costs $\mathbf{R}$ in (3.15), and thus reduces the feedback gains.

This gradual decrease in feedback gains cannot be obtained with a clamping strategy, where the gains are unaffected, or with an active set method, where the gains instantaneously decrease to zero when the constraint becomes active (Tassa, Mansard, and Todorov, 2014).

As a secondary effect, we see that the feedback gains on the left front leg increase before lifting the leg. Because the right hind leg is forced to have low feedback gains in the upcoming phase, the body position before lifting the foot is of high importance, which is reflected in the gains.

In addition, we note that the distance to the constraint boundary can be regulated by choosing a different barrier scaling μ. This means that using a finite μ, in contrast to decreasing it to zero as done in an interior point method, can be used to trade some optimality for a larger stability margin.

3.4.3 Feedback Structure

We visualize the feedback matrix for ANYmal in a full stance configuration. The gains obtained without using the frequency-dependent cost function are shown in Fig. 3.6a. The state acting on each column of the matrix is shown above the figure, and the control input affected by each row is shown on the left. The color intensity shows the magnitude of each entry in the feedback matrix, with zero shown as white and the highest gain shown in black.

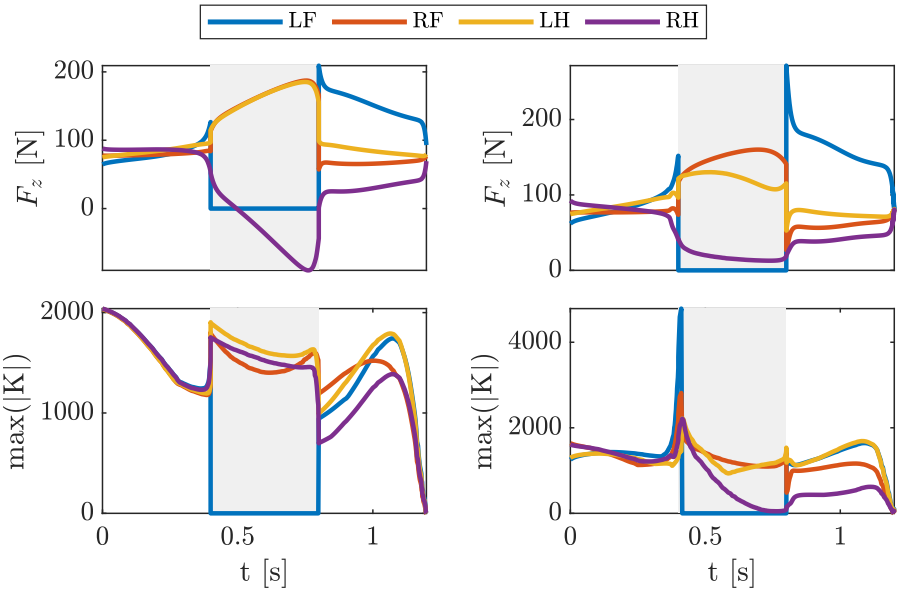


Figure 3.5: Optimal vertical contact force trajectory and maximum feedback gain associated with that contact force when planning a simultaneous step and reach task without (left), and with (right) inequality constraints. The time where the left front (LF) is in the air is marked in gray.

In the joint velocity part of the feedback matrix, one can see how the equality constraints that require zero velocity at the end-effectors are reflected: The joint velocity commands are highly dependent on the linear and angular velocity of the base to achieve this constraint. This empirically verifies that (3.15) indeed produces feedback matrices that are consistent with the constraints.

The feedback matrix in Fig 3.6a can be compared to the feedback matrix obtained when using the frequency shaped cost function, shown in Fig 3.6b. We split the feedback matrix in Fig 3.6b into four parts, with the vertical split between system and filter state, and horizontal split between system and auxiliary input, corresponding to the partitioning in (3.19). First, the left side is inspected. Here we recognize a feedback pattern similar to that in Fig 3.6a, obtained without the frequency-dependent cost function. Indeed since the shaping functions in (3.17) have unit DC-gain, the feedback matrix $K_{\boldsymbol{\nu},\mathbf{x}}$ is expected to be approximately equal to the matrix obtained without frequency

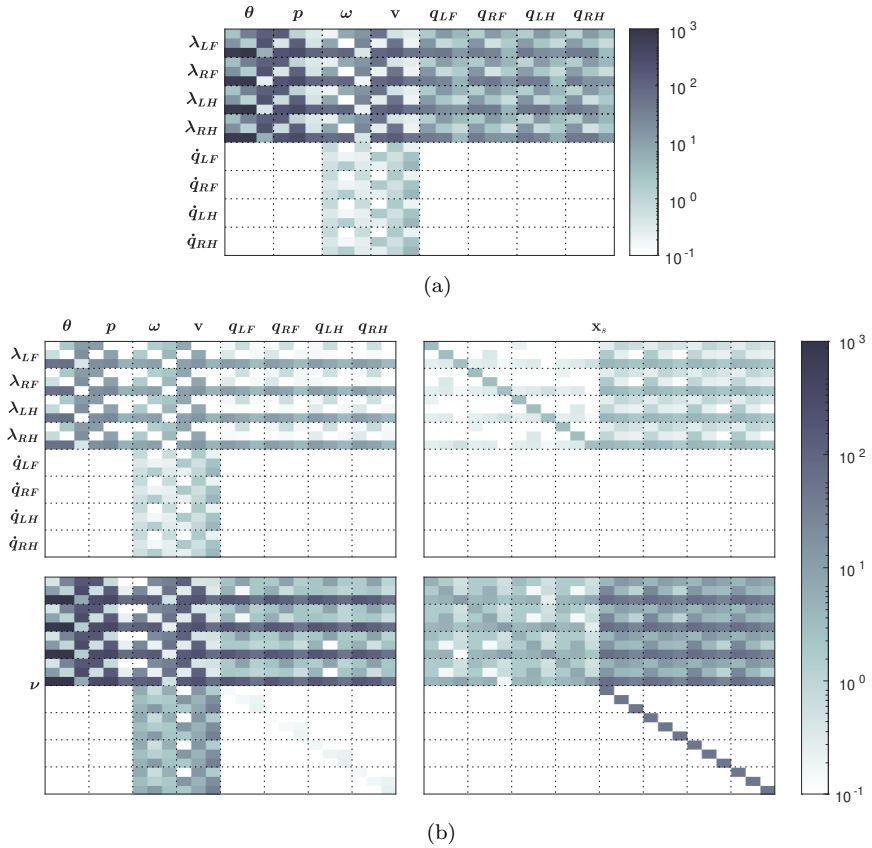


Figure 3.6: The feedback matrix in stance configuration without (a) and with (b) frequency shaped cost. The color of each state-input entry represents the magnitude of the feedback. In (b), the matrix is split into four parts with input and states associated with each block marked at the left and bottom. The top left for example shows the gains between system states, $\mathbf{x}$, and system inputs.

shaping. Furthermore, we can recognize that the direct gains from state to contact forces are an order of magnitude smaller than before.

For the frequency shaped feedback policy, the main feedback flows from system state to auxiliary inputs. The auxiliary inputs then drive the filter states, $\mathbf{x}_s$, which in turn provide the adaptation of the system contact force inputs.

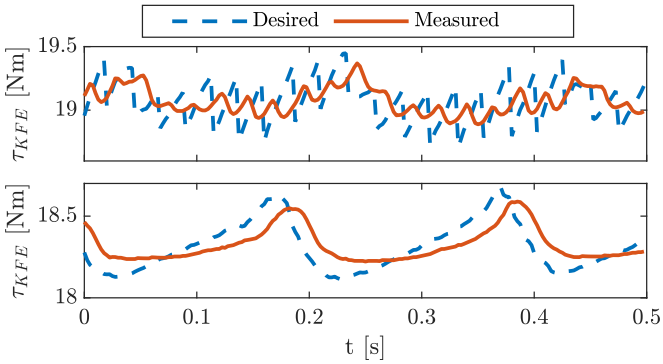


Figure 3.7: Desired and measured torque signals in the left front knee under a constant disturbance of 5.7 kg. The desired torque is the result of both feedforward and feedback signals. Where in the top plot the feedback is provided by the conventional tracking controller, in the bottom plot the feedback from the frequency shaped SLQ algorithm is used.

For the joint velocities, however, the feedback from filter states to joint velocities is zero, as expected. The fact that end-effector velocities are constrained to be zero is still reflected in the feedback matrix in exactly the same way as before. Feedback terms therefore appear in the bottom right partition of Fig 3.6b, to satisfy the equality constraint, $\mathbf{u} = \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\boldsymbol{\nu}$, for the rows associated with joint velocities.

This analysis shows that using the frequency-dependent cost function introduces smoothness and reduced direct gains where possible, but at the same time still respects hard equality constraints on the original system inputs.

3.4.4 Hardware Experiments: Disturbance Rejection

As seen in the accompanying video[1], when LQR gains obtained from the SLQ-MPC are used on hardware, the system becomes unstable even in full stance phase. As demonstrated in Section 3.4.3, the frequency shaped formulation reduces the direct gains between state and input, which enables successful deployment on hardware. All hardware experiments are therefore performed with both frequency shaping and inequality constraints active.

We first perform a simple experiment to verify the qualitative difference observed in simulation between using only a feedforward policy with a conven-

[1] A video of the experiments is available at `https://youtu.be/KrTrLGDA6FQ`

tional tracking controller or when using a feedback policy. The robot is put in a standing configuration with the desired position set to the initial position with zero velocity. Afterward, we place a 5.7 kg mass ($\approx$ 15 % of the total mass) on top of the robot to induce a constant disturbance. Fig. 3.7 shows the resulting desired and measured torque trajectories in the left front knee after the system reaches an equilibrium.

In the top plot, the points at which the feedforward policy is updated are clearly visible. At each update, the state reference is reset to the measured reference, which effectively nullifies the feedback of the tracking controller. Since the feedforward control signal does not account for the additional disturbance, the system deviates from the desired trajectory and builds up feedback in the tracking controller until the next update arrives. In the bottom plot, where the feedback policy updates arrive at the same rate as the feedforward case, there are no discontinuous jumps in desired torque. This verifies our earlier observation in simulation.

3.4.5 Hardware Experiments: Dynamic Walking

Finally, we demonstrate that the proposed method achieves stable walking with all computations running on the onboard computers. We use a gait known as *dynamic walk*. The gait pattern is shown in Fig. 3.8. It consists of a mixture of underactuated and overactuated contact configurations when two and three feet are on the ground, respectively. The proposed friction cone constraint ensures that the trajectory does not require negative contact forces and thus successfully navigates the intriguing pattern of support polygons.

A receding horizon of 1.0 s was used, for which the MPC reaches an update frequency of approximately 15 Hz. The resulting desired and measured torque and joint velocity trajectories are shown in Fig. 3.9 for the left front leg. The desired and measured signals are close to each other at all time, showing that the applied feedback policy respects the bandwidth limits of the actuators. More dynamic gaits such as pace and trot motions are feasible as well. In the linked video a continuous transition between the two gaits is shown. The feedback MPC is able to skilfully coordinate leg and body motions in these unstable and underactuated situations.

3.5 Conclusion

In this work, we proposed to use feedback MPC as an effective way to handle the slow update rate associated with the computational restrictions of mobile

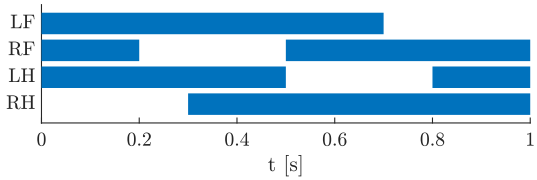


Figure 3.8: Gait pattern for the dynamic walk used in the hardware experiments. Colored areas represent that a leg is in contact.

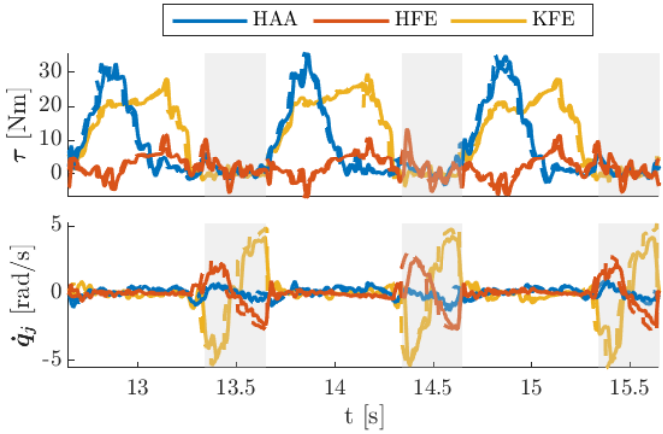


Figure 3.9: Torque and joint velocity trajectories during three cycles of a dynamic walk with ANYmal. The desired (dashed line) and measured (full line) are shown for the HAA (Hip Abduction Adduction), HFE (Hip Flexion Extension), and KFE (Knee Flexion Extension).

platforms. The sensitivity of CoM control to uncertainties and sampling period has been recently analyzed in (Villa, Englsberger, and Wieber, 2019), which provides a theoretical basis for our observation that stable walking is possible with low update rates.

We proposed a relaxed barrier function method to extend the SLQ algorithm to optimization problems with inequality constraints. In particular, the friction cone is implemented through a perturbed second-order cone constraint. This formulation adds a convex penalty to the cost function and avoids numerical ill-conditioning at the origin of the cone.

A frequency-aware MPC approach was used to systematically include the bandwidth limit of the actuators in the feedback policy design. This was a key factor to achieve closed-loop stability on hardware without any detuning of the low-level actuator controllers as suggested in (Mason et al., 2016). The frequency-aware approach effectively allows to set high gains in the low-frequency spectrum and to attenuate gains in high frequency. It thus increases the robustness of the feedback policy in the presence of high-frequency disturbances. We showed that the feedback policy is consistent with the constraints of the locomotion task. We empirically confirmed that the MPC policy reduces the feedback gains near the boundaries of the friction cone to respect the inequality constraints. We also demonstrated that the optimized policy sets zero gains on the contact force of the swing legs and encodes the zero end-effector velocity constraint for stance legs to satisfy state-input equality constraints.

4

Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions

Grandia, R., Taylor, A. J., Singletary, A., Hutter, M., and Ames, A. D. (2020). "Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions". In: *Proceedings of Robotics: Science and Systems*. Corvalis, Oregon, USA

DOI: 10.15607/RSS.2020.XVI.098

Video: https://youtu.be/weNv-FIRKiE

The theoretical unification of Nonlinear Model Predictive Control (NMPC) with Control Lyapunov Functions (CLFs) provides a framework for achieving optimal control performance while ensuring stability guarantees. In this paper we present the first real-time realization of a unified NMPC and CLF controller on a robotic system with limited computational resources. These limitations motivate a set of approaches for efficiently incorporating CLF stability constraints into a general NMPC formulation. We evaluate the performance of the proposed methods compared to baseline CLF and NMPC controllers with a robotic Segway platform both in simulation and on hardware. The addition of a prediction horizon provides a performance advantage over CLF based controllers, which operate optimally point-wise in time. Moreover, the explicitly imposed stability constraints remove the need for difficult cost function and parameter tuning required by NMPC. Therefore the unified controller improves the performance of each isolated controller and simplifies the overall design process.

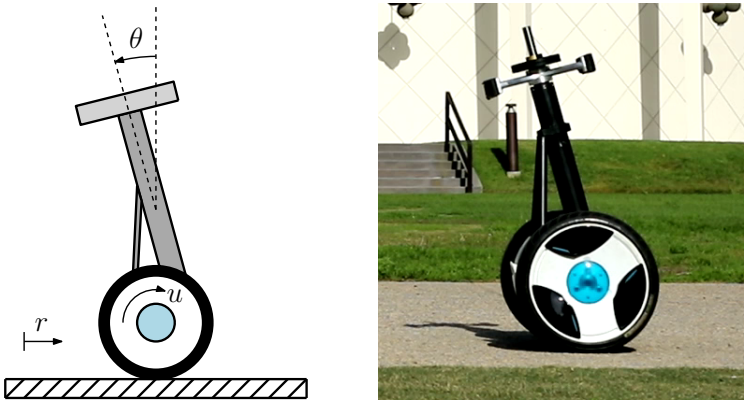


Figure 4.1: Left: Segway model for simulation and control design. Right: Physical Segway system in outdoor experiment environment.

4.1 Introduction

Deploying autonomous and versatile robots into the real world comes with the challenge of ever increasing complexity in sensing, decision making, and actuation. One difficulty in designing controllers for such complex systems lies in the need to simultaneously meet a large set of design requirements. Achieving stable and safe behavior is often in conflict with performance objectives, and finding the right balance between these requirements can be a challenging task.

A disjoint, hierarchical approach is typically taken in this context. High-level trajectories are planned to satisfy performance objectives via computationally intensive nonlinear optimization, and local feedback controllers are separately designed to ensure stability. The goal of this paper is to directly integrate these two components by constructing controllers that simultaneously optimize performance along a horizon and satisfy local stability constraints in a computationally efficient manner. In particular, we seek to unify guarantee-based methods of nonlinear control with the optimization-based view in model predictive control. To this end, we combine the guarantees endowed by CLFs (Artstein, 1983; Sontag, 1989a) with the optimal performance of NMPC (Bock and Plitt, 1984; Mayne et al., 2000; Rawlings, Mayne, and Diehl, 2017).

Lyapunov methods are a powerful tool for certifying stability properties of nonlinear systems (Khalil, 2002). The use of CLFs to synthesize stabilizing controllers for robotic platforms has become increasingly popular (Galloway et al., 2015; Ma et al., 2017; Nguyen and Sreenath, 2015), often via Quadratic

Programs (QPs) (Ames et al., 2014; Ames and Powell, 2013). Despite the optimization-based formulation of these controllers, they often fail to achieve long-term optimal behavior. This deficiency arises due to the fact that the cost of these optimization problems fails to incorporate the future behavior of the system, but is instead point-wise optimal (Freeman and Kokotović, 1996).

In contrast, NMPC emphasizes performance by solving a finite horizon optimal control problem online and applying the first element of the computed open-loop input trajectory to the system (Rawlings, Mayne, and Diehl, 2017). This optimization is repeatedly solved with each newly measured state to obtain the next control input trajectory. While this class of control laws can achieve strong performance in practice (Garcia, Prett, and Morari, 1989; Grüne and Pannek, 2017; Qin and Badgwell, 2003), and allows intuitive specification of the desired behaviour, additional assumptions must be met to certify closed-loop stability. In classical discrete-time NMPC, stability is guaranteed by an appropriately designed terminal penalty and terminal constraint (Grüne and Pannek, 2017; H. Chen and F. Allgöwer, 1998; Mayne et al., 2000).

The integration of Lyapunov methods with NMPC is not a new idea. Lyapunov methods have been used to construct stabilizing terminal conditions (Jadbabaie, Yu, and Hauser, 2001), or to analyze stability in the absence thereof (Jadbabaie and Hauser, 2005). Another approach incorporates the stability condition required by a CLF along the prediction horizon found with NMPC (Primbs, Nevistic, and Doyle, 2000; Yu et al., 2001). As noted in (Primbs, Nevistic, and Doyle, 2000), this approach has several desirable properties such as the absence of a terminal cost, stability for any horizon length, and recovery of the CLF-QP (Ames et al., 2014) or infinite horizon optimal controllers when considering the limiting behavior of the finite horizon. The idea of imposing stability constraints along the horizon has appeared in other forms such as contractive state constraints (de Oliveira Kothare and Morari, 2000), and has been applied within the context of chemical process control (Mahmood and Mhaskar, 2014; Wu et al., 2018), economic cost functions (Heidarinejad, Liu, and Christofides, 2012), and switched nonlinear systems (Mhaskar, El-Farra, and Christofides, 2005).

While this existing work has analyzed the stability and optimality properties obtained through the unification of CLFs and NMPC, there has been little attention to the practical and computational aspects of the resulting nonlinear optimization problem. Limited computational resources and fast system dynamics present a challenge to the deployment of these unified methods to

modern robotic systems. Indeed, to the best of our knowledge, such a control scheme has not yet been applied to robotic systems experimentally.

To achieve the goal of experimental realizability, we develop a new methodology for combining CLFs with NMPC. We describe practical methods to efficiently solve the resulting nonlinear optimization problems and ultimately realize the proposed controllers in simulation and, for the first time, experimentally on a robotic platform; in this case, a Segway hardware platform seen in Fig. 4.1. While each proposed method provides theoretical stability guarantees, significant differences in computational efficiency and performance are observed. Furthermore, we find that the pairing of these control methodologies leads to improved performance over CLF methods and significantly reduced tuning of prediction horizon length and terminal conditions for NMPC methods.

Our paper is organized as follows. Section 4.2 provides a review of CLFs and the stability guarantees they yield, and reviews the NMPC problem and how it is solved in practice. In Section 4.3 we propose a set of methods for incorporating CLF stability constraints into the NMPC problem, and provide additional details on implementation. Lastly, in Section 4.4 we provide results from both simulation and hardware that demonstrate the ability of this unified control approach to achieve stability and improve performance.

4.2 Background

In this section we provide background information on CLFs and NMPC. This information supports the specific framework unifying CLFs and NMPC in Section 4.3.

4.2.1 Control Lyapunov Functions

Consider a state space $\mathcal{X} \subset \mathbb{R}^n$ and a control input space $\mathcal{U} \subset \mathbb{R}^m$, where it is assumed $\mathcal{X}$ is path-connected and $\mathbf{0} \in \mathcal{X}$. Consider the control-affine dynamic system given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}. \tag{4.1}$$

where $\mathbf{x} \in \mathcal{X}$, $\mathbf{u} \in \mathcal{U}$, and $\mathbf{f} : \mathcal{X} \to \mathbb{R}^n$ and $\mathbf{g} : \mathcal{X} \to \mathbb{R}^{n \times m}$ are Lipschitz continuous on $\mathcal{X}$. Further assume that $\mathbf{f}(\mathbf{0}) = \mathbf{0}$, or that the origin is an equilibrium point of the system. As in (Khalil, 2002), we define a class $\mathcal{K}$ function as a continuous function $\alpha : [0, a) \to \mathbb{R}_+$, with $a > 0$, $\alpha(0) = 0$ and α strictly monotonically increasing (denoted $\alpha \in \mathcal{K}$). If $a = \infty$ and

$\lim_{r\to\infty}\alpha(r)=\infty$, then α is said to be a *class* $\mathcal{K}_\infty$ *function* ($\alpha\in\mathcal{K}_\infty$). This type of function can be interpreted as a type of nonlinear gain function, noting the linear gain function $\alpha(r)=kr$ with $k>0$ satisfies this definition. Given this definition, we define CLFs as in (Artstein, 1983; Lin and Sontag, 1991).

Definition 1 (*Control Lyapunov Functions)***.** A continuously differentiable function $V:\mathcal{X}\to\mathbb{R}_+$ is a *Control Lyapunov Function (CLF)* for (4.1) on $\mathcal{X}$ if there exists $\alpha_1,\alpha_2,\alpha_3\in\mathcal{K}_\infty$ such that for all $\mathbf{x}\in\mathcal{X}$:

$$\alpha_1(\|\mathbf{x}\|)\leq V(\mathbf{x})\leq\alpha_2(\|\mathbf{x}\|) \tag{4.2}$$

$$\inf_{\mathbf{u}\in\mathcal{U}}\dot{V}(\mathbf{x},\mathbf{u})\leq-\alpha_3(\|\mathbf{x}\|), \tag{4.3}$$

This definition can be restated with $\alpha_1,\alpha_2,\alpha_3\in\mathcal{K}$ with resulting stability guarantees holding locally. The existence of a CLF for (4.1) implies the existence of a continuous (except possibly at $\mathbf{x}=\mathbf{0}$) state-feedback controller $\mathbf{k}:\mathcal{X}\to\mathcal{U}$ that renders the origin globally asymptotically stable (Artstein, 1983; Sontag, 1989b). It is possible to make $\mathbf{k}$ continuous at $\mathbf{x}=\mathbf{0}$ if V satisfies the small-control property (Sontag, 1989a). If the functions $\alpha_1,\alpha_2,\alpha_3$ take the form $\alpha_i(r)=c_ir^2$, $i=1,2,3$, the resulting stability is global exponential stability, with the magnitude of the state upper bounded by a function exponentially decaying in time:

$$\|\mathbf{x}(t)\|\leq M\|\mathbf{x}(0)\|e^{-\gamma t} \tag{4.4}$$

with $M,\gamma>0$. Similarly, the CLF can be upper bounded:

$$V(\mathbf{x}(t))\leq V(\mathbf{x}(0))e^{-\gamma t} \tag{4.5}$$

This preceding bound will be useful for enforcing Lyapunov stability guarantees within the discrete time NMPC problem.

The CLF definition implies the existence of a point-wise set of stabilizing control inputs:

$$\mathcal{U}_{\text{clf}}(\mathbf{x})=\{\mathbf{u}\in\mathcal{U}\mid\dot{V}(\mathbf{x},\mathbf{u})\leq-\alpha_3(\|\mathbf{x}\|)\}. \tag{4.6}$$

Thus a CLF characterizes a stabilizing feedback controller as a controller $\mathbf{k}:\mathcal{X}\to\mathcal{U}$ such that $\mathbf{k}(\mathbf{x})\in\mathcal{U}_{\text{clf}}(\mathbf{x})$ for all $\mathbf{x}\in\mathcal{X}$. Furthermore, upon selection of such a controller, the CLF is a certificate of stability for the closed loop system:

$$\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x})+\mathbf{g}(\mathbf{x})\mathbf{k}(\mathbf{x}). \tag{4.7}$$

Establishing that a given function V serves as a CLF for (4.1) is often done for robotic systems constructively by specifying a controller taking values in $\mathcal{U}_{\text{clf}}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ (Kolathaya and Veer, 2019; Taylor et al., 2019). Note that for any $\mathbf{x} \in \mathcal{X}$ the set $\mathcal{U}_{\text{clf}}(\mathbf{x})$ is described by an affine inequality in $\mathbf{u}$ due to the affine nature of the dynamics:

$$\dot{V}(\mathbf{x}, \mathbf{u}) = \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x}) \left(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} \right). \tag{4.8}$$

Due to this, the CLF itself may then be used to synthesize a optimization-based controller with more desirable properties using quadratic programs (Ames et al., 2014; Ames and Powell, 2013; Galloway et al., 2015). Specifically, we obtain a feedback control law $\mathbf{k}(\mathbf{x})$ that satisfies the inequality (4.3):

$$\begin{aligned} \mathbf{k}(\mathbf{x}) = \ & \underset{\mathbf{u} \in \mathcal{U}}{\operatorname{argmin}} \quad \frac{1}{2}\mathbf{u}^\top \mathbf{R}\mathbf{u} + \mathbf{q}^\top \mathbf{u} \\ & \text{s.t.} \qquad \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x}) \left(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} \right) \leq -\alpha_3(\|\mathbf{x}\|)) \end{aligned} \tag{4.9}$$

where $\mathbf{R}$ is positive definite and $\mathcal{U}$ assumed to be a polytope. Feasibility of this optimization problem is guaranteed by the satisfaction of the constraint (4.3) and Lipschitz continuity of this controller has been studied in (Jankovic, 2018; Morris, Powell, and Ames, 2015).

This controller is point-wise optimal (Freeman and Kokotović, 1996), and takes a greedy approach to specifying control inputs. This often leads to poor performance compared to even non-optimization based controllers as there is no consideration of the future behavior of the system when the input is chosen. In addition to challenges in achieving longer horizon optimality, these controllers face difficulty in implementation on robotic platforms. The stability guarantees endowed by these controllers assume a continuous-time implementation, which is not possible on many modern digital control systems. Instead, control inputs are chosen and held for a small interval of time in a zero-order-hold manner. Lyapunov stability of zero-order-hold systems has been studied utilizing an approximate discretization of the nonlinear dynamics (Nešić and Teel, 2004; Nešić, Teel, and Sontag, 1999), or in the context of model predictive control (Grüne, Nešić, and Pannek, 2007; Gyurkovics and Elaiw, 2004; Mhaskr, El-Farra, and Christofides, 2006; Nešić and Grüne, 2006).

4.2.2 Nonlinear Model Predictive Control

NMPC offers an alternative to CLF-based methods for controlling nonlinear systems, and is inherently designed to resolve the challenges of longer horizon optimality at the expensive of higher online computational cost.

We consider a *direct* NMPC approach to transform the continuous optimal control problem into a finite dimensional Nonlinear Program (NLP) (Bock and Plitt, 1984). The continuous control signal $\mathbf{u}(t)$ is parameterized over subintervals of the prediction horizon $[0, T]$ to obtain a finite dimensional decision problem. This creates a fixed grid of nodes $k \in \{0, \dots, N\}$ defining control times t_k separated by intervals of duration $\delta t = T/(N-1)$. In this work, we consider a piecewise constant, or zero-order-hold, parameterization of the input. Denoting $\mathbf{x}_k = \mathbf{x}(t_k)$ and integrating the continuous dynamics in (4.1) over an interval leads to a discrete time representation of the dynamics:

$$\mathbf{x}_{k+1} = \mathbf{f}^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{x}_k + \int_{t_k}^{t_k+\delta t} \mathbf{f}(\mathbf{x}(\tau)) + \mathbf{g}(\mathbf{x}(\tau))\mathbf{u}_k \, \mathrm{d}\tau. \tag{4.10}$$

The integral in (4.10) is numerically approximated with an integration method of choice to achieve a desired approximation accuracy of the evolution of the continuous time system under the zero-order-hold commands.

The general NMPC problem presented below can be formulated by defining and evaluating a cost function and constraints on the grid of nodes. Here, we write the problem in parametric form, depending on the current measured state $\hat{\mathbf{x}}$ and additional parameters contained in $\mathbf{p}$, and with a subset of the constraints implemented as soft-constraints with slack terms:

$$\min_{\mathbf{X},\mathbf{U},\mathbf{S}} \quad l_N(\mathbf{x}_N, \mathbf{p}) + \phi(\mathbf{s}_N) + \sum_{k=0}^{N-1} l_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}) + \phi(\mathbf{s}_k) \tag{4.11a}$$

$$\text{s.t.} \quad \mathbf{x}_0 - \hat{\mathbf{x}} = \mathbf{0}, \tag{4.11b}$$

$$\mathbf{x}_{k+1} - \mathbf{f}_k^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, \qquad k = 0, \dots, N-1, \tag{4.11c}$$

$$\mathbf{h}_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}) \leq \mathbf{s}_k, \qquad k = 0, \dots, N-1, \tag{4.11d}$$

$$\mathbf{h}_N(\mathbf{x}_N, \mathbf{p}) \leq \mathbf{s}_N, \tag{4.11e}$$

$$\mathbf{s}_k \geq \mathbf{0}, \qquad k = 0, \dots, N, \tag{4.11f}$$

where $\mathbf{X} = [\mathbf{x}_0^\top, \dots \mathbf{x}_N^\top]^\top$, $\mathbf{U} = [\mathbf{u}_0^\top, \dots \mathbf{u}_{N-1}^\top]^\top$, and $\mathbf{S} = [\mathbf{s}_0^\top, \dots \mathbf{s}_N^\top]^\top$ are the sequences of state, input, and slack variables respectively. The nonlinear cost and constraint functions l_k, h_k, l_N, and h_N, are allowed to vary depending

on the node index k and are dependent on problem specific parameters $\mathbf{p}$ and the current measured state $\hat{\mathbf{x}}$. The slack variables are penalized with $\phi(\mathbf{s}) = \mathbf{z}^\top \mathbf{s} + \frac{1}{2}\mathbf{s}^\top \mathbf{Z}\mathbf{s}$, an exact $\ell_1 - \ell_2$ penalty (Scokaert and Rawlings, 1999). Collecting all decision variables into a vector, $\mathbf{w} = [\mathbf{X}^\top, \mathbf{U}^\top, \mathbf{S}^\top]^\top$, problem (4.11) can be framed as a general NLP:

$$\min_{\mathbf{w}} \quad F(\mathbf{w}, \mathbf{p}) \quad \text{s.t.} \quad \begin{cases} \mathbf{G}(\mathbf{w}, \mathbf{p}) & = \mathbf{0}, \\ \mathbf{H}(\mathbf{w}, \mathbf{p}) & \leq \mathbf{0}. \end{cases} \tag{4.12}$$

4.2.3 Sequential Quadratic Programming (SQP)

Interior-Point (IP) methods and Sequential Quadratic Programming (SQP) are two popular families of algorithms for solving general NLPs (Nocedal and Wright, 2006). Additionally, the sparsity of (4.12) induced by the underlying structure of (4.11) can be exploited to obtain solutions in real-time and at a high sampling rate, which is necessary in many dynamic robotic applications. An overview of recent advances in sparsity exploiting algorithms and software tools is provided in (Kouzoupis et al., 2018).

SQP approaches offer a distinct advantage in that successive problem instances may be warm-started with solutions from preceding instances. This serves to further decrease computation time as it is often only feasible to take a single SQP step per control iteration (Li and Biegler, 1989). As NMPC computes optimal control inputs over a horizon, successive instances of (4.11) are similar and portions of the preceding optimal control sequence can be use to warm-start the following iteration, enabling convergence across multiple iterations of the problem, rather than iterating until convergence on one instance of the problem (M. Diehl and H.G. Bock and J. P. Schlöder and R. Findeisen and Z. Nagy and F. Allgöwer, 2002).

SQP based methods apply Newton-type iterations to Karush-Kuhn-Tucker (KKT) optimality conditions for (4.12), assuming some regularity conditions on the constraints (Mangasarian and Fromovitz, 1967). The Lagrangian of the NLP in (4.12) is defined as:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \mathbf{p}) = F(\mathbf{w}, \mathbf{p}) + \boldsymbol{\lambda}^\top \mathbf{G}(\mathbf{w}, \mathbf{p}) + \boldsymbol{\mu}^\top \mathbf{H}(\mathbf{w}, \mathbf{p}), \tag{4.13}$$

with Lagrange multipliers $\boldsymbol{\lambda}$, and $\boldsymbol{\mu} \geq \mathbf{0}$ corresponding to equality and inequality constraints, respectively. The Newton iterations can be equivalently

Algorithm 1 Sequential Quadratic Programming (SQP)

Given $\mathbf{p}, \mathbf{w}_0, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, F, \mathbf{G}, \mathbf{H}$
Initialize $(i, \mathbf{w}_i, \boldsymbol{\lambda}_i, \boldsymbol{\mu}_i) \leftarrow (0, \mathbf{w}_0, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0)$
while NotConverged$(\mathbf{w}_i, \boldsymbol{\lambda}_i, \boldsymbol{\mu}_i)$ **do**
  compute $\nabla_{\mathbf{w}} F(\mathbf{w}_i, \mathbf{p}), \mathbf{B}_i, \mathbf{H}(\mathbf{w}_i, \mathbf{p}), \nabla_{\mathbf{w}} \mathbf{H}(\mathbf{w}_i, \mathbf{p}), \mathbf{G}(\mathbf{w}_i, \mathbf{p}), \nabla_{\mathbf{w}} \mathbf{G}(\mathbf{w}_i, \mathbf{p})$.
  $(\delta \mathbf{w}_i, \boldsymbol{\lambda}_i^{\mathrm{QP}}, \boldsymbol{\mu}_i^{\mathrm{QP}}) \leftarrow$ Solve (4.14)
  $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \delta \mathbf{w}_i$
  $\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i^{\mathrm{QP}}$
  $\boldsymbol{\mu}_{i+1} \leftarrow \boldsymbol{\mu}_i^{\mathrm{QP}}$
  $i \leftarrow i + 1$
end while

computed by solving the following potentially non-convex QP (Nocedal and Wright, 2006):

$$\min_{\delta \mathbf{w}} \quad \nabla_{\mathbf{w}} F(\mathbf{w}_i, \mathbf{p})^\top \delta \mathbf{w} + \frac{1}{2} \delta \mathbf{w}^\top \mathbf{B}_i \delta \mathbf{w} \tag{4.14a}$$

$$\text{s.t} \quad \mathbf{G}(\mathbf{w}_i, \mathbf{p}) + \nabla_{\mathbf{w}} \mathbf{G}(\mathbf{w}_i, \mathbf{p})^\top \delta \mathbf{w} = \mathbf{0}, \tag{4.14b}$$

$$\mathbf{H}(\mathbf{w}_i, \mathbf{p}) + \nabla_{\mathbf{w}} \mathbf{H}(\mathbf{w}_i, \mathbf{p})^\top \delta \mathbf{w} \leq \mathbf{0}, \tag{4.14c}$$

where the decision variables, $\delta \mathbf{w} = \mathbf{w} - \mathbf{w}_i$, define the update step relative to the current iteration w_i, and the Hessian $\mathbf{B}_i = \nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{w}_i, \boldsymbol{\lambda}_i, \boldsymbol{\mu}_i, \mathbf{p})$. Computing the solution to (4.14) provides a decision variable update, $\delta \mathbf{w}_i$, and updated Lagrange multipliers $\boldsymbol{\lambda}_i^{\mathrm{QP}}$ and $\boldsymbol{\mu}_i^{\mathrm{QP}}$. These iterations are ran until the variables $\mathbf{w}_i$, $\boldsymbol{\lambda}_i$, and $\boldsymbol{\mu}_i$ converge. This iterative approach is summarized in Algorithm 3.

Convergence of the SQP algorithm leads to a state and input sequence, $\mathbf{X}^\star$ and $\mathbf{U}^\star$, respectively. The first element of $\mathbf{U}^\star$, denoted as $\mathbf{u}_0$, can be applied to the system, after which the SQP algorithm is run again to determine a new control input sequence. The application of SQP as a subroutine within a NMPC feedback controller is provided in Algorithm 2.

4.3 Unifying CLFs with NMPC

In this section we explore different ways of integrating the stability based CLF-QP and performance driven NMPC controllers discussed in Section 4.2. These different methods will be evaluated experimentally in Section 4.4.

Algorithm 2 SQP - NMPC

Given $\mathbf{w}^0, \boldsymbol{\lambda}^0, \boldsymbol{\mu}^0$
Initialize $(j, \mathbf{w}^j, \boldsymbol{\lambda}^j, \boldsymbol{\mu}^j) \leftarrow (0, \mathbf{w}^0, \boldsymbol{\lambda}^0, \boldsymbol{\mu}^0)$
while ControllerIsRunning() **do**
  $\hat{\mathbf{x}} \leftarrow$ StateEstimation()
  $\mathbf{y}_{ref} \leftarrow$ Commands()
  $\mathbf{p} \leftarrow (\hat{\mathbf{x}}, \mathbf{y}_{\text{ref}})$
  $(\mathbf{w}^{j+1}, \boldsymbol{\lambda}^{j+1}, \boldsymbol{\mu}^{j+1}) \leftarrow$ Solve SQP$(\mathbf{p}, \mathbf{w}^j, \boldsymbol{\lambda}^j, \boldsymbol{\mu}^j)$
  $\mathbf{U}^\star \leftarrow$ ExtractInputSequence$(\mathbf{w}^{j+1})$
  $\mathbf{u}_0 \leftarrow$ ExtractFirstInput$(\mathbf{U}^\star)$
  $\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} \leftarrow$ ApplyInput$(\mathbf{u}_0)$
  $j \leftarrow j + 1$
end while

The NMPC framework presented in Algorithm 2 can be interpreted as a closed-loop feedback controller, $\mathbf{k}_{\text{NMPC}} : \mathcal{X} \to \mathcal{U}$. As described in Section 4.2.1, for each state $\mathbf{x} \in \mathcal{X}$, a CLF defines a point-wise set of stabilizing control inputs $\mathcal{U}_{\text{clf}}(\mathbf{x})$ given in (4.6). To inherit the stability guarantees provided by the CLF, we need to restrict the NMPC controller to these stabilizing inputs, such that $\mathbf{k}_{\text{NMPC}}(x) \in \mathcal{U}_{\text{clf}}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$.

As noted in (Primbs, Nevistic, and Doyle, 2000), if only the first input in the input sequence is applied before the input sequence is recomputed, the restriction of the NMPC controller to stabilizing inputs can be achieved by directly imposing the CLF condition only on the first input, subject to the current measured state $\hat{\mathbf{x}}$:

$$h_{CLF}(\hat{\mathbf{x}}, \mathbf{u}_0) = \frac{\partial V}{\partial \mathbf{x}}(\hat{\mathbf{x}}) \left(\mathbf{f}(\hat{\mathbf{x}}) + \mathbf{g}(\hat{\mathbf{x}})\mathbf{u}_0\right) \leq -\alpha_3(\|\hat{\mathbf{x}}\|). \tag{4.15}$$

As in the case of the controller (4.9), for a given state, this constraint is affine in the decision variable $\mathbf{u}_0$. Due to this, the SQP subroutine (4.14) contains the CLF constraint without approximation, and will therefore, in the same way as the CLF-QP, compute a stabilizing control input after solving just one QP. In the context of dynamic robotic platforms, this provides an advantage over general NMPC in that it does not require the computational cost of multiple Newton iterations to converge to a potentially stabilizing control input.

Beyond the constraint on the first input, any further modifications to the NMPC problem are done explicitly to increase performance or accommodate the discrete time implementation of control inputs. In the following, we will

discuss additional constraints that serve to increase performance beyond that of the controller given by (4.9) while achieving stability.

4.3.1 Extended Horizon Constraints

While the preceding CLF constraint (4.15) enforces the selection of a stabilizing control input, the resulting theoretical stability properties rely on the controller being applied continuously in time, as noted in Section 4.2.1. As this is not possible in practice, it desirable to incorporate additional stability constraints that enforce stable behavior when control is implemented in a zero-order-hold fashion. NMPC is an advantageous framework for these types of constraints as future states to reflect desired stability properties.

In particular, consider the bound in time on the Lyapunov function established with exponential stability in (4.5). While this bound is continuous in time, comparison principles (Khalil, 2002) may be used to formulate an analogous discrete time bound at the kth node:

$$h_{LLS}(\mathbf{x}_k, \hat{\mathbf{x}}) = V(\mathbf{x}_k) - V(\hat{\mathbf{x}})e^{-\gamma k \cdot \delta t} \leq 0. \tag{4.16}$$

The constraints given by h_{CLF} and h_{LLS} can be combined in varying ways along the length of the prediction horizon. The basic approach, denoted *CLF-0* and presented in (4.17a) to (4.17f), implements the h_{CLF} constraint only at the initial node. This is extended in the *CLF-All* approach, where the h_{CLF} constraint is enforced at each node in the horizon in (4.17g). These constraints are no longer affine since at nodes $k \geq 1$ both state $\mathbf{x}_k$ and input $\mathbf{u}_k$ are decision variables, and are non-linearly coupled in (4.15).

In the approach denoted *LLS-N*, we enforce the h_{CLF} constraint at the first node and enforce the h_{LLS} constraint at only the final node in the prediction horizon in (4.17h). A similar bound, which relies on evaluating V at the final node after the system has been simulated under a control law given by Sontag's universal formula (Sontag, 1989b), is enforced in (Primbs, Nevistic, and Doyle, 2000). Our bound differs in that it is controller independent and only relies on the bound from exponential stability. Lastly, in the approach denoted *LLS-All*, the level set constraint h_{LLS} is applied at each node in (4.17h), forming

an exponentially contracting funnel along the horizon. Additionally, for all formulations, the inputs are bounded $\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \bar{\mathbf{u}}$, enforcing $\mathbf{u}_k \in \mathcal{U}$.

CLF-NMPC:

$$\min_{\mathbf{X},\mathbf{U},\mathbf{S}} \quad \phi(s_N) + \sum_{k=0}^{N-1} \frac{1}{2}\mathbf{u}_k^\top \mathbf{u}_k + \phi(s_k) \tag{4.17a}$$

$$\text{s.t} \qquad \mathbf{x}_0 - \hat{\mathbf{x}} = \mathbf{0}, \tag{4.17b}$$

$$\mathbf{x}_{k+1} - \mathbf{f}_k^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, \qquad k = 0, \dots, N-1, \tag{4.17c}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \bar{\mathbf{u}}, \qquad k = 0, \dots, N-1, \tag{4.17d}$$

$$s_k \geq 0, \qquad k = 0, \dots, N, \tag{4.17e}$$

$$\textit{CLF-0}: \qquad h_{CLF}(\hat{\mathbf{x}}, \mathbf{u}_0) \leq s_0, \tag{4.17f}$$

Additionally, for

$$\textit{CLF-ALL}: \qquad h_{CLF}(\mathbf{x}_k, \mathbf{u}_k) \leq s_k, \qquad k = 1, \dots, N-1, \tag{4.17g}$$

$$\textit{LLS-N}: \qquad h_{LLS}(\mathbf{x}_N, \hat{\mathbf{x}}) \leq s_N, \tag{4.17h}$$

$$\textit{LLS-All}: \qquad h_{LLS}(\mathbf{x}_k, \hat{\mathbf{x}}) \leq s_k, \qquad k = 1, \dots, N. \tag{4.17i}$$

4.3.2 Quadratic Approximation Strategy

When applying the SQP algorithm presented in Section 4.2.2 to the nonlinear formulations presented in (4.17), the quadratic subproblem (4.14) is repeatedly solved. As we seek to deploy NMPC on dynamic robotic platforms, it is critical that these optimization problems are well conditioned and do not provide difficulty to numerical solvers. In particular, when B_i in (4.14a) is positive semi-definite (p.s.d.), the resulting QP is convex and can be efficiently solved (Kouzoupis et al., 2018; Stellato et al., 2018).

To ensure this, an approximate p.s.d. Hessian can be used instead of the full Hessian of the Lagrangian. For (4.17), the objective function has a least-squares form, *i.e.*, $F(\mathbf{w}, \mathbf{p}) = \frac{1}{2}\|\mathbf{R}(\mathbf{w}, \mathbf{p})\|^2$, in which case the Gauss-Newton approximation,

$$\mathbf{B}_i \approx \nabla_{\mathbf{w}}\mathbf{R}(\mathbf{w}_i, \mathbf{p})^\top \nabla_{\mathbf{w}}\mathbf{R}(\mathbf{w}_i, \mathbf{p}), \tag{4.18}$$

proves effective in practice (Bock, 1983; Houska, Ferreau, and Diehl, 2011). This neglects the curvature of $\mathbf{R}(\mathbf{w}, \mathbf{p})$, as well as the contribution by the curvature of the constraints. We use this strategy for the *CLF-0* and *CLF-All* formulations.

For the *LLS-N* and *LLS-All* formulations, we retain the contribution of the LLS constraints in the Hessian approximation. The second term in (4.16) is independent of the decision variables, and the properties of this constraint are thus directly determined by the structure of the CLF V. In particular, if the CLF used is convex (as is the case for many constructive techniques for producing CLFs (Kolathaya and Veer, 2019; Taylor et al., 2019)), then the approximation

$$\mathbf{B}_i \approx \nabla_{\mathbf{w}}\mathbf{R}(\mathbf{w}_i, \mathbf{p})^\top \nabla_{\mathbf{w}}\mathbf{R}(\mathbf{w}_i, \mathbf{p}) + \sum_k \mu_{k,LLS} \nabla^2_{\mathbf{w}} h_{k,LLS}, \tag{4.19}$$

remains positive definite, and, as we will show in Section 4.4.3, improves convergence compared to a Gauss-Newton approximation.

4.3.3 Baseline Comparisons

To understand how unifying these two control methodologies impacts performance and stability, it is necessary to compare against baseline controllers given by both (4.9) and (4.11). In particular, elements should be shared between controllers to limit the impact of tuning on performance. To this end, we begin by synthesizing a CLF using feedback-linearization based constructive techniques discussed in (Taylor et al., 2019), which enables the consideration of underactuated systems.

Consider an output $\mathbf{y} : \mathcal{X} \to \mathbb{R}^k$ with relative degree 2 (Sastry, 1999) and $k \leq m$, a time-varying reference trajectory $\mathbf{y} : \mathbb{R}_+ \to \mathbb{R}^k$, and define the tracking error $\mathbf{e} : \mathcal{X} \times \mathbb{R}_+ \to \mathbb{R}^k$:

$$\mathbf{e}(x, t) = \mathbf{y}(x) - \mathbf{y}_d(t). \tag{4.20}$$

A feedback-linearizing controller exists, $\mathbf{k}_{\text{fbl}} : \mathcal{X} \to \mathcal{U}$ that yields linear closed-loop error dynamics given by:

$$\dot{\boldsymbol{\eta}}(\mathbf{x}, t) = \mathbf{A}\boldsymbol{\eta}(\mathbf{x}, t), \quad \boldsymbol{\eta} = \begin{bmatrix} \mathbf{e}(\mathbf{x}, t) \\ \dot{\mathbf{e}}(\mathbf{x}, t) \end{bmatrix} \tag{4.21}$$

where the eigenvalues of $\mathbf{A} \in \mathbb{R}^{2k \times 2k}$ have negative real part. For any positive definite $\mathbf{Q} \in \mathbb{R}^{2k \times 2k}$, the Continuous Time Lyapunov Equation (CTLE):

$$\mathbf{A}^\top \mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q} \tag{4.22}$$

has a positive definite solution $\mathbf{P} \in \mathbb{R}^{2k\times 2k}$. This enables the synthesis of a quadratic (and convex) Lyapunov function $V : \mathcal{X} \times \mathbb{R}_+ \to \mathbb{R}$ given by $V(\mathbf{x}, t) = \boldsymbol{\eta}(\mathbf{x}, t)^\top \mathbf{P}\boldsymbol{\eta}(\mathbf{x}, t)$ with time derivative $\dot{V}(\mathbf{x}, t) = -\boldsymbol{\eta}(\mathbf{x}, t)^\top \mathbf{Q}\boldsymbol{\eta}(\mathbf{x}, t)$, which is negative definite. Furthermore, the existence of the feedback linearizing controller $\mathbf{k}_{\text{fbl}}$ implies that V is a CLF, as:

$$\inf_{\mathbf{u}\in\mathcal{U}} \dot{V}(\mathbf{x}, t, \mathbf{u}) \leq -\boldsymbol{\lambda}_{\min}(\mathbf{Q})\|\boldsymbol{\eta}(\mathbf{x}, t)\|_2^2. \tag{4.23}$$

This CLF can be used in the following *CLF-QP* controller that achieves exponential stability with $\gamma = \frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})}$. The constraint is slacked for numerical conditioning with $z, Z \geq 0$.

CLF-QP:

$$\begin{aligned} \min_{\mathbf{u}\in\mathcal{U}, s\in\mathbb{R}_+} \quad & \frac{1}{2}\mathbf{u}^\top\mathbf{u} + zs + \frac{1}{2}Zs^2 \\ \text{s.t.} \quad & \dot{V}(\mathbf{x}, t, \mathbf{u}) \leq -\boldsymbol{\lambda}_{\min(\mathbf{Q})}\|\boldsymbol{\eta}(\mathbf{x}, t)\|_2^2 + s \end{aligned} \tag{4.24}$$

To synthesize a baseline *NMPC-β* controller, elements from the construction of the CLF can be utilized. In particular, $\mathbf{Q}$ can be used as a running cost on the state, and the terminal value of the CLF can penalized as in (Jadbabaie, Yu, and Hauser, 2001):

NMPC-β:

$$\min_{\mathbf{X},\mathbf{U}} \quad \beta V(\mathbf{x}_N, t_N) + \sum_{k=0}^{N-1} \boldsymbol{\eta}(\mathbf{x}_k, t_k)^\top \mathbf{Q}\boldsymbol{\eta}(\mathbf{x}_k, t_k) + \frac{1}{2}\mathbf{u}_k^\top \mathbf{u}_k \tag{4.25a}$$

$$\text{s.t} \quad \mathbf{x}_0 - \hat{\mathbf{x}} = \mathbf{0}, \tag{4.25b}$$

$$\mathbf{x}_{k+1} - \mathbf{f}_k^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, \qquad k = 0, \dots, N-1, \tag{4.25c}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \bar{\mathbf{u}}, \qquad k = 0, \dots, N-1, \tag{4.25d}$$

That is, the CLF is used as a terminal cost and scaled up with the parameter β. As noted in (Rawlings, Mayne, and Diehl, 2017) if β is selected large enough, stability can be achieved without the need to specify a terminal state constraint. The baseline NMPC problem has constraints on the initial condition and dynamic evolution given by (4.11b) and (4.11c), but does not include any CLF-based constraints. The inputs are also constrained such that $\mathbf{u} \in \mathcal{U}$.

4.4 Simulation & Experimental Results

In this section we provide details on the implementation of the methods established in Section 4.3 on a Segway platform, and discuss simulation and experimental results.

4.4.1 Segway System & Implementation

Dynamics simulations provide an environment for assessing attainable levels of performance of the various approaches. The simulated dynamics model reflects a modified Ninebot E+ Segway platform, seen in Fig. 4.1. We consider a planar representation of the Segway, with state $\mathbf{x} = \begin{bmatrix} r & \theta & \dot{r} & \dot{\theta} \end{bmatrix}^\top \in \mathbb{R}^4$ where r is the horizontal position and θ is the pitch angle. The input to the Segway, $\mathbf{u} = \begin{bmatrix} i \end{bmatrix} \in [-20, 20]$, is current to the Segway motors. The equations of motion are derived via the Newton-Euler equations for an asymmetric, two-wheeled inverted pendulum with torque input. The asymmetry of the system leads to an unforced equilibrium at $\mathbf{x}_e = \begin{bmatrix} 0, \theta_e, 0, 0 \end{bmatrix}$ with $\theta_e = 0.138$. In the NMPC controllers, a forward Euler time discretization is used with $\delta t = 0.01\,\text{s}$.

State estimation on the physical Segway is done with wheel encoders and Inertial Measurement Unit (IMU) data from a VectorNav VN-100. All computations are performed on board on an ARM Cortex-A57 (quad-core) @ 2 GHz CPU running the ERIKA3 RTOS. For each NMPC formulation, all functions, gradients, and Hessians in (4.14) are found using the CasADi auto-differentiation framework (Andersson et al., 2019). This leads to a QP with a fixed sparsity pattern, which we solve with the sparsity-exploiting solver OSQP (Stellato et al., 2018). We solve a single QP per control iteration, unless otherwise stated.

4.4.2 Simulation Results

To compare the behavior of the different control approaches, we considered a stabilization task. In particular, we simulated the system under each controller with a fixed initial condition and an objective of stabilizing to the unforced equilibrium point $\mathbf{x}_e$. The performance of each controller was quantified by the average input norm over a 2 second time horizon, which provides an assessment of the total input used. These averages appear in Table 4.1.

We see that the *CLF-QP* and *CLF-0* controllers select identical inputs over the entire trajectory and have the largest average input of any CLF-based controllers. This indicates that the addition of a prediction horizon to the

Table 4.1: Average input norm along the simulation horizon against prediction horizon length (N) for different controller formulations defined in Section 4.3. Absences indicate failure to stabilize the system.

N	1	10	20	30	40	50
CLF-QP	1.085	1.085	1.085	1.085	1.085	1.085
CLF-0	1.085	1.085	1.085	1.085	1.085	1.085
CLF-All	1.085	1.072	0.952	0.849	**0.794**	**0.769**
LLS-N	1.083	0.957	0.889	0.842	0.808	0.784
LLS-All	1.083	0.956	0.887	0.839	0.805	0.782
NMPC-0.1	-	-	3.232	2.435	2.036	1.783
NMPC-1	-	3.026	2.019	1.732	1.574	1.471
NMPC-10	**0.828**	**0.607**	**0.704**	**0.823**	0.926	1.006

baseline *CLF-QP* controller only improves its performance if the stability h_{CLF} constraint is applied further along the horizon. We also see that with no horizon ($N=1$), all of the CLF-based controllers recover similar performance to the baseline *CLF-QP*. If we consider the controllers that impose constraints further along the horizon we see that their average input consistently decreases with horizon length, with the *CLF-All* controller marginally outperforming the *LLS* controllers at the longest horizon lengths.

The performance of the baseline *NMPC-*β controllers heavily depended on the weighting of the terminal cost. At shorter horizons the *NMPC-0.1* and *NMPC-1* controllers failed to stabilize the system, and saw improved performance as the horizon grew longer. In contrast, the *NMPC-10* controller demonstrated the best performance of all controllers at shorter horizons, but saw worsening performance as the horizon grew longer. This illustrates the issues that arise when both stability and performance are achieved through the cost, rather than decoupled through constraints as in the CLF-based formulations.

To more clearly understand the possible behaviors of the various *CLF-NMPC* controllers, we visualize the solutions to each controller obtained at the initial condition, *i.e.*, $\hat{\mathbf{x}} = \mathbf{x}(0)$ in Fig. 4.2. As the *CLF-0* controller is only required to satisfy the stability constraint at the initial node, its input quickly drops to zero and the h_{CLF} constraint is violated in the next step in the horizon. In contrast, the *CLF-All* controller meets the h_{CLF} constraint along the entirety of the horizon as required. Despite meeting this constraint on $\dot{V}$ at each node, the *CLF-All* controller slightly fails to meet the implied level-set bounds along the horizon. This is due to the fact that the constraint is checked only at the

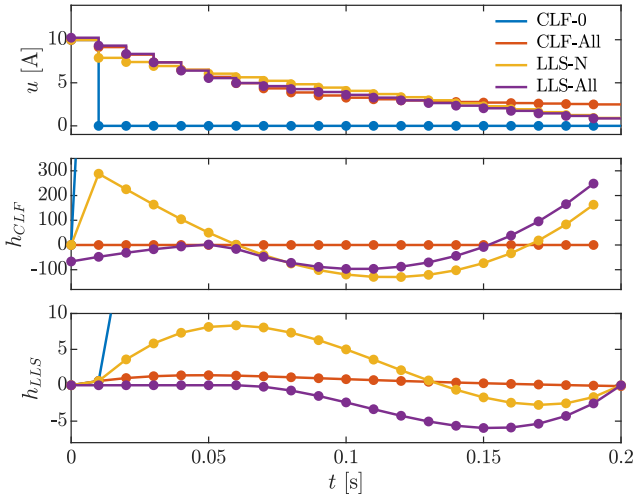


Figure 4.2: Initial *CLF-NMPC* solutions stabilizing to $\mathbf{x}_e$ for $N = 20$ and $\hat{\mathbf{x}} = [0, \frac{\pi}{8}, 0, 0]^\top$. **Top:** The optimal input sequence determined by each controller, with the *CLF-0* controller dropping to zero beyond the first node. **Middle:** Evaluation of the bound on $\dot{V}$ in (4.15). The controllers with the h_{CLF} constraint meet this bound at all required points, while the two *LLS* controllers violated it at various points along the trajectory. **Bottom:** Evaluation of the the level set bound in (4.16). This bound is satisfied at all necessary points by the *LSS* controllers, but is violated by the *CLF-All* controller due to the zero-order-hold implementation.

beginning of each interval and is not required to hold over the interval which the control input is held over.

The *LLS* controllers both satisfy the level constraints at the required points, with the *LLS-N* controller violating the level set bounds earlier in the horizon. Despite meeting these level set bounds and the required points, both controllers violate the associated bound on $\dot{V}$, with the *LLS-All* controller satisfying the h_{LLS} constraint loosely early in the trajectory before satisfying it tightly at the end of the trajectory.

The evolution of the system under these controllers with a horizon length $N = 30$ is captured in Fig. 4.3. We see that all CLF-based controllers satisfy the stability constraint (4.15) during the entire simulation. The most significant difference in the behavior of these controllers arises in the magnitude of the input taken early in the trajectory. The best performing controllers applying

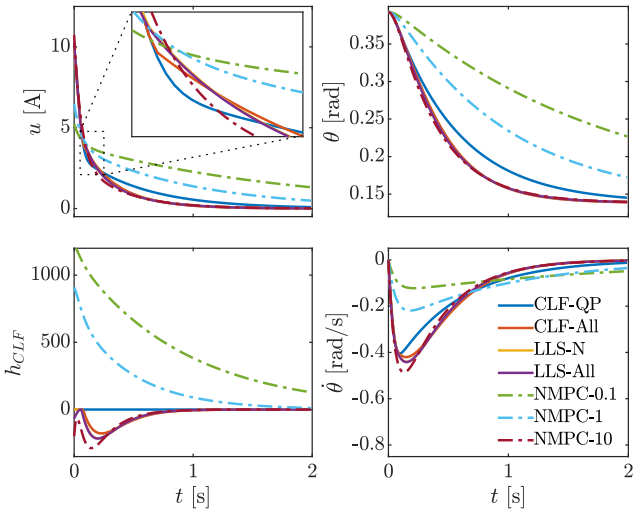


Figure 4.3: Simulation results stabilizing to $\mathbf{x}_e$ for $\mathbf{x}(0) = [0, \frac{\pi}{8}, 0, 0]^\top$. The evaluation of the CLF constraint (4.15) along the simulation horizon is shown in the bottom-left plot. The system input, pitch angle, and pitch rate are shown in the top-left, top-right, and bottom-right respectively. The lower weighted *NMPC-β* controllers take significantly more time to converge to the equilibrium point.

higher inputs earlier in the trajectory to stabilize the system quickly and avoid accruing input over more of the trajectory.

We additionally perform a simulation in the opposite direction, driving the Segway from its unforced equilibrium $\mathbf{x}_e$ to a desired state, $\mathbf{x}_d = [0.0, \frac{\pi}{8}, 0.0, 0.0]^\top$. The resulting trajectories are seen in Fig. 4.4. In this scenario we see that the *CLF-QP* controller demonstrates significantly different behavior from the *CLF-NMPC* controllers. In particular, the *CLF-QP* controller takes an initial input to drive the system towards the desired state, and then allows gravity to carry it to this point. This results in a large overshoot past the desired state, where as the *CLF-NMPC* controllers approach the equilibrium point more slowly.

The *NMPC-β* controllers fail to converge to the desired state. This arises due to the fact that the cost on the input is not centered about the input that makes the desired state an equilibrium point, *i.e.*, $\|\mathbf{u} - \mathbf{u}_{\text{ref}}\|_2^2$. Because both the input and the state error appear in the cost, larger error is accepted in the state to minimize the input. This highlights the flexibility in choosing the cost function in the *CLF-NMPC* formulations. General cost functions that do

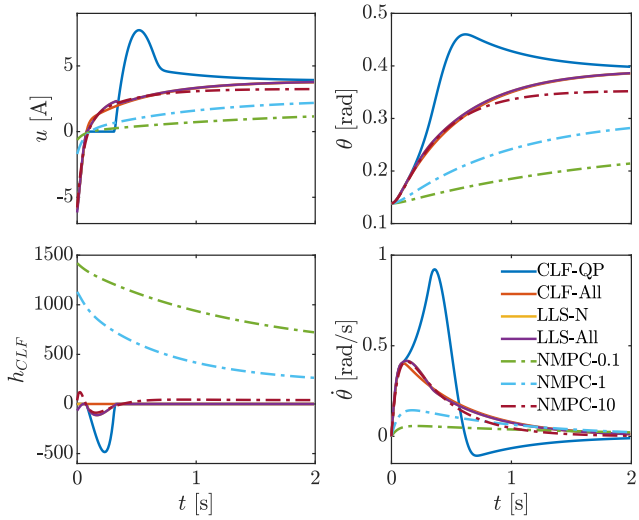


Figure 4.4: Simulation results stabilizing to $\mathbf{x}_d = [0, \frac{\pi}{8}, 0, 0]^\top$ from $\mathbf{x}^\star$. The evaluation of the CLF constraint (4.15) along the simulation horizon is shown in the bottom-left plot. The system input, pitch angle, and pitch rate are shown in the top-left, top-right, and bottom-right respectively. The *CLF-QP* controller overshoots the desired state, while the *CLF-NMPC* controllers slowly approach it. The *NMPC-β* controllers do not converge due to the trade-off between input and state error in the cost function.

not necessarily obtain their minimum at the goal can be used, which opens up the possibility of using economic cost functions (Heidarinejad, Liu, and Christofides, 2012).

4.4.3 Numerical Results

To understand the feasibility of deploying these control approaches on the computationally limited Segway platform, we investigated the convergence rate of each formulation on a single instance of the NMPC problem. We also considered variants of the *LLS* formulations using the Gauss-Newton approximation in (4.18) (denoted *LLS-N*^{GN} and *LLS-All*^{GN}) and compare them to using the modified Hessian in (4.19).

We execute Algorithm 3 for each formulation until the constraints are sufficiently satisfied, $\|\mathbf{c}(\mathbf{w})\|_1 \leq 10^{-6}$, and no further progress is made in cost, $|F(w_i) - F(w_{i-1})| \leq 10^{-6}$. The optimization is fully cold-started such that

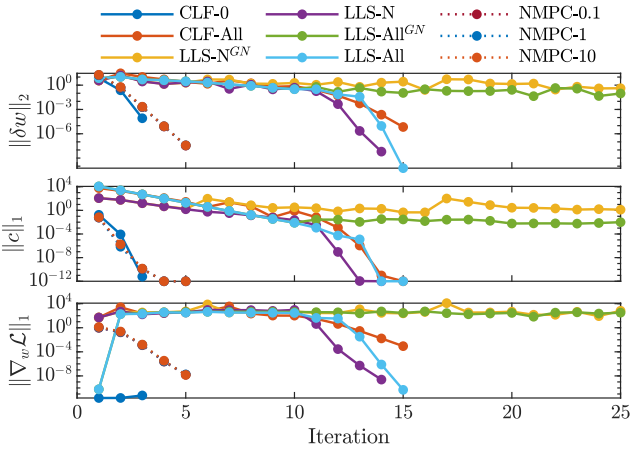


Figure 4.5: Convergence of Algorithm 3 applied to a cold-started NMPC problem with $\hat{\mathbf{x}} = \mathbf{x}_d = [0, \frac{\pi}{8}, 0, 0]^\top$. **Top:** Step size, $\|\delta \mathbf{w}\|_2$, **Middle:** Constraint violation, $\|\mathbf{c}(\mathbf{w})\|_1$, **Bottom:** First order optimality condition, $\|\nabla_{\mathbf{w}} \mathcal{L}\|_1$. The *CLF-0* and *NMPC-β* methods converge quickly, while the *CLF-NMPC* methods take longer. The *LLS* formulations without the modified Hessian in (4.19) stop progressing beyond a certain point.

all decision variables and Lagrange multipliers are initialized to zero. The problem is set up with $\hat{\mathbf{x}} = \mathbf{x}_d = [0, \frac{\pi}{8}, 0, 0]^\top$ and a horizon length of $N = 30$. The step size, constraint satisfaction, and first order optimality are plotted in Fig. 4.5.

The *CLF-0* and *NMPC-β* formulations converge rapidly as they have a quadratic cost function and affine constraints. The only nonlinearity in these problems therefore arises from the system dynamics, making the QP subproblem a good model for the full problem. The *CLF-All*, *LLS-N*, and *LLS-All* have nonlinear constraints along the horizon, and therefore take significantly longer to converge. Nonetheless, the convergence rate accelerates over the last few iterations, indicating that these formulations can still perform well when starting close to the solution. Finally, the *LLS-N*^{GN}, and *LLS-All*^{GN} formulations initially decrease the constraint violation, but fail to make further progress after a certain point.

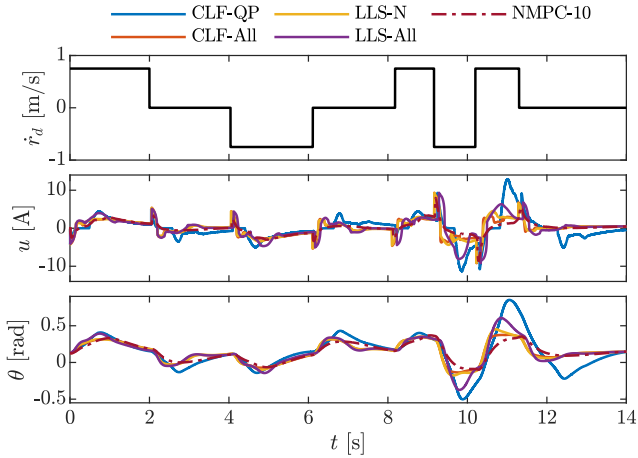


Figure 4.6: Experimental results from trajectory tracking. **Top:** Desired velocity profile, **Middle:** Input profile, **Bottom:** Pitch angle profile. The *CLF-QP* controller displays more aggressive behavior due to no prediction horizon, while the *NMPC-1* controller fails to stabilize the system.

4.4.4 Experimental Results

In addition to simulation, we also demonstrate the ability of this unified control approach on the physical Segway in Fig. 4.1. In particular, we define a desired angular trajectory in time:

$$\theta_d(\mathbf{x}, t) = \theta_e - K_v(\dot{r} - \dot{r}_d(t)), \tag{4.26}$$

where $\dot{r}_d(t)$ is a commanded velocity and $K_v = 0.3$ is a velocity feedback gain, leading to error dynamics given by $\mathbf{e}(\mathbf{x}, t) = [\theta - \theta_d(\mathbf{x}, t)]$. These dynamics are used to synthesize a quadratic CLF as per Section 4.3. This CLF was used to formulate a *CLF-QP* controller given by (4.24), a *NMPC-1* and *NMPC-10* controller with cost given by (4.25a), and *CLF-All*, *LLS-N*, and *LLS-All* controllers. Each controller was used to track the desired trajectory and then stabilize the system. The results of these experiments can be seen in Fig. 4.6 and the accompanying video[1]. We see that all controllers except the *NMPC-1* controller are able to stabilize the system. The *CLF-QP* displays aggressive behavior compared to the *NMPC* controllers as it does not incorporate a prediction horizon. The average input and control frequency

[1] `https://youtu.be/weNv-FlRKiE`

along the experimental horizon is seen in Table 4.2. We see that at a horizon of $N = 30$ the *NMPC-10* controller has the best performance, with the *NMPC-1* controller omitted due to failure to stabilize. Of the *CLF-NMPC* controllers the *CLF-All* controller has the best performance. We note that although the baseline *NMPC-10* controller outperforms the proposed methods, this required tuning of the cost function and matches the behavior seen in simulation at this horizon length. We see that the *CLF-NMPC* methods have a higher computational cost than the *NMPC-10* controller. This follows as the NLP has additional constraints related to stability that must be met. In that sense, *LLS-N* is an appealing approach among the *CLF-NMPC* methods, as it imposes only two stability constraints.

Table 4.2: Average input norm ($\|\mathbf{u}\|_{\text{avg}}$) and computation time ($t_{CPU}$) in ms along the experiment horizon with prediction horizon $N = 30$ for the different controller formulations defined in Section 4.3.

	CLF-QP	CLF-All	LLS-N	LLS-All	NMPC-10
$\|\mathbf{u}\|_{\text{avg}}$	2.081	1.594	1.666	1.898	**1.152**
t_{CPU}	1.25	5.56	4.17	6.13	3.11

4.5 Conclusion

In conclusion, we have presented a novel set of approaches for unifying CLFs and NMPC on robotic platforms with limited computational resources. The use of a SQP algorithm with modified Hessian was proposed to efficiently solve the resulting nonlinear optimization problem. The different unified formulations were analyzed in simulation, for the first time demonstrated on hardware, and were shown to improve performance beyond baseline CLF and NMPC methods. In particular for forced equilibria, the CLF-NMPC method converges without modifications while the cost-driven baseline NMPC does not. Furthermore, the unified methods all achieved stability where as the stability baseline NMPC methods was sensitive to cost function parameters. As system complexity increases, such manual tuning becomes increasingly difficult. In this space, we see an opportunity for the presented CLF-NMPC methods, where stability is explicitly embedded and requires no further tuning.

5

Multi-layered safety for legged robots via control barrier functions and model predictive control

Grandia, R., Taylor, A. J., Ames, A. D., and Hutter, M. (2021). "Multi-layered safety for legged robots via control barrier functions and model predictive control". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8352–8358

DOI: 10.1109/ICRA48506.2021.9561510

Video: https://youtu.be/TCDIirXfByE

The problem of dynamic locomotion over rough terrain requires both accurate foot placement together with an emphasis on dynamic stability. Existing approaches to this problem prioritize immediate safe foot placement over longer term dynamic stability considerations, or relegate the coordination of foot placement and dynamic stability to heuristic methods. We propose a multi-layered locomotion framework that unifies Control Barrier Functions (CBFs) with Model Predictive Control (MPC) to simultaneously achieve safe foot placement and dynamic stability. Our approach incorporates CBF based safety constraints both in a low frequency kino-dynamic MPC formulation and a high frequency inverse dynamics tracking controller. This ensures that safety-critical execution is considered when optimizing locomotion over a longer horizon. We validate the proposed method in a 3D stepping-stone scenario in simulation and experimentally on the ANYmal quadruped platform.

5.1 Introduction

A key motivation behind the development of legged robots is their ability to overcome complex terrain. Because legged locomotion only requires discrete footholds, obstacle such as steps, gaps, and stairs can be traversed, making legged robots a compelling alternative to wheeled systems. When a statically stable motion pattern is considered, several mature strategies for rough terrain locomotion have been proposed and successfully demonstrated on hardware for bipedal (Griffin et al., 2019), quadrupedal (Fankhauser et al., 2018; Mastalli et al., 2020b), and hexapedal (Belter, Labecki, and Skrzypczynski, 2016) robots. However, inspired by the fast and dynamic motions seen in nature, the use of dynamic gaits—a gait where individual contact phases are statically unstable—is still an active area of research.

The challenge in dynamic locomotion lies in the fact that foothold locations are not only constrained by the terrain, but also affect the dynamic stability of the resulting contact configuration. Additionally, as the speed of the motions increases, the inertial and nonlinear effects described by the full rigid body dynamics of the system become more relevant. There is therefore a need for methods that can guarantee a safe foot placement while simultaneously considering the future impact on the dynamic stability of the system. A classical locomotion challenge that demands safe foot placement and dynamic stabilization is the "stepping-stones" scenario, see Fig. 5.1, where viable foothold locations are discontinuous and sparsely available. We propose to combine the safety guarantees endowed by CBFs with the longer horizon considered in MPC to guarantee safe foot placement while achieving dynamic locomotion and high tracking performance.

5.1.1 Related Work

Control Barrier Functions (Ames, Grizzle, and Tabuada, 2014) are a tool for synthesizing controllers that ensure safety of nonlinear systems (Ames et al., 2019; Jankovic, 2018). Moreover, CBFs have been used in the stepping-stones problem via a Quadratic Programming (QP) based tracking controller (Nguyen et al., 2016; Nguyen and Sreenath, 2015). An offline optimized walking trajectory, or a library thereof (Nguyen et al., 2020), is tracked and locally modified to satisfy CBF safety constraints. While promising in simulation, we are not aware of the successful transfer of a CBF based stepping controller to hardware, despite extensions that add robustness (Nguyen and Sreenath, 2016a), or a learning based model error correction (Choi et al., 2020). Indeed, in (Nguyen

Figure 5.1: ANYmal (Hutter et al., 2016) performing a trotting gait on stepping-stones.

et al., 2018), the stepping-stones problem is demonstrated experimentally by increasing the look-ahead horizon of the gait library and through subsequent gait interpolation rather than a CBF based method. We hypothesize that it is exactly this reasoning over a longer horizon that is missing with the CBF-QP control formulation.

In contrast, Model Predictive Control has become a central method for the online synthesis and control of dynamic systems over a given time horizon (Rawlings, Mayne, and Diehl, 2017). In the context of the stepping-stones problem, a distinction can be made between MPC based approaches where the footholds locations are determined separately from the torso motion optimization (Jenelten et al., 2020; Kim et al., 2020; Villarreal et al., 2020), and MPC based approaches where the foothold location and torso motions are jointly optimized. The benefit of jointly optimizing torso and leg motions has been demonstrated in the field of trajectory optimization (Dai, Valenzuela, and Tedrake, 2014; Winkler et al., 2018). Following this idea, real-time capable methods have been proposed with the specification of leg motions made at the position (Bledt, Wensing, and Kim, 2017), velocity (Farshidian et al., 2017b), or acceleration level (Neunert et al., 2018). One challenge of this approach is its computational costs, which can be resolved by coupling a low-frequency MPC controller with a high-frequency tracking controller (Grandia et al., 2019b).

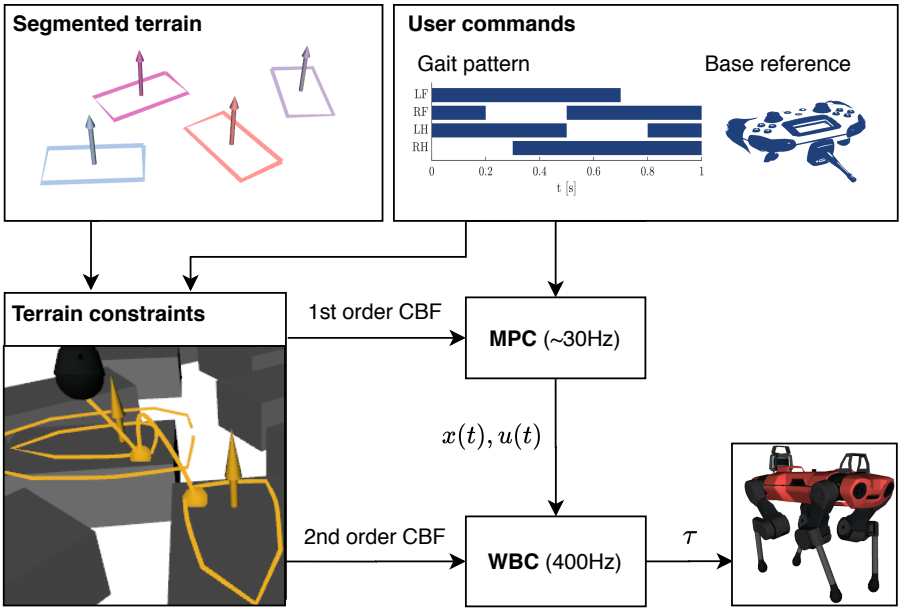


Figure 5.2: Overview of the proposed multi-layed control setup showing both the MPC and Whole-Body Controller (WBC) layer receiving terrain CBF constraints.

5.1.2 Contribution

In this work, we build upon a kino-dynamic MPC formulation (Farshidian et al., 2017b) where joint velocities and contact forces are decision variables in a low frequency MPC controller. This allows direct integration of CBF safety constraints into the MPC formulation similar to (Grandia et al., 2020). By jointly optimizing torso and leg motions our method avoids the heuristic coordination that is needed when foot placement and torso motion are delegated to separate controllers. A higher rate tracking controller is implemented that fuses inverse dynamics with the CBF safety constraints to offer guarantees of safety with the whole-body dynamics in consideration. In the context of collision avoidance, CBFs can be thought of (and have shown to be) a generalization of artificial potential fields used in inverse dynamics methods (Khatib, 1985)(Singletary et al., 2021). Finally, we note that the combination of discrete time CBFs with MPC has been considered in (Zeng, Zhang, and Sreenath, 2021), but it did not consider a multi-layered approach nor provided experimental results.

The main contributions of this work are two-fold. First, we propose a multi-layered control approach that combines CBFs with MPC (see Fig. 5.2). This framework allows CBF safety constraints on the position coordinates of robotic systems to be incorporated in a low frequency MPC controller determining desired velocities as well as in a high frequency tracking controller that incorporates the dynamics of the system. Compared to standard CBF approaches, this adds a horizon when determining safe control inputs. Compared to MPC approaches, the safety critical constraint is enforced at a higher rate, and incorporates a higher fidelity whole-body dynamics model. The second contribution is, to the best of the author's knowledge, the first successful experimental demonstration of CBFs, not only as an approach to the stepping-stones problem, but on a legged robot.

5.2 Background

This section provides a review of CBFs and nonlinear MPC. Consider the nonlinear control affine system given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{5.1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$. $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are locally Lipschitz continuous on $\mathbb{R}^n$. Given a Lipschitz continuous state-feedback controller $\mathbf{k} : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}^m$, the closed-loop system dynamics are:

$$\dot{\mathbf{x}} = \mathbf{f}_{\text{cl}}(\mathbf{x}, t) \triangleq \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{k}(\mathbf{x}, t). \tag{5.2}$$

The assumption on local Lipschitz continuity of $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{k}$ implies that $\mathbf{f}_{\text{cl}}$ is locally Lipschitz continuous. Thus for any initial condition $\mathbf{x}_0 := \mathbf{x}(0) \in \mathbb{R}^n$ there exists a maximum time interval $I(\mathbf{x}_0) = [0, t_{\max})$ such that $\mathbf{x}(t)$ is the unique solution to (5.2) on $I(\mathbf{x}_0)$ (Perko, 2013).

5.2.1 Control Barrier Functions

The notion of safety that we consider in this paper is formalized by specifying a *safe set* in the state space that the system must remain in. In particular, consider a time-varying set $\mathcal{C}_t \subset \mathbb{R}^n$ defined as the 0-superlevel set of a continuously differentiable function $h : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}$, yielding:

$$\mathcal{C}_t \triangleq \{\mathbf{x} \in \mathbb{R}^n : h(\mathbf{x}, t) \geq 0\}, \tag{5.3}$$

We refer to $\mathcal{C}_t$ as the *safe set*. This construction motivates the following definitions of forward invariant and safety:

Definition 2 (*Forward Invariant & Safety*)**.** A time-varying set $\mathcal{C}_t \subset \mathbb{R}^n$ is *forward invariant* if for every $\mathbf{x}_0 \in \mathcal{C}_0$, the solution $\mathbf{x}(t)$ to (5.2) satisfies $\mathbf{x}(t) \in \mathcal{C}_t$ for all $t \in I(\mathbf{x}_0)$. The system (5.2) is *safe* on the set $\mathcal{C}_t$ if the set $\mathcal{C}_t$ is forward invariant.

Certifying the safety of the closed-loop system (5.2) with respect to a set $\mathcal{C}_t$ may be impossible if the controller $\mathbf{k}$ was not chosen to enforce the safety of $\mathcal{C}_t$. Control Barrier Functions can serve as a synthesis tool for attaining the forward invariance, and thus the safety of a set. Before defining CBFs, we note a continuous function $\alpha : (-\infty, \infty) \to \mathbb{R}$, is said to belong to *extended class* $\mathcal{K}_\infty$ *(*$\alpha \in \mathcal{K}_{\infty,]}$*)* if α is strictly monotonically increasing, $\alpha(0) = 0$, and if $\lim_{r\to\infty} \alpha(r) = \infty$, and $\lim_{r\to-\infty} \alpha(r) = -\infty$.

Definition 3 (*Control Barrier Function (CBF)*, (Ames et al., 2017))**.** Let $\mathcal{C}_t \subset \mathbb{R}^n$ be the time-varying 0-superlevel set of a continuously differentiable function $h : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}$ with 0 a regular value. The function h is a time-varying *Control Barrier Function* (CBF) for (5.1) on $\mathcal{C}_t$ if there exists $\alpha \in \mathcal{K}_{\infty,e}$ such that for all $\mathbf{x} \in \mathbb{R}^n$ and $t \in \mathbb{R}_+$:

$$\sup_{\mathbf{u}\in\mathbb{R}^m} \dot{h}(\mathbf{x}, t, \mathbf{u}) \triangleq \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}, t)\,(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}) + \frac{\partial h}{\partial t}(\mathbf{x}, t) \geq -\alpha(h(\mathbf{x}, t)). \quad (5.4)$$

Controllers that take inputs satisfying (5.4) ensure the safety of the closed-loop system (5.2) (Ames, Grizzle, and Tabuada, 2014).

Given a nominal (but not necessarily safe) locally Lipschitz continuous controller $\mathbf{k}_d : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}^m$, a possible controller taking values satisfying (5.4) is the safety-critical CBF-QP:

$$\begin{aligned} \mathbf{k}(\mathbf{x}, t) = \ & \underset{\mathbf{u}\in\mathbb{R}^m}{\operatorname{argmin}} \quad \frac{1}{2}\|\mathbf{u} - \mathbf{k}_d(\mathbf{x}, t)\|_2^2 \\ & \text{s.t.} \qquad \dot{h}(\mathbf{x}, t, \mathbf{u}) \geq -\alpha(h(\mathbf{x}, t)). \end{aligned} \quad \text{(CBF-QP)}$$

5.2.2 Nonlinear Model Predictive Control

We consider the following nonlinear optimal control problem with cost functional

$$\min_{\mathbf{u}(\cdot)} \quad \Phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t), t)\ \mathrm{d}t, \tag{5.5}$$

where $\mathbf{x}(t)$ is the state, $\mathbf{u}(t)$ is the input at time t, $L(\cdot)$ is an intermediate cost, and $\Phi(\cdot)$ is the cost at the terminal state $\mathbf{x}(T)$. The goal is to find a continuous control signal $\mathbf{u} : I(\mathbf{x}_0) \to \mathbb{R}^m$ that minimizes this cost subject to the system dynamics, initial condition, and general constraints:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \tag{5.6}$$
$$\mathbf{x}(0) = \mathbf{x}_0, \tag{5.7}$$
$$\mathbf{g}(\mathbf{x}, \mathbf{u}, t) = \mathbf{0}, \tag{5.8}$$
$$\mathbf{h}(\mathbf{x}, \mathbf{u}, t) \geq \mathbf{0}. \tag{5.9}$$

Various methods exist to solve this problem (Rawlings, Mayne, and Diehl, 2017), and a detailed discussion is beyond the scope of this paper. In this work we use the Sequential Linear Quadratic (SLQ) method, which is a Differential Dynamic Programming (DDP) based algorithm for continuous-time systems. In particular, the method in (Grandia et al., 2019b) is being used which extends the SLQ formulation of (Farshidian et al., 2017a) for use with inequality constraints.

5.3 Multi-Layered Control Formulation

In this section we present a multi-layered control formulation that unifies CBFs with MPC to achieve safety and longer horizon optimality for a general robotic system. Consider a robotic system with generalized coordinates $\mathbf{q} \in \mathbb{R}^d$ and coordinate rates $\dot{\mathbf{q}} \in \mathbb{R}^d$ with dynamics given by:

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\boldsymbol{\tau}, \tag{5.10}$$

with inertia matrix $\mathbf{D}$, centrifugal and Coriolis terms $\mathbf{C}$, gravitational forces $\mathbf{g}$, actuation matrix $\mathbf{B}$, and torques $\boldsymbol{\tau} \in \mathbb{R}^m$. Consider a continuously differentiable function $h : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ that determines a time-varying safe set for the position coordinates of the robot, with a time derivative given by:

$$\dot{h}(\mathbf{q}, \dot{\mathbf{q}}, t) = \frac{\partial h}{\partial \mathbf{q}}(\mathbf{q}, t)\dot{\mathbf{q}} + \frac{\partial h}{\partial t}(\mathbf{q}, t). \tag{5.11}$$

The torques $\boldsymbol{\tau}$ do not appear in this time derivative, making it impossible to choose inputs that ensure the barrier constraint:

$$\dot{h}(\mathbf{q}, \dot{\mathbf{q}}, t) \geq -\alpha_1(h(\mathbf{q}, t)), \tag{5.12}$$

is met for some $\alpha_1 \in \mathcal{K}_{\infty,e}$. This challenge is often resolved through the notion of *exponential* CBFs (Nguyen and Sreenath, 2016b), in which an auxiliary function $h_e : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ is defined as:

$$h_e(\mathbf{q}, \dot{\mathbf{q}}, t) = \dot{h}(\mathbf{q}, \dot{\mathbf{q}}, t) + \alpha_1(h(\mathbf{q}, t)), \tag{5.13}$$

$$\dot{h}_e(\mathbf{q}, \dot{\mathbf{q}}, t, \boldsymbol{\tau}) = \frac{\partial h_e}{\partial \mathbf{q}}(\mathbf{q}, \dot{\mathbf{q}}, t)\dot{\mathbf{q}} + \frac{\partial h_e}{\partial \dot{\mathbf{q}}}(\mathbf{q}, \dot{\mathbf{q}}, t)\ddot{\mathbf{q}} + \frac{\partial h_e}{\partial t}(\mathbf{q}, \dot{\mathbf{q}}, t). \tag{5.14}$$

As $\ddot{\mathbf{q}}$ appears in affine relation to $\boldsymbol{\tau}$ in (5.10), h_e can serve as a CBF for the set $\mathcal{C}_{t,e} \triangleq \left\{(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{2d} : h_e(\mathbf{q}, \dot{\mathbf{q}}, t) \geq 0\right\}$ by enforcing:

$$\dot{h}_e(\mathbf{q}, \dot{\mathbf{q}}, t, \boldsymbol{\tau}) \geq -\alpha_2(h_e(\mathbf{q}, \dot{\mathbf{q}}, t)), \tag{5.15}$$

for some $\alpha_2 \in \mathcal{K}_{\infty,e}$. Enforcing the forward invariance of this set implies the desired safety constraint (5.12) is met, implying the forward invariance of the set $\mathcal{C}_t \cap \mathcal{C}_{t,e}$. Thus the constraint on the position coordinates of the robot are met.

Typical approaches using exponential CBFs only enforce the final constraint (5.15), often in a CBF-QP controller (Rosolia and Ames, 2021). In practice, when the desired controller $\mathbf{k}_d$ is synthesized without considering safety, this can lead to aggressive behavior when the system approaches the boundary of the safe set. Using MPC in a multi-layered setup allows the safety constraint to be incorporated into the specification of $\mathbf{k}_d$. When the MPC directly operates on the full state and input of (5.10), the safety constraint in (5.15) is readily incorporated, as was done in (Zeng, Zhang, and Sreenath, 2021). In contrast, we consider a MPC controller that operates on a reduced order model in which case the barrier constraint in (5.12) is added to the MPC problem instead. For simplicity of exposition, we present here an MPC layer that operates on a purely kinematic model of the system.

Given a current estimate of the state $(\hat{\mathbf{q}}, \dot{\hat{\mathbf{q}}})$ at time $\hat{t}$, a kinematic MPC solves the following optimization problem:

Low-Frequency Safe Kinematic MPC:

$$\min_{\mathbf{q}^d(t), \dot{\mathbf{q}}^d(t)} \quad \Phi(\mathbf{q}^d(T)) + \int_0^T L(\mathbf{q}^d(t), \dot{\mathbf{q}}^d(t), t)\ \mathrm{d}t,$$

$$\begin{aligned} \text{s.t} \quad & \mathbf{q}^d(0) = \hat{\mathbf{q}}, \\ & \frac{\partial \mathbf{q}^d}{\partial t} = \dot{\mathbf{q}}^d, \\ & \dot{h}(\mathbf{q}^d, \dot{\mathbf{q}}^d, t) + \alpha_1(h(\mathbf{q}^d, t)) \geq 0, \end{aligned}$$

where $\mathbf{q}^d(t)$ and $\dot{\mathbf{q}}^d(t)$ are trajectories of generalized coordinates and velocities, forming the safe desired trajectory for the tracking controller. A desired acceleration is obtained through a combination of tracking terms and a forward difference of the desired velocities:

$$\ddot{\mathbf{q}}^d = \frac{\dot{\mathbf{q}}^d(\hat{t} + \delta t) - \dot{\mathbf{q}}^d(\hat{t})}{\delta t} + \mathbf{D}(\dot{\mathbf{q}}^d(\hat{t}) - \dot{\hat{\mathbf{q}}}) + \mathbf{P}(\mathbf{q}^d(\hat{t}) - \hat{\mathbf{q}}). \tag{5.16}$$

Drawing inspiration from the inverse dynamics approach in (Reher, Kann, and Ames, 2020), the high-frequency controller is given by:

High-Frequency ID-CBF-QP:

$$\begin{aligned} k(\mathbf{q}, \dot{\mathbf{q}}, t) = \underset{\boldsymbol{\tau},\, \ddot{\mathbf{q}}}{\operatorname{argmin}} \quad & \frac{1}{2}\|\ddot{\mathbf{q}} - \ddot{\mathbf{q}}^d\|_2^2 \\ \text{s.t.} \quad & \mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\boldsymbol{\tau}, \\ & \dot{h}_e(\mathbf{q}, \dot{\mathbf{q}}, t, \boldsymbol{\tau}) \geq -\alpha_2(h_e(\mathbf{q}, \dot{\mathbf{q}}, t)). \end{aligned}$$

This controller seeks to track the desired acceleration determined by the low-frequency MPC controller while ensuring that the full dynamics are incorporated into the determination of safe inputs according to (5.15).

5.4 ANYmal Implementation

In this Section we provide an overview of how the multi-layer control formulation discussed in Section 5.3 is applied to the ANYmal quadrupedal robotic platform. An overview of the control structure is provided in Fig. 5.2.

5.4.1 MPC System Model

We apply our approach to the kino-dynamic model of a quadruped robot, which describes the dynamics of a single free-floating body along with the kinematics for each leg. The state $\mathbf{x} \in \mathbb{R}^{24}$ and input $\mathbf{u} \in \mathbb{R}^{24}$ are defined as:

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{\theta}^\top, & \mathbf{p}^\top, & \boldsymbol{\omega}^\top, & \mathbf{v}^\top, & \mathbf{q}^\top \end{bmatrix}^\top, \quad \mathbf{u} = \begin{bmatrix} \boldsymbol{\lambda}_B^\top, & \dot{\mathbf{q}}^{d^\top} \end{bmatrix}^\top, \tag{5.17}$$

where $\boldsymbol{\theta} \in \mathbb{R}^3$ is the orientation of the base in Euler angles, $\mathbf{p} \in \mathbb{R}^3$ is the position of the center of mass in the world frame $\mathcal{F}_W$, $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular rate of the base, $\mathbf{v} \in \mathbb{R}^3$ is the linear velocity of the center of mass in the body frame $\mathcal{F}_B$, and $\mathbf{q} \in \mathbb{R}^{12}$ is the joint positions. The joint positions for leg i are given by $\mathbf{q}_i \in \mathbb{R}^3$. The inputs of the model are end-effector contact forces $\boldsymbol{\lambda}_B \in \mathbb{R}^{12}$ in the body frame and desired joint velocities $\dot{\mathbf{q}}^d \in \mathbb{R}^{12}$ with equations of motion:

$$\begin{cases} \dot{\boldsymbol{\theta}} = \mathbf{T}(\boldsymbol{\theta})\boldsymbol{\omega}, \\ \dot{\mathbf{p}} = {}_W\mathbf{R}_B(\boldsymbol{\theta})\,\mathbf{v}, \\ \dot{\boldsymbol{\omega}} = \mathbf{I}^{-1}\left(-\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} + \sum_{i=1}^{4} \mathbf{r}_{B_i}(\mathbf{q}_i) \times \boldsymbol{\lambda}_{B_i}\right), \\ \dot{\mathbf{v}} = \mathbf{g}(\boldsymbol{\theta}) + \frac{1}{m}\sum_{i=1}^{4} \boldsymbol{\lambda}_{B_i}, \\ \dot{\mathbf{q}} = \dot{\mathbf{q}}^d, \end{cases}$$

where ${}_W\mathbf{R}_B : \mathbb{R}^3 \to SO(3)$ is the rotation matrix from $\mathcal{F}_B$ to $\mathcal{F}_W$ and $\mathbf{T} : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ transforms angular velocities to the Euler angles derivatives. Model parameters include the gravitational acceleration in the body frame $\mathbf{g} : \mathbb{R}^3 \to \mathbb{R}^3$, the total mass $m \in \mathbb{R}_+$, and the moment of inertia $\mathbf{I} \in \mathbb{R}^{3\times3}$. The moment of inertia is assumed constant and taken at the upright state of the robot. We denote $\mathbf{r}_{B_i} : \mathbb{R}^3 \to \mathbb{R}^3$ as the position of foot i relative to the center of the mass in the body frame.

5.4.2 MPC Constraints

In this subsection we list the constraints that are included in the low-frequency kino-dynamic MPC controller.

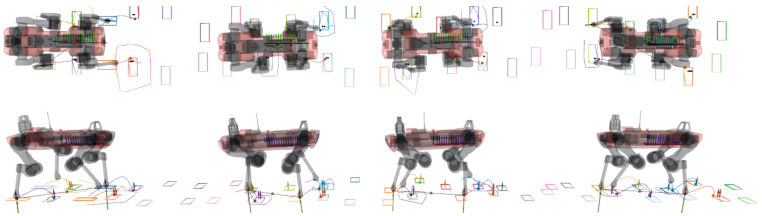


Figure 5.3: ANYmal traversing stepping-stones in simulation using the multi-layer CBF-MPC controller. The target foothold regions as well as the contracting barrier constraints are shown at snapshots in the motion. See the video in the supplementary material for the full motion.

5.4.2.1 Mode Constraints

The mode constraints capture the different modes of each leg at any given point in time. We assume that the mode sequence is a predefined function of time. The resulting mode-dependent constraints are

$$\begin{cases} \mathbf{v}_{W_i}(\mathbf{x}, \mathbf{u}) = \mathbf{0}, & \text{if } i \text{ is a stance leg,} \\ \mathbf{n}^\top \mathbf{v}_{W_i}(\mathbf{x}, \mathbf{u}) = c(t), \quad \boldsymbol{\lambda}_{B_i} = \mathbf{0}, & \text{if } i \text{ is a swing leg,} \end{cases}$$

where $\mathbf{v}_{W\,i}$ is the end-effector velocity in world frame. These constraints ensure that stance legs remain on the ground and a swing legs follow a predefined curve $c : \mathbb{R}_+ \to \mathbb{R}$ in the direction of the local surface normal $\mathbf{n} \in \mathbb{R}^3$ to avoid foot scuffing.

5.4.2.2 Friction Cone Constraints

The end-effector forces are constrained to lie in the friction cone, $\boldsymbol{\lambda}_{W_i} \in \mathcal{Q}(\mathbf{n}, \mu_c)$, defined by the surface normal $\mathbf{n}$ and friction coefficient $\mu_c = 0.7$. After resolving the contact forces in the local frame of the surface, given by $\mathbf{F} = [F_x, F_y, F_z]$, a second-order cone constraint is specified,

$$h_{\text{cone}} = \mu_c F_z - \sqrt{F_x^2 + F_y^2} \geq 0. \tag{5.18}$$

5.4.2.3 State-Only Foot Placement Constraints

When foot-placement is formulated as a state-only constraint (rather than encoded in a CBF), it is specified as the following inequality constraint on stance feet:

$$\mathbf{h}_i^t(\mathbf{x}) = \mathbf{A}_i \cdot \mathbf{p}_{W_i}(\mathbf{x}) + \mathbf{b}_i \geq \mathbf{0}, \tag{5.19}$$

where $\mathbf{A}_i \in \mathbb{R}^{p_i \times 3}$, $\mathbf{B}_i \in \mathbb{R}^{p_i}$, and $\mathbf{p}_{W_i} : \mathbb{R}^{24} \to \mathbb{R}^3$ is the position of foot i in the world frame. The matrix $\mathbf{A}_i$ and $\mathbf{B}_i$ project the position of foot i on to the target terrain and form a set of half-space constraints to ensure the foot lands within a desired target region. Instead of constraining the stance feet, a similar constraint can be placed on the swing feet with a constraint set that shrinks in time and converges to the desired foot placement region:

$$\mathbf{h}_i^w(\mathbf{x}, t) = \mathbf{A}_i \cdot \mathbf{p}_{W_i}(\mathbf{x}) + \mathbf{b}_i + s(t) \cdot \mathbf{1} \geq \mathbf{0}, \tag{5.20}$$

where $s : \mathbb{R}_+ \to \mathbb{R}_+$ converges to 0 as the t approaches the duration of the swing phase.

5.4.2.4 Barrier Foot Placement Constraints

When posed as a CBF constraint as in the proposed low-frequency Safe Kinematic MPC controller, the foot placement constraints are specified with constant $\gamma \in \mathbb{R}_{++}$ as:

$$\mathbf{h}_{e,i}^w(\mathbf{x}, \dot{\mathbf{q}}, t) = \dot{\mathbf{h}}_i^w(\mathbf{x}, t, \mathbf{u}) + \gamma \mathbf{h}_i^w(\mathbf{x}, t) \geq \mathbf{0}. \tag{5.21}$$

5.4.3 Whole-Body Tracking Control

The control signal $\mathbf{u}$ determined by the low-frequency MPC layer consists of contact forces and desired joint velocities. A high-frequency hierarchical inverse dynamics controller is used to convert the optimized MPC trajectory into torque commands (Dario Bellicoso et al., 2016). This WBC approach considers the full nonlinear rigid body dynamics of the system. At each priority, a Quadratic Program (QP) is solved in the null space of all higher priority tasks. Each task is a equality or inequality constraint that is affine in the generalized accelerations, torques, and contact forces. The CBF constraints, which are by design affine in the control torques, are therefore readily integrated into this framework. The full list of tasks is given in Table 5.1.

As described in Section 5.3, the following CBF constraint incorporating the dynamics can be included in the whole-body controller:

$$\dot{\mathbf{h}}_{e,i}^w(\mathbf{x}, \dot{\mathbf{q}}, t, \boldsymbol{\tau}) + \xi \mathbf{h}_{e,i}^w(\mathbf{x}, \dot{\mathbf{q}}, t) \geq \mathbf{0} \tag{5.22}$$

with $\xi \in \mathbb{R}_{++}$. Finally, the torque derived from the whole body controller, $\boldsymbol{\tau}_{\text{wbc}} \in \mathbb{R}^{12}$, is computed. To compensate for model uncertainty for swing legs

Table 5.1: WHOLE-BODY CONTROL TASK HIERARCHY.

Priority	Type	Task
0	$=$	Floating base equations of motion.
	$\geq$	Torque limits.
	$\geq$	Friction cone constraint.
	$=$	No motion at the contact points.
	$\geq$	**Control barrier constraints.**
1	$=$	Torso linear and angular acceleration.
	$=$	Swing leg motion tracking.
2	$=$	Contact force tracking.

(on hardware, not in simulation), the integral of joint acceleration error with gain $K \in \mathbb{R}_{++}$ is added to the torque applied to the system:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\text{wbc}} - K \int_{t_0^{sw}}^{t} (\ddot{\mathbf{q}} - \ddot{\mathbf{q}}_{\text{wbc}}) \, \mathrm{d}t \tag{5.23}$$

While this modification implies $\boldsymbol{\tau}$ may not satisfy the CBF condition in (5.22), we note that $\boldsymbol{\tau}_{\text{wbc}}$ may not satisfy (5.22) in the presence of model uncertainty. To achieve safe behavior in practice, it is necessary to balance the choice of safe inputs with model uncertainty.

5.4.4 User Commands & Terrain Selection

User commanded twists and a desired gait pattern are provided to the robot via joystick and extrapolated to a state reference signal $\mathbf{x}_{\text{ref}}(t)$. The reference input $\mathbf{u}_{\text{ref}}(t)$ is constructed by equally distributing the weight over all contact feet. The MPC cost function is a frequency dependent quadratic cost around the reference trajectories to promote smooth optimal inputs (Grandia et al., 2019a).

We assume that a segmented terrain model with each segment described by a planar boundary and a surface normal is available. For each contact phase within the MPC horizon, the terrain segment is selected that is closest to the reference end-effector position determined by $\mathbf{x}_{\text{ref}}(t)$, evaluated at the middle of the stance phase. A convex polygon is fit to the selected terrain, starting from the reference end-effector position projected onto the segment boundary. This polygon, together with the surface normal, define the half spaces for the constraints in (5.19) and (5.20).

5.5 Results

We evaluate the controller proposed in Section 5.4 in simulation on a classical stepping-stones scenario as shown in Fig. 5.3. The stones are configured with a pattern of 0.5 m width and 0.35 m longitudinal spacing, with random displacements up to 10, 15, and 5 cm, in longitudinal, lateral, and vertical direction respectively. The controller is commanded to perform a trotting gait with a forward velocity of 0.25 m/s, and commanded to stop on the final stone. We compare our proposed controller, numbered V, against four alternative formulations and report results in Table 5.2.

Table 5.2: SIMULATION RESULTS

	I	II	III	IV	V
MPC constr.	None	CBF	State	State	CBF
WBC constr.	CBF	None	None	CBF	CBF
num. steps	28	140	140	140	140
num. missteps	5	6	5	**0**	**0**
avg. misstep [mm]	1.4	2.5	4.3	**-**	**-**
total swing time [s]	11.0	49.0	48.6	48.4	48.6
$\mathbf{h}_i^w < 0$ time [s]	2.4	2.3	15.3	2.6	**0.4**
$\mathbf{h}_{e,i}^w < 0$ time [s]	3.3	5.4	15.6	3.7	**0.8**

As seen in the supplementary video [1], the controller with no foot placement constraints in the MPC controller and a CBF constraint in the high-frequency controller (denoted CBF-QP, and the closest to the related work (Nguyen et al., 2016)) is able to enforce safety for a number of steps, but quickly destabilizes. The absence of information on the safety constraint in the MPC layer results in an abrupt and strong correction for safety by the high-frequency CBF. This approach work well only when the stepping-stones are placed close to the nominal gait of the robot, but it fails in this more challenging scenario.

The second and third controllers include foot placement constraints in the MPC controller, but not in the high-frequency controller. In the second controller the constraints are implemented as CBFs through (5.21) and in the third controller they are implemented as state constraints through (5.19). Both of these controllers are able to successfully traverse the length of the stepping-stones scenario. We see that the controllers exhibit similar numbers of missteps, but the MPC controller with CBFs has smaller average misstep size.

[1] `https://youtu.be/TCDIirXfByE`

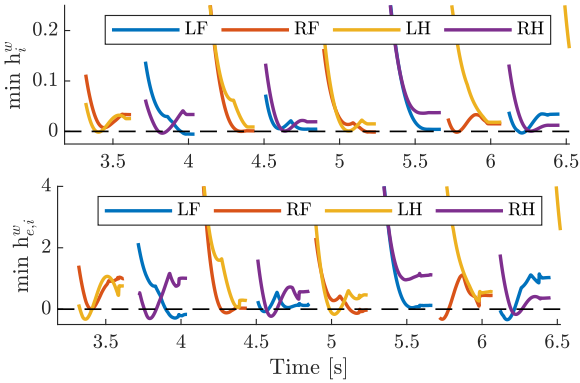


Figure 5.4: The minimum values of $\mathbf{h}_i^w$ and $\mathbf{h}_{e,i}^w$ per leg during the stepping-stones simulation with the CBF only at MPC level.

The fourth and fifth controller enforce the high frequency CBFs (5.22) and contain either state constraints (5.19) or CBFs (5.21) in the MPC formulation. Both controllers complete the scenario without missteps. However, the proposed controller shows the least amount of time violating the barrier conditions. The difference can be explained through the results in Fig. 5.6. Because the MPC with state constraints (top) is not aware of the CBF condition, it plans for a trajectory that violates these constraints during the swing phase. During execution, the high-frequency tracking controller strictly enforces the CBF, resulting in a deviation from the MPC plan. Such abrupt deviations can cause problems, for example when operating close to kinematic limits. Consistently enforcing the CBF condition removes this mismatch (bottom).

The simulation experiments indicate that including CBF constraints in the high-frequency controller leads to safer behavior, and that including terrain constraints in the MPC controller prevents the high-frequency CBF from destabilizing the gait. Finally, enforcing CBF constraints in both layers of the hierarchy prevents an inconsistency that results in large deviations from the optimal solution determined by the MPC layer. The values of $\mathbf{h}_i^w$ and $\mathbf{h}_{e,i}^w$ for the controller with CBFs only in the MPC and for with the CBFs in both WBC & MPC can be seen in Fig. 5.4 and 5.5. The controller with CBFs at both levels has smaller violations of constraints (5.20) and (5.21).

We evaluate the efficacy of this method experimentally on the ANYmal robotic platform. All computation runs on a single onboard PC (Intel i7-8850H, 2.6 GHz, hexa-core 64-bit) with the MPC solver running asynchronously at 30 Hz and the whole-body QP tracking controller running at 400 Hz.

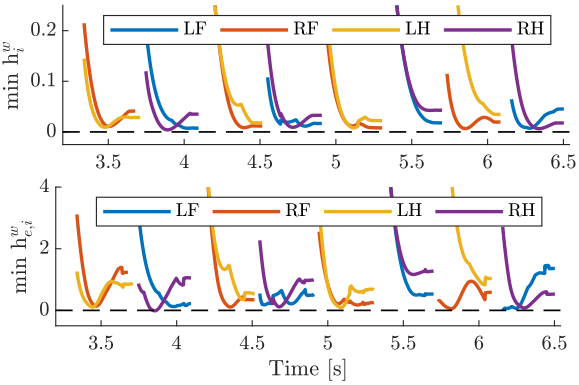


Figure 5.5: The minimum values of $\mathbf{h}_i^w$ and $\mathbf{h}_{e,i}^w$ per leg during the stepping-stones simulation with the CBF in both WBC & MPC.

The robot is initialized on pre-mapped terrain and receives external base twist and gait commands. The size of the segmented regions are decreased by 5 cm with respect to the real boundary to provide a margin for state estimation errors. In the supplementary video we visualize the internal state of the controller. For legs that are in swing, a projection of the barrier constraint in (5.20) onto the terrain is plotted. This barrier constraint shrinks over time and converges to the selected target foothold region at foot contact. Furthermore, it can be seen how the foothold target is large when stepping onto the wooden pallet. This shows that the proposed method can seamlessly transition between rough and flat terrain, restricting the motion only when necessary for safe foot placement. The values of $\mathbf{h}_i^w$ and $\mathbf{h}_{e,i}^w$ for several steps can be seen in Fig. 5.7. Both constraints are rarely violated, which confirms that the safety constraints are successfully transferred to hardware.

5.6 Conclusion

We proposed a multi-layered control framework that combines CBFs with MPC. Simulation experiments show that enforcing CBF constraints on both the MPC and QP tracking layer outperforms variants where they are enforced at only one of the layers. Additionally, we validated the viability of the approach on hardware by demonstrating dynamic locomotion on stepping-stones with safety constraints. Future work includes developing a perception pipeline to automatically perform terrain-based segmentation from sensor data and studying the theoretical properties of the proposed controller.

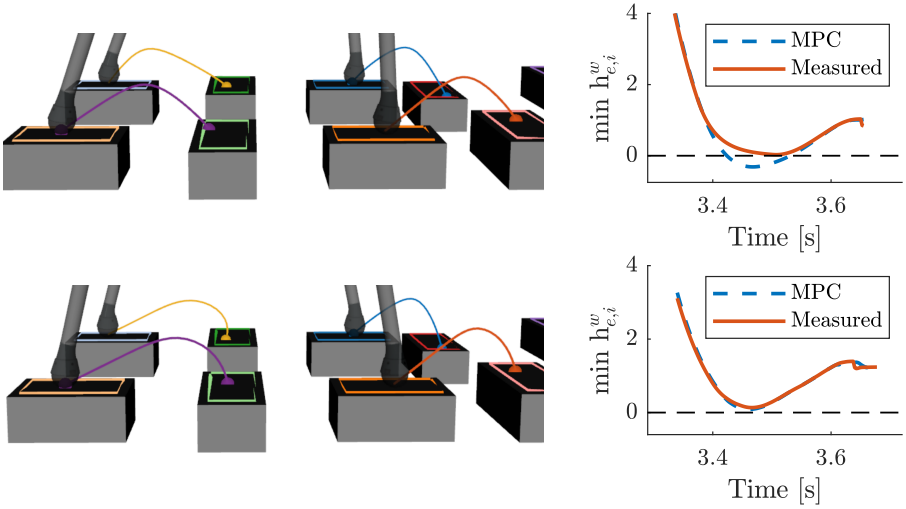


Figure 5.6: Visualization of the planned MPC trajectories for different constraint formulations. **Top:** MPC with state constraints on the touchdown location (controller IV). **Bottom:** MPC with CBF constraints (controller V). The plots on the right show the planned and measured values of $\mathbf{h}^w_{e,i}$ for the right front foot, with deviations from the MPC optimal trajectory occurring when CBF constraints are absent from the MPC formulation.

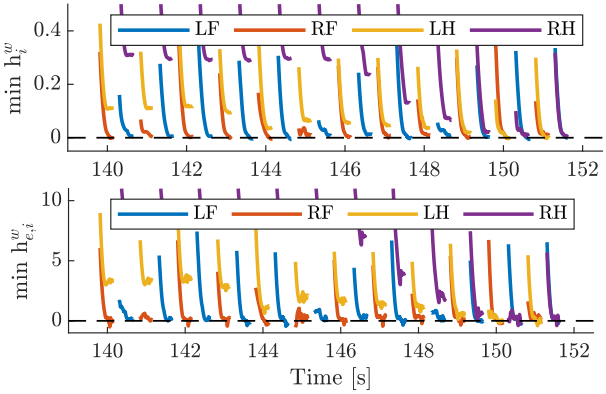


Figure 5.7: The minimum values of $\mathbf{h}^w_i$ and $\mathbf{h}^w_{e,i}$ per leg during the stepping-stones hardware experiment for the proposed controller with CBF constraints in WBC & MPC.

6

Perceptive Locomotion through Nonlinear Model Predictive Control

Grandia, R., Jenelten, F., Yang, S., Farshidian, F., and Hutter, M. (2022). "Perceptive Locomotion through Nonlinear Model Predictive Control". *(submitted to) IEEE Transactions on Robotics*

DOI: 10.48550/ARXIV.2208.08373

Video: https://youtu.be/v6MhPl2ICsc

Dynamic locomotion in rough terrain requires accurate foot placement, collision avoidance, and planning of the underactuated dynamics of the system. Reliably optimizing for such motions and interactions in the presence of imperfect and often incomplete perceptive information is challenging. We propose to mitigate the numerical challenges posed by the terrain by extracting a sequence of convex inequality constraints as local approximations of foothold feasibility and embedding them directly into an online model predictive controller. Based on this idea, we present a complete perception, planning, and control pipeline, that can optimize motions for all degrees of freedom of the robot in real-time. Steppability classification, plane segmentation, and a signed distance field are precomputed per elevation map to minimize the computational effort during the model predictive control optimization. We validate the proposed method in scenarios with gaps, slopes, and stepping stones in simulation and experimentally on the ANYmal quadruped platform, resulting in state-of-the-art dynamic climbing.

6.1 Introduction

Inspired by nature, the field of legged robotics aims to enable the deployment of autonomous systems in rough and complex environments. Indeed, during the recent DARPA subterranean challenge, legged robots were widely adopted, and highly successful (Bouman et al., 2020; Tranzatto et al., 2021). Still, complex terrains that require precise foot placements, e.g., negative obstacles and stepping stones as shown in Fig. 6.1, remain difficult.

A key challenge lies in the fact that both the terrain and the system dynamics impose constraints on contact location, force, and timing. When taking a model-based approach, mature methods exist for perceptive locomotion with a slow, static gait (Belter, Labecki, and Skrzypczynski, 2016; Fankhauser et al., 2018; Griffin et al., 2019; Kalakrishnan et al., 2010; Mastalli et al., 2020b) and for blind, dynamic locomotion that assumes flat terrain (Bellicoso et al., 2018a; Bledt, Wensing, and Kim, 2017; Di Carlo et al., 2018). Learning-based controllers have recently shown the ability to generalize blind locomotion to challenging terrain with incredible robustness (Lee et al., 2020; Miki et al., 2022a; Siekmann et al., 2021). Still, tightly integrating perception to achieve coordinated and precise foot placement remains an active research problem.

In an effort to extend dynamic locomotion to uneven terrain, several methods have been proposed to augment foothold selection algorithms with perceptive information (Jenelten et al., 2020; Kim et al., 2020; Villarreal et al., 2020). These approaches build on a strict hierarchy of first selecting footholds and optimizing torso motion afterward. This decomposition reduces the computational complexity but relies on hand-crafted coordination between the two modules. Additionally, separating the legs from the torso optimization makes it difficult to consider kinematic limits and collision avoidance between limbs and terrain.

Trajectory optimization with full robot dynamics has shown impressive results in simulation (Dai, Valenzuela, and Tedrake, 2014; Mordatch, Todorov, and Popović, 2012; Winkler et al., 2018) and removes the need for engineered torso-foot coordination. Complex motions can be automatically discovered by including the entire terrain in the optimization. However, computation times are often too long for online deployment. Additionally, due to the non-convexity, non-linearity, and discontinuity introduced by optimizing over arbitrary terrain, these methods can get stuck in poor local minima. Dedicated work on providing an initial guess is needed to find feasible motions reliably (Melon et al., 2020).

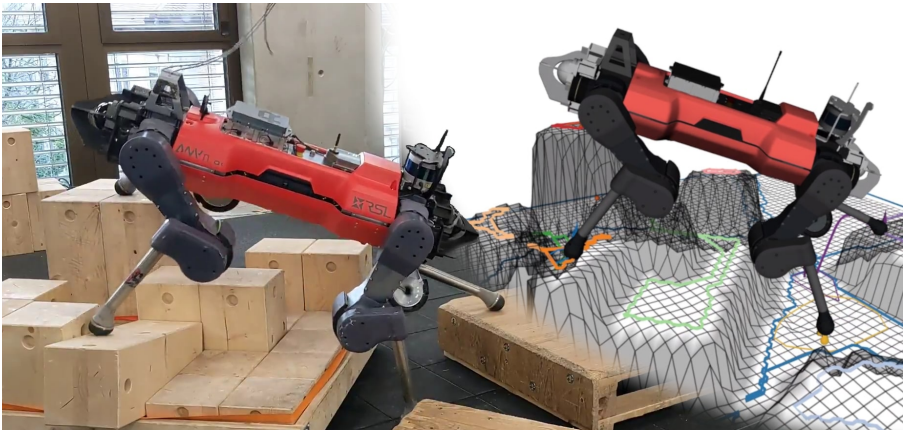


Figure 6.1: ANYmal walking on uneven stepping stones. In the shown configuration, the top foothold is 60 cm above the lowest foothold. The top right visualizes the internal terrain representation used by the controller.

This work presents a planning and control framework that optimizes over all degrees of freedom of the robot, considers collision avoidance with the terrain, and enables complex dynamic maneuvers in rough terrain. The method is centered around nonlinear Model Predictive Control (MPC) with a multiple-shooting discretization (Bock and Plitt, 1984; Rawlings, Mayne, and Diehl, 2017). However, in contrast to the aforementioned work, where the full terrain is integrated into the optimization, we get a handle on the numerical difficulty introduced by the terrain by exposing the terrain as a series of geometric primitives that approximate the local terrain. In this case, we use convex polygons as foot placement constraints, but different shapes can be used as long as they lead to well-posed constraints in the optimization. Additionally, a signed distance field (SDF) is used for collision avoidance. We empirically demonstrate that such a strategy is an excellent trade-off between giving freedom to the optimization to discover complex motions and the reliability with which we can solve the formulated problem.

6.1.1 Contributions

We present a novel approach to locomotion in challenging terrain where perceptive information needs to be considered, and nontrivial motions are required. This is achieved by jointly optimizing over all degrees of freedom of the robot instead of decoupling base and feet motions in a hierarchical approach. The

complete perception, planning, and control pipeline contains the following contributions:

- Tractable integration of perceptive information into motion optimization through a sequence of geometric primitives. In this case, convex polygons are used to describe local foothold constraints.
- A model formulation that allows collision avoidance of limbs with the terrain and considers configuration-dependent inertia and center of mass location.
- Online segmentation of the elevation map into steppable regions, and efficient precomputation of a signed distance field.
- A detailed description of numerical techniques that enable fast and reliable online solutions to the nonlinear problem. The implementation is publicly available[1] as part of the MPC toolbox OCS2 (*OCS2: An open source library for Optimal Control of Switched Systems*).

6.1.2 Outline

An overview of the proposed method is given in Fig. 6.2. The perception pipeline at the top of the diagram runs at 20 Hz and is based on an elevation map constructed from pointcloud information. For each map update, classification, segmentation, and other precomputation are performed to prepare for the high number of perceptive queries during motion optimization. At the core of the framework, we use nonlinear MPC at 100 Hz to plan a motion for all degrees of freedom and bring together user input, perceptive information, and the measured state of the robot. Finally, state estimation, whole-body torque control, and reactive behaviors are executed at a rate of 400 Hz.

After a review of related work in section 6.2, this paper is structured similarly to Fig. 6.2. First, we present the perception pipeline in section 6.3. Afterward, the formulated optimal control problem and corresponding numerical optimization strategy are discussed in sections 6.4 & 6.5. We introduce the motion execution layer in section 6.6. The resulting method is evaluated on the quadrupedal robot ANYmal (Hutter et al., 2016) (see Fig. 6.1) in section 6.7, and concluded with section 6.8.

[1] https://github.com/leggedrobotics/ocs2/tree/main/ocs2_sqp

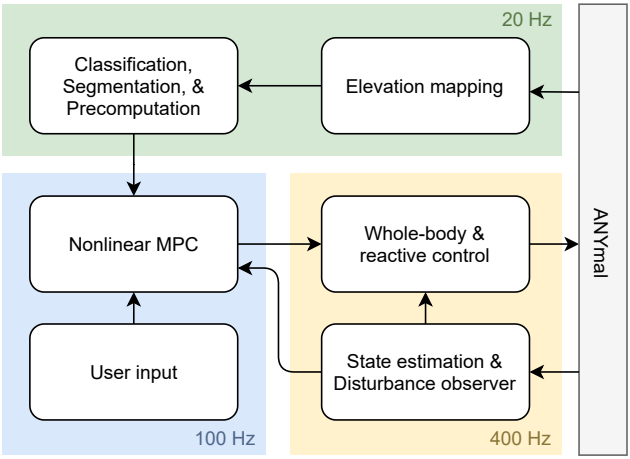


Figure 6.2: Schematic overview of the proposed method together with the update rate of each component.

6.2 Related Work

6.2.1 Decomposing Locomotion

When assuming a quasi-static gait with a predetermined stepping sequence, the planning problem on rough terrain can be simplified and decomposed into individual contact transitions, as demonstrated in the early work on *LittleDog* (Kalakrishnan et al., 2010; Kolter, Rodgers, and Ng, 2008). In a one-step-ahead fashion, one can check the next foothold for kinematic feasibility, feasibility w.r.t. the terrain, and the existence of a statically stable transition. This problem can be efficiently solved by sampling and checking candidate footholds (Tonneau et al., 2018). Afterward, a collision-free swing leg trajectory to the desired foothold can be generated with CHOMP (Zucker et al., 2013) based on an SDF. Fully onboard perception and control with such an approach were achieved by Fankhauser et al. (Fankhauser et al., 2018). Instead of one-step-ahead planning, an RRT graph can be built to plan further ahead (Belter, Labecki, and Skrzypczynski, 2016). Sampling over templated foothold transitions achieves similar results (Mastalli et al., 2015; Mastalli et al., 2020b).

In this work, we turn our attention to dynamic gaits, where statically stable transitions between contact configurations are not available. In model-based

approaches to dynamic, perceptive locomotion, a distinction can be made between methods where the footholds locations are determined separately from the torso and those where the foothold locations and torso motions are jointly optimized.

Several methods in which footholds are selected before optimizing the torso motions, initially designed for flat terrain, have been adapted to traverse rough terrain (Bajracharya et al., 2013; Bazeille et al., 2014). These methods typically employ some form of Raibert heuristic (Raibert, 1986) to select the next foothold and adapt it based on perceptive information such as a traversability estimate (Wermelinger et al., 2016). The work of Bellicoso et al. (Bellicoso et al., 2018a) was extended by including a batch search for feasible footholds based on a given terrain map and foothold scoring (Jenelten et al., 2020). Similarly, in (Kim et al., 2020), the foot placement is adapted based on visual information resulting in dynamic trotting and jumping motions. In (Magana et al., 2019), the authors proposed to train a convolutional neural network (CNN) to speed up the online evaluation of such a foothold adaptation pipeline. This CNN was combined with the MPC strategy in (Di Carlo et al., 2018) to achieve perceptive locomotion in simulation (Villarreal et al., 2020). In (Gangapurwala et al., 2022) and (Yu et al., 2021), a Reinforcement Learning (RL) policy has replaced the heuristic foothold selection.

However, since foothold locations are chosen before optimizing the torso motion, their effect on dynamic stability and kinematic feasibility is not directly considered, requiring additional heuristics to coordinate feet and torso motions to satisfy whole-body kinematics and dynamics. Moreover, it becomes hard to consider collisions of the leg with the terrain because the foothold is already fixed. In our approach, we use the same heuristics to find a suitable nominal foothold in the terrain. However, instead of fixing the foothold to that particular location, a region is extracted around the heuristic in which the foothold is allowed to be optimized.

The benefit of jointly optimizing torso and leg motions has been demonstrated in the field of trajectory optimization. One of the first demonstrations of simultaneous optimization of foot placement and a zero-moment point (ZMP) (Vukobratović and Borovac, 2004) trajectory was achieved by adding 2D foot locations as decision variables to an MPC algorithm (Herdt et al., 2010). More recently, Kinodynamic (Farshidian et al., 2017a), Centroidal (Orin, Goswami, and Lee, 2013; Sleiman et al., 2021), and full dynamics models (Herzog, Schaal, and Righetti, 2016; Pardo et al., 2017) have been used for simultaneous optimization of 3D foot locations and body motion. Alternatively, a single rigid

body dynamics (SRBD) model can be extended with decision variables for Cartesian foothold locations (Winkler et al., 2018). Real-time capable methods have been proposed with the specification of leg motions on position (Bledt, Wensing, and Kim, 2017), velocity (Farshidian et al., 2017b), or acceleration level (Neunert et al., 2018). One challenge of this line of work is the computational complexity arising from the high dimensional models, already in the case of locomotion on flat terrain. Our method also uses a high-dimensional model and falls in this category. A key consideration when extending the formulations with perceptive information has thus been to keep computation within real-time constraints.

Finally, several methods exist that additionally optimize gait timings or even the contact sequence together with the whole-body motion. This can be achieved through complementarity constraints (Dai, Valenzuela, and Tedrake, 2014; Mordatch, Todorov, and Popović, 2012; Posa, Cantu, and Tedrake, 2014), mixed-integer programming(Aceituno-Cabezas et al., 2018; Marcucci et al., 2017), or by explicitly integrating contact models into the optimization (Carius et al., 2018; Neunert et al., 2018). Alternatively, the duration of each contact phase can be included as a decision variable (Ponton et al., 2018; Winkler et al., 2018) or found through bilevel optimization (Farshidian et al., 2017c; Seyde et al., 2019).

6.2.2 Terrain Representation

The use of an elevation map has a long-standing history in the field of legged robotics (Herbert et al., 1989), and it is still an integral part of many perceptive locomotion controllers today. Approaches where footholds are selected based on a local search or sampling-based algorithm can directly operate on such a structure. However, more work is needed when integrating the terrain into a gradient-based optimization.

Winkler et al. (Winkler et al., 2018) uses an elevation map for both foot placement and collision avoidance. The splines representing the foot motion are constrained to start and end on the terrain with equality constraints. An inequality constraint is used to avoid the terrain in the middle of the swing phase. Ignoring the discontinuity and non-convexity from the terrain makes this approach prone to poor local minima, motivating specialized initialization schemes (Melon et al., 2020) for this framework.

In (Jenelten et al., 2021), a graduated optimization scheme is used, where a first optimization is carried out over a smoothened version of the terrain. The

solution of this first optimization is then used to initialize an optimization over the actual elevation map. In a similar spirit, Mordatch (Mordatch, Todorov, and Popović, 2012) considers a general 3D environment and uses a soft-min operator to smoothen the closest point computation. A continuation scheme is used to gradually increase the difficulty of the problem over consecutive optimizations.

Deits et al. (Deits and Tedrake, 2014) describe a planning approach over rough terrain based on mixed-integer quadratic programming (MIQP). Similar to (Griffin et al., 2019), convex safe regions are extracted from the terrain, and footstep assignment to a region is formulated as a discrete decision. The foothold optimization is simplified because only convex, safe regions are considered during planning. We follow the same philosophy of presenting the terrain as a convex region to the optimization. However, we remove the mixed-integer aspect by pre-selecting the convex region. The benefits are two-fold: First, we do not require a global convex decomposition of the terrain, which is a hard problem in general (Bertrand et al., 2020), and instead, only produce a local convex region centered around a nominal foothold. Second, the MIQP approach does not allow for nonlinear costs and dynamics, which limits the range of motions that can be expressed. We first explored the proposed terrain representation as part of our previous work (Grandia et al., 2021), but relied on offline mapping, manual terrain segmentation, and did not yet consider terrain collisions.

6.2.3 Motion Optimization

For trajectory optimization, large-scale optimization software like SNOPT (Gill, Murray, and Saunders, 2005) and IPOPT (Wächter and Biegler, 2006) are popular. They are the workhorse for offline trajectory optimization in the work of Winkler (Winkler et al., 2018), Dai (Dai, Valenzuela, and Tedrake, 2014), Mordatch (Mordatch, Todorov, and Popović, 2012), Posa (Posa, Cantu, and Tedrake, 2014), and Pardo (Pardo et al., 2017). These works show a great range of motions in simulation, but it typically takes minutes to hours to find a solution.

A different line of work uses specialized solvers that exploit the sparsity that arises from a sequential decision making process. Several variants of Differential Dynamic Programming (DDP) (Jacobson and Mayne, 1970) have been proposed in the context of robotic motion optimization, e.g., iLQR (Howell, Jackson, and Manchester, 2019; Tassa, Erez, and Todorov, 2012), SLQ (Farshidian et al., 2017a), and FDDP (Mastalli et al., 2020a).

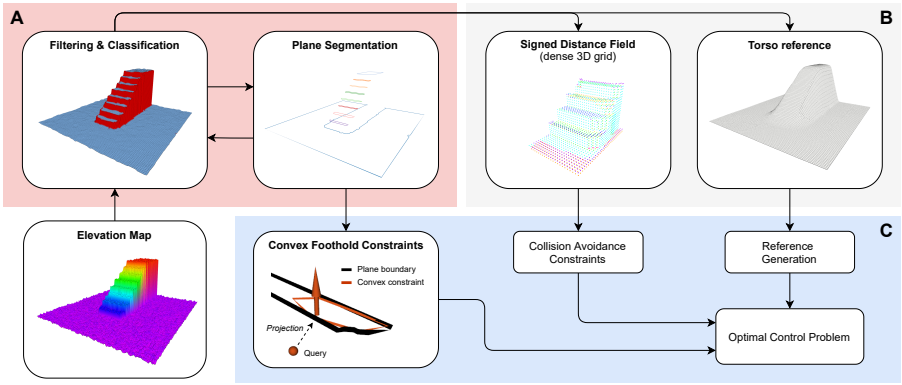


Figure 6.3: Perception pipeline overview. (**A**) The elevation map is filtered and classified into steppable and non-steppable cells. All steppable areas are segmented into planes. After segmentation, the steppablity classification is refined. (**B**) A signed distance field and torso reference layer are precomputed to reduce the required computation time during optimization. **(C)** Convex foothold constraints are obtained from the plane segmentation. The signed distance field enables collision avoidance, and the torso reference is used to generate height and orientation references.

With a slightly different view on the problem, the field of (nonlinear) model predictive control (Mayne, 2014; Rawlings, Mayne, and Diehl, 2017) has specialized in solving successive optimal control problems under real-time constraints. See (Kouzoupis et al., 2018) for a comparison of state-of-the-art quadratic programming (QP) solvers that form the core of second-order optimization approaches to the nonlinear problem. For time-critical applications, the real-time iteration scheme can be used to trade optimality for lower computational demands (Diehl, Bock, and Schlöder, 2005): In a Sequential Quadratic Programming (SQP) approach to the nonlinear problem, at each control instance, only a single QP optimization step is performed.

The current work was initially built on top of a solver in the first category (Farshidian et al., 2017a). However, a significant risk in classical DDP-based approaches is the need to perform a nonlinear system rollout along the entire horizon. Despite the use of a feedback policy, these forward rollouts can diverge, especially in the presence of underactuated dynamics. This same observation motivated Mastalli et al. to design FDDP to maintain *gaps* between shorter rollouts, resulting in a formulation that is equivalent to direct multiple-shooting formulations with only equality constraints (Bock and Plitt, 1984; Mastalli et al., 2020a).

We directly follow the multiple-shooting approach with a real-time iteration scheme and leverage the efficient structure exploiting QP solver HPIPM (Frison and Diehl, 2020). However, as also mentioned in (Mastalli et al., 2020a), one difficulty is posed in deciding a stepsize for nonlinear problems, where one now has to monitor both the violation of the system dynamics and minimization of the cost function. To prevent an arbitrary trade-off through a merit function, we suggest using a filter-based line-search instead (Fletcher and Leyffer, 2002), which allows a step to be accepted if it reduces either the objective function or the constraint violation.

6.3 Terrain Perception and Segmentation

An overview of the perception pipeline and its relation to the MPC controller is provided in Fig. 6.3. The pipeline can be divided into three parts: (**A**) steppability classification and segmentation, (**B**) precomputation of the SDF and torso reference, and (**C**) integration into the optimal control problem.

The elevation map, represented as a 2.5D grid (Fankhauser and Hutter, 2016) with a 4 cm resolution is provided by the GPU based implementation introduced in (Miki et al., 2022b). The subsequent map processing presented in this work runs on the CPU and is made available as part of that same open-source library. Both (**A**) and (**B**) are computed once per map and run asynchronously to the motion optimization in (**C**).

6.3.1 Filtering & Classification

The provided elevation map contains empty cells in occluded areas. As a first step, we perform *inpainting* by filling each cell with the minimum value found along the occlusion border. Afterwards, a median filter is used to reduce noise and outliers in the map.

Steppablity classification is performed by thresholding the local surface inclination and the local roughness estimated through the standard deviation (Chilian and Hirschmüller, 2009). Both quantities can be computed with a single pass through the considered neighbourhood of size N:

$$\boldsymbol{\mu} = \frac{1}{N}\sum_i \mathbf{c}_i, \quad \mathbf{S} = \frac{1}{N}\sum_i \mathbf{c}_i\mathbf{c}_i^\top, \quad \boldsymbol{\Sigma} = \mathbf{S} - \boldsymbol{\mu}\boldsymbol{\mu}^\top, \tag{6.1}$$

where $\boldsymbol{\mu}$ and $\mathbf{S}$ are the first and second moment, and $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ is the positive semi-definite covariance matrix of the cell positions $\mathbf{c}_i$. The variance in

normal direction, σ_n^2, is then the smallest eigenvalue of $\mathbf{\Sigma}$, and the surface normal, $\mathbf{n}$, is the corresponding eigenvector. For steppability classification we use a neighbourhood of $N = 9$, and set a threshold of 2 cm on the standard deviation in normal direction and a maximum inclination of 35°, resulting in the following classification:

$$\text{steppability} = \begin{cases} 1 & \text{if } \sigma_n \leq 0.02, \text{ and } n_z \geq 0.82, \\ 0 & \text{otherwise,} \end{cases} \tag{6.2}$$

where n_z denotes the z-coordinate of the surface normal.

6.3.2 Plane Segmentation

After the initial classification, the plane segmentation starts by identifying continuous regions with the help of a connected component labelling (Wu, Otoo, and Suzuki, 2009). For each connected region of cells, we compute again the covariance as in (6.1), where N is now the number of cells in the connected region, and accept the region as a plane based on the following criteria:

$$\text{planarity} = \begin{cases} 1 & \text{if } \sigma_n \leq 0.025, n_z \geq 0.87, \text{ and } N \geq 4 \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

Notice that here we loosen the bound on the standard deviation to 2.5 cm, tighten the bound on the inclination to 30°, and add the constraint that at least 4 cells form a region. If the planarity condition is met, the surface normal and mean of the points define the plane.

If a region fails the planarity condition, we trigger RANSAC (Schnabel, Wahl, and Klein, 2007) on that subset of the data. The same criteria in (6.3) are used to find smaller planes within the connected region. After the algorithm terminates, all cells that have not been included in any plane have their steppability updated and set to 0.

At this point, we have a set of plane parameters with connected regions of the map assigned to them. For each of these regions, we now extract a 2D contour from the elevation map (Suzuki and be, 1985), and project it along the z-axis to the plane to define the boundary in the frame of the plane. It is important to consider that regions can have holes, so the boundary is defined as an outer polygon together with a set of polygons that trace the enclosed holes. Finally, if the particular region allows, we shrink the boundary inwards

(and holes outwards) to provide a safety margin. If the inscribed area is not large enough, the plane boundary is accepted without margin. In this way we obtain a margin where there is enough space to do so, but at the same time we do not reject small stepping stones, which might be crucial in certain scenarios.

6.3.3 Signed Distance Field

Before computing the SDF, we take advantage of the classification between terrain that will be potentially stepped on and terrain that will not be stepped on. To all cells that are non-steppable, we add a vertical margin of 2 cm, and dilate the elevation by one cell. The latter effectively horizontally inflates all non-steppable areas by the map resolution. This procedure corrects for the problem that edges tend to be underestimated in the provided elevation map.

We use a dense 3D voxel grid, where each voxel contains the value and 3D gradient. The previous motion plan is used to determine the 3D volume where distance information is needed. This volume is a bounding box that contains all collision bodies of the last available plan with a margin of 25 cm. This way, the size and shape of the SDF grid dynamically scales with the motion that is executed. Storing both value and derivative as proposed in (Pankert and Hutter, 2020) allows for efficient interpolation during optimization. However, in contrast to (Pankert and Hutter, 2020), where values and gradients are cached after the first call, we opt to precompute the full voxel grid to reduce the computation time during optimization as much as possible.

This is possible by taking advantage of the extra structure that the 2.5D representation provides. A detailed description of how the SDF can be efficiently computed from an elevation map is given in Appendix A, section 6.9.

6.3.4 Torso Reference Map

With user input defined as horizontal velocity and an angular rate along the z-direction, it is the responsibility of the controller to decide on the height and orientation of the torso. We would like the torso pose to be positioned in such a way that suitable footholds are in reach for all of the feet. We therefore create a layer that is a smooth interpolation of all steppable regions as described in (Jenelten et al., 2021). The use of this layer to generate a torso height and orientation reference is presented in section 6.4.5.

6.4 Motion Planning

In this section, we describe the nonlinear MPC formulation. In particular, we set out to define all components in the following nonlinear optimal control problem:

$$\begin{aligned}
&\underset{\mathbf{u}(\cdot)}{\text{minimize}} && \Phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t), t)\ \mathrm{d}t, && \text{(6.4a)}\\
&\text{subject to:} && \mathbf{x}(0) = \mathbf{x}_0, && \text{(6.4b)}\\
&&& \dot{\mathbf{x}} = \mathbf{f}^c(\mathbf{x}, \mathbf{u}, t), && \text{(6.4c)}\\
&&& \mathbf{g}(\mathbf{x}, \mathbf{u}, t) = \mathbf{0}, && \text{(6.4d)}
\end{aligned}$$

where $\mathbf{x}(t)$ is the state and $\mathbf{u}(t)$ is the input at time t. The term $L(\cdot)$ is a time-varying running cost, and $\Phi(\cdot)$ is the cost at the terminal state $\mathbf{x}(T)$. The goal is to find a control signal that minimizes this cost subject to the initial condition, $\mathbf{x}_0$, system dynamics, $\mathbf{f}^c(\cdot)$, and equality constraints, $\mathbf{g}(\cdot)$. Inequality constraints are all handled through penalty functions and will be defined as part of the cost function in section 6.4.6.

6.4.1 Robot Definition

We define the generalized coordinates and velocities as:

$$\mathbf{q} = \left[\boldsymbol{\theta}_B^\top, \mathbf{p}_B^\top, \mathbf{q}_j^\top\right]^\top, \quad \dot{\mathbf{q}} = \left[\boldsymbol{\omega}_B^\top, \mathbf{v}_B^\top, \dot{\mathbf{q}}_j^\top\right]^\top, \tag{6.5}$$

where $\boldsymbol{\theta}_B \in \mathbb{R}^3$ is the orientation of the base frame, $\mathcal{F}_B$, in Euler angles, $\mathbf{p}_B \in \mathbb{R}^3$ is the position of the base in the world frame, $\mathcal{F}_W$. $\boldsymbol{\omega}_B \in \mathbb{R}^3$ and $\mathbf{v}_B \in \mathbb{R}^3$ are the angular rate and linear velocity of the base in the body frame $\mathcal{F}_B$. Joint positions and velocities are given by $\mathbf{q}_j \in \mathbb{R}^{12}$ and $\dot{\mathbf{q}}_j \in \mathbb{R}^{12}$. The collection of all contact forces is denoted by $\boldsymbol{\lambda} \in \mathbb{R}^{12}$. When referring to these quantities per leg, we will use a subscript i, e.g. $\mathbf{q}_i \in \mathbb{R}^3$ or $\boldsymbol{\lambda}_i \in \mathbb{R}^3$. All subscripts for legs in contact are contained in the set $\mathcal{C}$. A graphical illustration of the robot together with the defined coordinate frames is provided in Fig. 6.4.

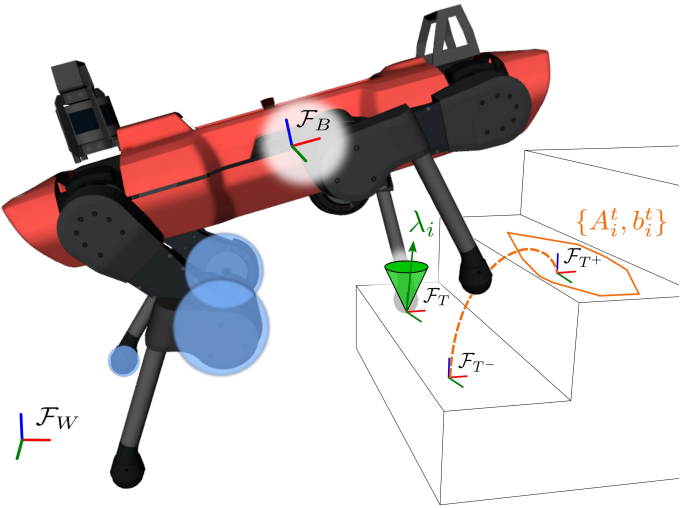


Figure 6.4: Overview of the coordinates frames and constraints used in the definition of the MPC problem. On the front left foot, a friction cone is shown, defined in the terrain frame $\mathcal{F}_T$. On the right front foot, a swing reference trajectory is drawn between the liftoff frame $\mathcal{F}_{T^-}$ and touchdown frame $\mathcal{F}_{T^+}$. Foot placement constraints are defined as a set of half-spaces in the touchdown frame. Stance legs have collision bodies at the knee, as illustrated on the right hind leg, while swing legs have collision bodies on both the foot and the knee, as shown on the left hind leg.

6.4.2 Torso Dynamics

To derive the torso dynamics used in this work, consider the full rigid body dynamics of the robot,

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^\top \boldsymbol{\tau} + \boldsymbol{\tau}^{\text{dist}} + \sum_{i \in \mathcal{C}} \mathbf{J}_i^\top(\mathbf{q}) \boldsymbol{\lambda}_i, \tag{6.6}$$

with inertia matrix $\mathbf{M} : \mathbb{R}^{18} \to \mathbb{R}^{18\times 18}$, generalized accelerations $\ddot{\mathbf{q}} \in \mathbb{R}^{18}$, and nonlinear terms $\mathbf{n} : \mathbb{R}^{18} \times \mathbb{R}^{18} \to \mathbb{R}^{18}$ on the left hand side. The right hand contains the selection matrix $\mathbf{S} = [\mathbf{0}_{12\times 6}, \mathbf{I}_{12\times 12}] \in \mathbb{R}^{12\times 18}$, actuation torques $\boldsymbol{\tau} \in \mathbb{R}^{12}$, disturbance forces $\boldsymbol{\tau}^{\text{dist}} \in \mathbb{R}^{18}$, contact Jacobians $\mathbf{J}_i : \mathbb{R}^{18} \to \mathbb{R}^{3\times 18}$, and contact forces $\boldsymbol{\lambda}_i \in \mathbb{R}^3$.

For these equations of motion, it is well known that for an articulated system, the underactuated, top 6 rows are of main interest for motion planning (Ponton

et al., 2018). These so-called centroidal dynamics govern the range of motion that can be achieved (Orin, Goswami, and Lee, 2013; Wieber, 2006). Solving the centroidal dynamics for base acceleration gives:

$$\begin{bmatrix}\dot{\boldsymbol{\omega}}_B \\ \dot{\mathbf{v}}_B\end{bmatrix} = \mathbf{M}_B^{-1}\left(\boldsymbol{\tau}_B^{\text{dist}} - \mathbf{M}_{Bj}\ddot{\mathbf{q}}_j - \mathbf{n}_B + \sum_{i\in\mathcal{C}} \mathbf{J}_{B,i}^{\top}\boldsymbol{\lambda}_i\right), \tag{6.7}$$

$$= \mathbf{f}_B(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}_j, \boldsymbol{\lambda}, \boldsymbol{\tau}_B^{\text{dist}}), \tag{6.8}$$

where $\mathbf{M}_B \in \mathbb{R}^{6\times 6}$ is the compound inertia tensor at the top left of $\mathbf{M}(\mathbf{q})$, and $\mathbf{M}_{Bj} \in \mathbb{R}^{6\times 12}$ is the top right block that encodes inertial coupling between the legs and base. The other terms with subscript B correspond to the top 6 rows of the same terms in (6.6).

To simplify the torso dynamics, we evaluate this function with zero inertial coupling forces from the joints, i.e. $\mathbf{M}_{Bj}\ddot{\mathbf{q}}_j = \mathbf{0}$. This simplification allows us to consider the legs only on velocity level and removes joint accelerations from the formulation. From here, further simplifications would be possible. Evaluating the function at a nominal joint configuration and zero joint velocity creates a constant inertia matrix and gives rise to the commonly used single rigid body assumption. While this assumption is appropriate on flat terrain, the joints move far away from their nominal configuration in this work, creating a significant shift in mass distribution and center of mass location.

6.4.3 Input Loopshaping

The bandwidth limitations of the series elastic actuators used in ANYmal pose an additional constraint on the set of motions that are feasible on hardware. Instead of trying to accurately model these actuator dynamics, we use a frequency-dependent cost function to penalize high-frequency content in the contact forces and joint velocity signals (Grandia et al., 2019a). For completeness, we present here the resulting system augmentation in the time domain:

$$\begin{aligned} \dot{\mathbf{s}}_\lambda &= \mathbf{A}_\lambda \mathbf{s}_\lambda + \mathbf{B}_\lambda \boldsymbol{\nu}_\lambda, & \dot{\mathbf{s}}_j &= \mathbf{A}_j \mathbf{s}_j + \mathbf{B}_j \boldsymbol{\nu}_j, \\ \boldsymbol{\lambda} &= \mathbf{C}_\lambda \mathbf{s}_\lambda + \mathbf{D}_\lambda \boldsymbol{\nu}_\lambda, & \dot{\mathbf{q}}_j &= \mathbf{C}_j \mathbf{s}_j + \mathbf{D}_j \boldsymbol{\nu}_j, \end{aligned} \tag{6.9}$$

where $\mathbf{s}_\lambda$ and $\mathbf{s}_j$ are additional states, and $\boldsymbol{\nu}_\lambda$ and $\boldsymbol{\nu}_j$ are auxiliary inputs, associated with contact forces and joint velocities respectively. When the filters ($\boldsymbol{\nu}_\lambda \to \boldsymbol{\lambda}$ and $\boldsymbol{\nu}_j \to \dot{\mathbf{q}}_j$) are low-pass filters, penalizing the auxiliary input is equivalent to penalizing high frequency content in $\boldsymbol{\lambda}$ and $\dot{\mathbf{q}}_j$.

An extreme case is obtained when choosing $\mathbf{A}_\lambda = \mathbf{D}_\lambda = \mathbf{0}$, $\mathbf{B}_\lambda = \mathbf{C}_\lambda = \mathbf{I}$, in which case the auxiliary input becomes the derivative, $\dot{\boldsymbol{\lambda}}$. This reduces to the common system augmentation technique that allows penalization of input rates (Rawlings, Mayne, and Diehl, 2017).

In our case we allow some direct control ($\mathbf{D} \neq \mathbf{0}$) and select $\mathbf{A}_\lambda = \mathbf{A}_j = \mathbf{0}$, $\mathbf{B}_\lambda = \mathbf{B}_j = \mathbf{I}$, $\mathbf{C}_\lambda = \frac{100}{4}\mathbf{I}$, $\mathbf{C}_j = \frac{50}{3}\mathbf{I}$, $\mathbf{D}_\lambda = \frac{1}{4}\mathbf{I}$, $\mathbf{D}_j = \frac{1}{3}\mathbf{I}$. This corresponds to a progressive increase in cost up to a frequency of 100 rad/s for $\boldsymbol{\lambda}$ and up to 50 rad/s for $\dot{\mathbf{q}}_j$, where high frequency components have their cost increased by a factor of 4 and 3 respectively.

6.4.4 System Dynamics

We are now ready to define the state vector $\mathbf{x} \in \mathbb{R}^{48}$ and input vector $\mathbf{u} \in \mathbb{R}^{24}$ used during motion optimization:

$$\mathbf{x} = \left[\boldsymbol{\theta}_B^\top, \mathbf{p}_B^\top, \boldsymbol{\omega}_B^\top, \mathbf{v}_B^\top, \mathbf{q}_j^\top, \mathbf{s}_\lambda^\top, \mathbf{s}_j^\top\right]^\top \quad \mathbf{u} = \left[\boldsymbol{\nu}_\lambda^\top, \boldsymbol{\nu}_j^\top\right]^\top. \tag{6.10}$$

Putting together the robot dynamics from section 6.4.2 and system augmentation described in 6.4.3 gives the continuous time MPC model $\dot{\mathbf{x}} = \mathbf{f}^c(\mathbf{x}, \mathbf{u}, t)$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{\theta}_B \\ \mathbf{p}_B \\ \boldsymbol{\omega}_B \\ \mathbf{v}_B \\ \mathbf{q}_j \\ \mathbf{s}_\lambda \\ \mathbf{s}_j\end{bmatrix} = \begin{bmatrix} \mathbf{T}(\boldsymbol{\theta}_B)\boldsymbol{\omega}_B \\ \mathbf{R}_B(\boldsymbol{\theta}_B)\,\mathbf{v}_B \\ \mathbf{f}_B(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{0}, \mathbf{C}_\lambda\mathbf{s}_\lambda + \mathbf{D}_\lambda\boldsymbol{\nu}_\lambda, \boldsymbol{\tau}_B^{\text{dist}}) \\ \mathbf{C}_j\mathbf{s}_j + \mathbf{D}_j\boldsymbol{\nu}_j \\ \mathbf{A}_\lambda\mathbf{s}_\lambda + \mathbf{B}_\lambda\boldsymbol{\nu}_\lambda \\ \mathbf{A}_j\mathbf{s}_j + \mathbf{B}_j\boldsymbol{\nu}_j \end{bmatrix}, \tag{6.11}$$

where $\mathbf{T}(\boldsymbol{\theta}_B) : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ provides the conversion between angular body rates and Euler angle derivatives, and $\mathbf{R}_B(\boldsymbol{\theta}_B) : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ provides the body to world rotation matrix. The disturbance wrench $\boldsymbol{\tau}_B^{\text{dist}}$ is considered a parameter and is assumed constant over the MPC horizon.

6.4.5 Reference Generation

The user commands 2D linear velocities and an angular rate in the horizontal plane, as well as a desired gait pattern. A full motion and contact force reference is generated to encode these user commands and additional motion

preferences into the cost function defined in section 6.4.6. This process is carried out before every MPC iteration.

As a first step, assuming a constant input along the horizon, a 2D base reference position and heading direction are extrapolated in the world frame. At each point in time, the 2D pose is converted to a 2D position for each hip. The smoothened elevation map, i.e. the *torso reference* layer shown in Fig 6.3, is interpolated at the 2D hip location. The interpolated elevation in addition to a desired nominal height, h_{nom}, gives a 3D reference position for each hip. A least-squares fit through the four hip positions gives the 6DoF base reference.

The extracted base reference and the desired gait pattern are used to derive nominal foothold locations. Here we use the common heuristic that the nominal foothold is located below the hip at the middle of the contact phase (Raibert, 1986). Additionally, for the first upcoming foothold, a feedback on the measured velocity is added:

$$\mathbf{p}_{i,nom} = \mathbf{p}_{i,hip,nom} + \sqrt{\frac{h_{nom}}{g}}(\mathbf{v}_{B,meas} - \mathbf{v}_{B,com}), \tag{6.12}$$

where $\mathbf{p}_{i,nom} \in \mathbb{R}^3$ is the nominal foothold, $\mathbf{p}_{i,hip,nom} \in \mathbb{R}^3$ is the nominal foothold location directly below the hip, and g is the gravitational constant. $\mathbf{v}_{B,meas}$ and $\mathbf{v}_{B,com}$ are measured and commanded base velocity respectively.

With the nominal foothold locations known, the plane segmentation defined in section 6.3.2 is used to adapt the nominal foothold locations to the perceived terrain. Each foothold is projected onto the plane that is closest and within kinematic limits. Concretely, we find the closest point on each of the available planes and pick the reference foothold, $\mathbf{p}_{i,ref}$, according to:

$$\underset{\mathbf{p}_{i,ref} \in \Pi(\mathbf{p}_{i,nom})}{\operatorname{argmin}} \|\mathbf{p}_{i,nom} - \mathbf{p}_{i,ref}\|_2^2 + w_{kin} f_{kin}(\mathbf{p}_{i,ref}), \tag{6.13}$$

where $\Pi(\mathbf{p}_{i,nom})$ is the set of projected candidate points, one per segmented plane, and f_{kin} is a kinematic penalty with weight w_{kin} that penalizes the point if the leg extension at liftoff or touchdown is beyond a threshold and if the foothold crosses over to the opposite side of the body. Essentially, this is a simplified version of the foothold batch search algorithm presented in (Jenelten et al., 2020), which searches over cells of the map instead of pre-segmented planes.

After computing all projected footholds, heuristic swing trajectories are computed with two quintic splines; from liftoff to apex and apex to touchdown. The spline is constrained by a desired liftoff and touchdown velocity, and an apex location is selected in such a way that the trajectory clears the highest terrain point between the footholds. Inverse kinematics is used to derive joint positions reference corresponding to the base and feet references.

Finally, contact forces references are computed by dividing the total weight of the robot equally among all feet that are in contact. Joint velocity references are set to zero.

6.4.6 Cost & Soft Inequality Constraints

The cost function (6.4a) is built out of several components. The running cost $L(\mathbf{x}, \mathbf{u}, t)$ can be split into tracking costs $L_{\boldsymbol{\epsilon}}$, loopshaping costs $L_{\boldsymbol{\nu}}$, and penalty costs $L_{\mathcal{B}}$:

$$L = L_{\boldsymbol{\epsilon}} + L_{\boldsymbol{\nu}} + L_{\mathcal{B}}. \tag{6.14}$$

The motion tracking cost are used to follow the reference trajectory defined in section 6.4.5. Tracking error are defined for the base, $\boldsymbol{\epsilon}_B$, and for each foot ,$\boldsymbol{\epsilon}_i$,

$$\boldsymbol{\epsilon}_B = \begin{bmatrix} \log(\mathbf{R}_B \mathbf{R}_{B,ref}^\top)^\vee \\ \mathbf{p}_B - \mathbf{p}_{B,ref} \\ \boldsymbol{\omega}_B - \boldsymbol{\omega}_{B,ref} \\ \mathbf{v}_B - \mathbf{v}_{B,ref} \end{bmatrix}, \boldsymbol{\epsilon}_i = \begin{bmatrix} \mathbf{q}_i - \mathbf{q}_{i,ref} \\ \dot{\mathbf{q}}_i - \dot{\mathbf{q}}_{i,ref} \\ \mathbf{p}_i - \mathbf{p}_{i,ref} \\ \mathbf{v}_i - \mathbf{v}_{i,ref} \\ \boldsymbol{\lambda}_i - \boldsymbol{\lambda}_{i,ref} \end{bmatrix}, \tag{6.15}$$

where $\log(\mathbf{R}_B \mathbf{R}_{B,ref}^\top)^\vee$ is the logarithmic map of the orientation error, represented as a 3D rotation vector, and $\mathbf{p}_i$ and $\mathbf{v}_i$ are the foot position and velocity in world frame. Together with diagonal, positive definite, weight matrices $\mathbf{W}_B$ and $\mathbf{W}_i$, these errors form the following nonlinear least-squares cost:

$$L_{\boldsymbol{\epsilon}} = \frac{1}{2} \|\boldsymbol{\epsilon}_B\|^2_{\mathbf{W}_B} + \sum_{i=1}^{4} \frac{1}{2} \|\boldsymbol{\epsilon}_i\|^2_{\mathbf{W}_i}. \tag{6.16}$$

As discussed in section 6.4.3, high-frequency content in joint velocities and contact forces are penalized through a cost on the corresponding auxiliary input. This cost is a simple quadratic cost:

$$L_{\boldsymbol{\nu}} = \frac{1}{2} \boldsymbol{\nu}_\lambda^\top \mathbf{R}_\lambda \boldsymbol{\nu}_\lambda + \frac{1}{2} \boldsymbol{\nu}_j^\top \mathbf{R}_j \boldsymbol{\nu}_j, \tag{6.17}$$

where $\mathbf{R}_\lambda$ and $\mathbf{R}_j$ are constant, positive semi-definite, weight matrices. To obtain an appropriate scaling and avoid further manual tuning, these matrices are obtained from the quadratic approximation of the motion tracking cost (6.16), with respect to $\boldsymbol{\lambda}$ and $\dot{\mathbf{q}}_j$ respectively, at the nominal stance configuration of the robot.

All inequality constraints are handled through the penalty cost. In this work, we use relaxed barrier functions (Feller and Ebenbauer, 2017; Hauser and Saccon, 2006). This penalty function is defined as a log-barrier on the interior of the feasible space and switches to a quadratic function at a distance δ from the constraint boundary.

$$\mathcal{B}(h) = \begin{cases} -\mu \ln(h), & h \geq \delta, \\ \frac{\mu}{2}\left(\left(\frac{h-2\delta}{\delta}\right)^2 - 1\right) - \mu \ln(\delta), & h < \delta. \end{cases} \tag{6.18}$$

The penalty is taken element-wise for vector-valued inequality constraints. The sum of all penalties is given as follows:

$$L_{\mathcal{B}} = \sum_{i=1}^{4} \mathcal{B}_j\left(\mathbf{h}_i^j\right) + \sum_{i \in \mathcal{C}} \mathcal{B}_t\left(\mathbf{h}_i^t\right) + \mathcal{B}_\lambda\left(h_i^\lambda\right) + \sum_{c \in \mathcal{D}} \mathcal{B}_d\left(h_c^d\right), \tag{6.19}$$

with joint limit constraints $\mathbf{h}_i^j$ for all legs, foot placement and friction cones constraints, $\mathbf{h}_i^t$ and h_i^λ, for legs in contact, and collision avoidance constraints h_c^d for all bodies in a set $\mathcal{D}$.

The joint limits constraints contain upper $\{\overline{\mathbf{q}}_j, \overline{\dot{\mathbf{q}}}_j, \overline{\boldsymbol{\tau}}\}$ and lower bounds $\{\underline{\mathbf{q}}_j, \underline{\dot{\mathbf{q}}}_j, \underline{\boldsymbol{\tau}}\}$ for positions, velocities, and torques:

$$\mathbf{h}_i^j = \begin{bmatrix} \overline{\mathbf{q}}_j - \mathbf{q}_j \\ \mathbf{q}_j - \underline{\mathbf{q}}_j \\ \overline{\dot{\mathbf{q}}}_j - \dot{\mathbf{q}}_j \\ \dot{\mathbf{q}}_j - \underline{\dot{\mathbf{q}}}_j \\ \overline{\boldsymbol{\tau}} - \boldsymbol{\tau} \\ \boldsymbol{\tau} - \underline{\boldsymbol{\tau}} \end{bmatrix} \geq \mathbf{0}, \tag{6.20}$$

where we approximate the joint torques by considering a static equilibrium in each leg, i.e. $\boldsymbol{\tau}_i = \mathbf{J}_{j,i}^\top \boldsymbol{\lambda}_i$.

The foot placement constraint is a set of linear inequality constraints in task space:

$$\mathbf{h}_i^t = \mathbf{A}_i^t \cdot \mathbf{p}_i + \mathbf{b}_i^t \geq \mathbf{0}, \tag{6.21}$$

where $\mathbf{A}_i^t \in \mathbb{R}^{m\times 3}$, and $\mathbf{b}_i^t \in \mathbb{R}^m$ define m half-space constraints in 3D. Each half-space is defined as the plane spanned by an edge of the 2D polygon and the surface normal of the touchdown terrain $\mathcal{F}_{T+}$. The polygon is obtained by initializing all m vertices at the reference foothold derived in section 6.4.5 and iteratively displacing them outwards. Each vertex is displaced in a round-robin fashion until it reaches the border of the segmented region or until further movement would cause the polygon to become non-convex. Similar to (Deits and Tedrake, 2015), we have favoured the low computational complexity of an iterative scheme over an exact approach of obtaining a convex inner approximation. The extracted constraints remain unaltered for a foot that is in the second half of an ongoing swing phase to prevent large, last-minute jumps in constraints.

The friction cone constraint is implemented as:

$$h_i^\lambda = \mu_c F_z - \sqrt{F_x^2 + F_y^2 + \epsilon^2} \geq 0, \tag{6.22}$$

with $[F_x, F_y, F_z]^\top = \mathbf{R}_T^\top \mathbf{R}_B \boldsymbol{\lambda}_i$, defining the forces in the local terrain frame. μ_c is the friction coefficient, and $\epsilon > 0$ is a parameter that ensures a continuous derivative at $\boldsymbol{\lambda}_i = \mathbf{0}$, and at the same time creates a safety margin (Grandia et al., 2019b).

The collision avoidance constraint is given by evaluation of the SDF at the center of a collision sphere, $\mathbf{p}_c$, together with the required distance given by the radius, r_c, and a shaping function $d_{\min}(t)$.

$$h_c^d = d^{SDF}(\mathbf{p}_c) - r_c - d_{\min}(t) \geq 0. \tag{6.23}$$

The primary use of the shaping function is to relax the constraint if a foot starts a swing phase from below the map. To avoid the robot using maximum velocity to escape the collision, we provide smooth guidance back to free space with a cubic spline trajectory. This happens when the perceived terrain is higher than the actual terrain, for example in case of a soft terrain like vegetation and snow, or simply because of drift and errors in the estimated map. The collision set $\mathcal{D}$ contains collision bodies for all knees and for all feet that are in swing phase, as visualized on the hind legs in Fig. 6.4.

Algorithm 3 Real-time iteration Multiple-shooting MPC

1: **Given:** previous solution $\mathbf{w}_i$
2: Discretize the continous problem to the form of (6.27)
3: Compute the linear quadratic approximation (6.30)
4: Compute the equality constraint projection (6.34)
5: $\delta\tilde{\mathbf{w}} \leftarrow$ Solve the projected QP subproblem (6.35)
6: $\delta\mathbf{w} \leftarrow \mathbf{P}\delta\tilde{\mathbf{w}} + \mathbf{p}$, back substitution using (6.33)
7: $\mathbf{w}_{i+1} \leftarrow$ Line-Search($\mathbf{w}_i$, $\delta\mathbf{w}$), (Algorithm 4)

Finally, we use a quadratic cost as the terminal cost in (6.4a). To approximate the infinite horizon cost incurred after the finite horizon length, we solve a Linear Quadratic Regulator (LQR) problem for the linear approximation of the MPC model and quadratic approximation of the intermediate costs around the nominal stance configuration of the robot. The Riccati matrix $\mathbf{S}_{\text{LQR}}$ of the cost-to-go is used to define the quadratic cost around the reference state:

$$\Phi(\mathbf{x}) = \frac{1}{2} \left(\mathbf{x} - \mathbf{x}_{ref}(T)\right)^\top \mathbf{S}_{\text{LQR}} \left(\mathbf{x} - \mathbf{x}_{ref}(T)\right). \tag{6.24}$$

6.4.7 Equality Constraints

For each foot in swing phase, the contact forces are required to be zero:

$$\boldsymbol{\lambda}_i = \mathbf{0}, \qquad \forall i \notin \mathcal{C}. \tag{6.25}$$

Additionally, for each foot in contact, the end-effector velocity is constrained to be zero. For swing phases, the reference trajectory is enforced only in the normal direction. This ensures that the foot lifts off and touches down with a specified velocity while leaving complete freedom of foot placement in the tangential direction.

$$\begin{cases} \mathbf{v}_i = \mathbf{0}, & \text{if } i \in \mathcal{C}, \\ \mathbf{n}^\top(t) \left(\mathbf{v}_i - \mathbf{v}_{i,ref} + k_p(\mathbf{p}_i - \mathbf{p}_{i,ref})\right) = 0, & \text{if } i \notin \mathcal{C}, \end{cases}$$

The surface normal, $\mathbf{n}(t)$, is interpolated over time since liftoff and touchdown terrain can have a different orientation.

6.5 Numerical Optimization

We consider a direct multiple-shooting approach to transforming the continuous optimal control problem into a finite-dimensional nonlinear program (NLP)

(Bock and Plitt, 1984). Since MPC computes control inputs over a receding horizon, successive instances of (6.27) are similar and can be efficiently warm-started when taking an SQP approach. Additionally, we follow the real-time iteration scheme where only one SQP step is performed per MPC update (M. Diehl and H.G. Bock and J. P. Schlöder and R. Findeisen and Z. Nagy and F. Allgöwer, 2002). In this way, the solution is improved across consecutive instances of the problem, rather than iterating until convergence for each problem.

As an overview of the approach described in the following sections, a pseudo-code is provided in Algorithm 3, referring to the relevant equations used at each step. Except for the solution of the QP in line 5, all steps of the algorithm are parallelized across the shooting intervals. The QP is solved using HPIPM (Frison and Diehl, 2020).

6.5.1 Discretization

The continuous control signal $\mathbf{u}(t)$ is parameterized over subintervals of the prediction horizon $[t, t+T]$ to obtain a finite-dimensional decision problem. This creates a grid of nodes $k \in \{0, \dots, N\}$ defining control times t_k separated by intervals of duration $\delta t \approx T/(N-1)$. Around gait transitions, δt is slightly shortened or extended such that a node is exactly at the gait transition.

In this work, we consider a piecewise constant, or zero-order-hold, parameterization of the input. Denoting $\mathbf{x}_k = \mathbf{x}(t_k)$ and integrating the continuous dynamics in (6.11) over an interval leads to a discrete time representation of the dynamics:

$$\mathbf{f}_k^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{x}_k + \int_{t_k}^{t_k+\delta t} \mathbf{f}^c(\mathbf{x}(\tau), \mathbf{u}_k, t)\ \mathrm{d}\tau. \tag{6.26}$$

The integral in (6.26) is numerically approximated with an integration method of choice to achieve the desired approximation accuracy of the evolution of the continuous time system under the zero-order-hold commands. We use an explicit second-order Runge-Kutta scheme.

The general nonlinear MPC problem presented below can be formulated by defining and evaluating a cost function and constraints on the grid of nodes.

$$\min_{\mathbf{X},\mathbf{U}} \quad \Phi(\mathbf{x}_N) + \sum_{k=0}^{N-1} l_k(\mathbf{x}_k, \mathbf{u}_k) \tag{6.27a}$$

$$\text{s.t.} \quad \mathbf{x}_0 - \hat{\mathbf{x}} = \mathbf{0}, \tag{6.27b}$$

$$\mathbf{x}_{k+1} - \mathbf{f}_k^d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, \qquad k = 0, \dots, N-1, \tag{6.27c}$$

$$\mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, \qquad k = 0, \dots, N-1, \tag{6.27d}$$

where $\mathbf{X} = [\mathbf{x}_0^\top, \dots \mathbf{x}_N^\top]^\top$, and $\mathbf{U} = [\mathbf{u}_0^\top, \dots \mathbf{u}_{N-1}^\top]^\top$, are the sequences of state and input variables respectively. The nonlinear cost and constraint functions l_k, and $\mathbf{g}_k$, are discrete sample of the continuous counterpart. Collecting all decision variables into a vector, $\mathbf{w} = [\mathbf{X}^\top, \mathbf{U}^\top]^\top$, problem (6.27) can be written as a general NLP:

$$\min_{\mathbf{w}} \quad \phi(\mathbf{w}), \quad \text{s.t.} \quad \begin{bmatrix} \mathbf{F}(\mathbf{w}) \\ \mathbf{G}(\mathbf{w}) \end{bmatrix} = \mathbf{0}, \tag{6.28}$$

where $\phi(\mathbf{w})$ is the cost function, $\mathbf{F}(\mathbf{w})$ is the collection of initial state and dynamics constraints, and $\mathbf{G}(\mathbf{w})$ is the collection of all general equality constraints.

6.5.2 Sequential Quadratic Programming (SQP)

SQP based methods apply Newton-type iterations to Karush-Kuhn-Tucker (KKT) optimality conditions, assuming some regularity conditions on the constraints (Mangasarian and Fromovitz, 1967). The Lagrangian of the NLP in (6.28) is defined as:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}_\mathbf{G}, \boldsymbol{\lambda}_\mathbf{H}) = \phi(\mathbf{w}) + \boldsymbol{\lambda}_\mathbf{F}^\top \mathbf{F}(\mathbf{w}) + \boldsymbol{\lambda}_\mathbf{G}^\top \mathbf{G}(\mathbf{w}), \tag{6.29}$$

with Lagrange multipliers $\boldsymbol{\lambda}_\mathbf{F}$ and $\boldsymbol{\lambda}_\mathbf{G}$, corresponding to the dynamics and equality constraints. The Newton iterations can be equivalently computed by

solving the following potentially non-convex QP (Nocedal and Wright, 2006):

$$\min_{\delta \mathbf{w}} \quad \nabla_{\mathbf{w}} \phi(\mathbf{w}_i)^\top \delta \mathbf{w} + \frac{1}{2} \delta \mathbf{w}^\top \mathbf{B}_i \delta \mathbf{w}, \tag{6.30a}$$

$$\text{s.t} \quad \mathbf{F}(\mathbf{w}_i) + \nabla_{\mathbf{w}} \mathbf{F}(\mathbf{w}_i)^\top \delta \mathbf{w} = \mathbf{0}, \tag{6.30b}$$

$$\mathbf{G}(\mathbf{w}_i) + \nabla_{\mathbf{w}} \mathbf{G}(\mathbf{w}_i)^\top \delta \mathbf{w} = \mathbf{0}, \tag{6.30c}$$

where the decision variables, $\delta \mathbf{w} = \mathbf{w} - \mathbf{w}_i$, define the update step relative to the current iteration $\mathbf{w}_i$, and the Hessian $\mathbf{B}_i = \nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{w}_i, \boldsymbol{\lambda}_{\mathbf{F}}, \boldsymbol{\lambda}_{\mathbf{G}})$. Computing the solution to (6.30) provides a candidate decision variable update, $\delta \mathbf{w}_i$, and updated Lagrange multipliers.

6.5.3 Quadratic Approximation Strategy

As we seek to deploy MPC on dynamic robotic platforms, it is critical that the optimization problem in (6.30) is well conditioned and does not provide difficulty to numerical solvers. In particular, when $\mathbf{B}_i$ in (6.30a) is positive semi-definite (p.s.d), the resulting QP is convex and can be efficiently solved (Kouzoupis et al., 2018).

To ensure this, an approximate, p.s.d Hessian is used instead of the full Hessian of the Lagrangian. For the tracking costs (6.16), the objective function has a least-squares form in which case the Generalized Gauss-Newton approximation,

$$\nabla^2_{\mathbf{w}} \left(\frac{1}{2} \| \boldsymbol{\epsilon}_i(\mathbf{w}) \|^2_{\mathbf{W}_i} \right) \approx \nabla_{\mathbf{w}} \boldsymbol{\epsilon}_i(\mathbf{w})^\top \mathbf{W}_i \nabla_{\mathbf{w}} \boldsymbol{\epsilon}_i(\mathbf{w}), \tag{6.31}$$

proves effective in practice (Houska, Ferreau, and Diehl, 2011). Similarly, for the soft constraints, we exploit to convexity of the penalty function applied to the nonlinear constraint (Verschueren et al., 2016):

$$\nabla^2_{\mathbf{w}} \left(\mathcal{B}(\mathbf{h}(\mathbf{w})) \right) \approx \nabla_{\mathbf{w}} \mathbf{h}(\mathbf{w})^\top \nabla^2_{\mathbf{h}} \mathcal{B}(\mathbf{h}(\mathbf{w})) \nabla_{\mathbf{w}} \mathbf{h}(\mathbf{w}), \tag{6.32}$$

where the diagonal matrix $\nabla^2_{\mathbf{h}} \mathcal{B}(\mathbf{h}(\mathbf{w}))$ maintains the curvature information of the convex penalty functions. The contribution of the constraints to the Lagrangian in (6.29) is ignored in the approximate Hessian since we do not have additional structure that allows a convex approximation.

6.5.4 Constraint Projection

The equality constraints in 6.4.7 were carefully chosen to have full row rank w.r.t. the control inputs, such that, after linearization, $\nabla_{\mathbf{w}}\mathbf{G}(\mathbf{w}_i)^\top$ has full row rank in (6.30c). This means that the equality constraints can be eliminated before solving the QP through a change of variables (Nocedal and Wright, 2006):

$$\delta\mathbf{w} = \mathbf{P}\delta\tilde{\mathbf{w}} + \mathbf{p}, \tag{6.33}$$

where the linear transformation satisfies

$$\nabla_{\mathbf{w}}\mathbf{G}(\mathbf{w}_i)^\top\mathbf{P} = \mathbf{0}, \quad \nabla_{\mathbf{w}}\mathbf{G}(\mathbf{w}_i)^\top\mathbf{p} = -\mathbf{G}(\mathbf{w}_i). \tag{6.34}$$

After substituting (6.33) into (6.30), the following QP is solved w.r.t. $\delta\tilde{\mathbf{w}}$.

$$\min_{\delta\tilde{\mathbf{w}}} \quad \nabla_{\tilde{\mathbf{w}}}\tilde{\phi}(\mathbf{w}_i)^\top\delta\tilde{\mathbf{w}} + \frac{1}{2}\delta\tilde{\mathbf{w}}^\top\tilde{\mathbf{B}}_i\delta\tilde{\mathbf{w}}, \tag{6.35a}$$

$$\text{s.t} \quad \tilde{\mathbf{F}}(\mathbf{w}_i) + \nabla_{\tilde{\mathbf{w}}}\tilde{\mathbf{F}}(\mathbf{w}_i)^\top\delta\tilde{\mathbf{w}} = \mathbf{0}. \tag{6.35b}$$

Because each constraint applies only to the variables at one node k, the coordinate transformation maintains the sparsity pattern of an optimal control problem and can be computed in parallel. Since this projected problem now only contains costs and system dynamics, solving the QP only requires one Ricatti-based iteration(Frison and Diehl, 2020). The full update $\delta\mathbf{w}$ is then obtained through back substitution into (6.33).

6.5.5 Line-Search

To select an appropriate stepsize, we employ a line-search based on the filter line-search used in IPOPT (Wächter and Biegler, 2006). In contrast to a line-search based on a merit function, where cost and constraints are combined to one metric, the main idea is to ensure that each update either improves the constraint satisfaction or the cost function. The constraint satisfaction $\theta(\mathbf{w})$ is measured by taking the norm of all constraints scaled by the time discretization:

$$\theta(\mathbf{w}) = \delta t \left|\left| \left[\mathbf{F}(\mathbf{w})^\top, \mathbf{G}(\mathbf{w})^\top\right]^\top \right|\right|_2. \tag{6.36}$$

In case of high or low constraint satisfaction, the behavior is adapted: When the constraint is violated beyond a set threshold, $\theta_{\max}$, the focus changes purely to decreasing the constraints; when constraint violation is below a minimum threshold, $\theta_{\min}$, the focus changes to minimizing costs.

Compared to the algorithm presented in (Wächter and Biegler, 2006), we remove recovery strategies and second-order correction steps, for which there is no time in the online setting. Furthermore, the history of iterates plays no role since we perform only one iteration per problem.

The simplified line-search as used in this work is given in Algorithm 4 and contains three distinct branches in which a step can be accepted. The behavior at high constraint violation is given by line 9, where a step is rejected if the new constraint violation is above the threshold and worse than the current violation. The switch to the low constraint behavior is made in line 13: if both new and old constraint violations are low and the current step is in a descent direction, we require that the cost decrease satisfies the Armijo condition in line 14. Finally, the primary acceptance condition is given in line 18, where either a cost or constraint decrease is requested. The small constants γ_ϕ, and γ_θ are used to fine-tune this condition with a required non-zero decrease in either quantity.

6.6 Motion Execution

The optimized motion planned by the MPC layer consists of contact forces and desired joint velocities. We linearly interpolate the MPC motion plan at the 400 Hz execution rate and apply the feedback gains derived from the Riccati backward pass to the measured state (Grandia et al., 2019b). The corresponding torso acceleration is obtained through (6.8). The numerical derivative of the planned joint velocities is used to determine a feedforward joint acceleration. A high-frequency whole-body controller (WBC) is used to convert the desired acceleration tasks into torque commands (Dario Bellicoso et al., 2016; Saab et al., 2013; Sentis and Khatib, 2006). A generalized momentum observer is used to estimate the contact state (Bledt et al., 2018). Additionally, the estimated external torques are filtered and added to the MPC and WBC dynamics as described in (Jenelten et al., 2021).

6.6.1 Event Based Execution

Inevitably, the measured contact state will be different from the planned contact state used during the MPC optimization. In this case, the designed contact forces cannot be provided by the whole-body controller. We have implemented simple reactive behaviors to respond to this situation and provide feedback to the MPC layer.

Algorithm 4 Backtracking Line-Search

Hyperparameters: $\alpha_{\min} = 10^{-4}, \theta_{\max} = 10^{-2}, \theta_{\min} = 10^{-6}, \eta = 10^{-4}, \gamma_\phi = 10^{-6}, \gamma_\theta = 10^{-6}, \gamma_\alpha = 0.5$
$\alpha \leftarrow 1.0$
$\theta_k \leftarrow \theta(\mathbf{w}_i)$
$\phi_k \leftarrow \phi(\mathbf{w}_i)$
Accepted $\leftarrow$ False
while *Not* Accepted and $\alpha \geq \alpha_{\min}$ **do**
    $\theta_{i+1} \leftarrow \theta(\mathbf{w}_i + \alpha\delta\mathbf{w})$
    $\phi_{i+1} \leftarrow \phi(\mathbf{w}_i + \alpha\delta\mathbf{w})$
    if $\theta_{i+1} > \theta_{\max}$ **then**
        if $\theta_{i+1} < (1-\gamma_c)\theta_i$ **then**
            Accepted $\leftarrow$ True
        end if
    else if $\max(\theta_{i+1}, \theta_i) < \theta_{\min}$ and $\nabla\phi(\mathbf{w}_i)^\top \delta\mathbf{w} < 0$ **then**
        if $\phi_{i+1} < \phi_i + \eta\alpha\nabla\phi(\mathbf{w}_i)^\top\delta\mathbf{w}$ **then**
            Accepted $\leftarrow$ True
        end if
    else
        if $\phi_{i+1} < \phi_i - \gamma_\phi\theta_i$ or $\theta_{i+1} < (1-\gamma_c)\theta_i$ **then**
            Accepted $\leftarrow$ True
        end if
    end if
    if *Not* Accepted **then**
        $\alpha \leftarrow \gamma_\alpha\alpha$
    end if
end while
if Accepted **then**
    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha\delta\mathbf{w}$
else
    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i$
end if

In case there is a planned contact, but no contact is measured, we follow a downward *regaining* motion for that foot. Under the assumption that the contact mismatch will be short, the MPC will start a new plan again from a closed contact state. Additionally, we propagate the augmented system in (6.9) with the information that no contact force was generated, i.e. $\mathbf{0} \stackrel{!}{=} \mathbf{C}_\lambda\mathbf{s}_\lambda + \mathbf{D}_\lambda\boldsymbol{\nu}_\lambda$. In this way, the MPC layer will generate contact forces that maintain the requested smoothness w.r.t. the executed contact forces.

When contact is measured, but no contact was planned, the behavior depends on the planned time till contact. If contact was planned to happen soon, the measured contact is sent to the MPC to generate the next plan from that early contact state. If no upcoming contact was planned, the measured contact is ignored.

6.6.2 Whole-body Control

The whole-body control (WBC) approach considers the full nonlinear rigid body dynamics of the system in (6.6), including the estimate of disturbance forces. Each task is formulated as an equality constraint, inequality constraint, or least-squares objective affine in the generalized accelerations, torques, and contact forces. While we have used a hierarchical resolution of tasks in the past (Dario Bellicoso et al., 2016), in this work, we instead use a single QP and trade off the tracking tasks with weights. We found that a strict hierarchy results in a dramatic loss of performance in lower priority tasks when inequalities constraints are active. Additionally, the complexity of solving multiple QPs and null-space projections in the hierarchical approach is no longer justified with the high quality motion reference coming from the MPC.

The complete list of tasks is given in Table 6.1. The first two blocks of tasks enforce physical consistency and inequality constraints on torques, forces, and joint configurations. The joint limit constraint is derived from an exponential Control Barrier Function (CBF) (Nguyen and Sreenath, 2016b) on the joint limits, $\underline{\mathbf{q}}_j \leq \mathbf{q}_j \leq \overline{\mathbf{q}}_j$, resulting in the following joint acceleration constraints:

$$\ddot{\mathbf{q}}_j + (\gamma_1 + \gamma_2)\dot{\mathbf{q}}_j + \gamma_1\gamma_2(\mathbf{q}_j - \underline{\mathbf{q}}_j) \geq \mathbf{0}, \tag{6.37}$$

$$-\ddot{\mathbf{q}}_j - (\gamma_1 + \gamma_2)\dot{\mathbf{q}}_j + \gamma_1\gamma_2(\overline{\mathbf{q}}_j - \mathbf{q}_j) \geq \mathbf{0}, \tag{6.38}$$

with scalar parameters $\gamma_1 > 0, \gamma_2 > 0$. These CBF constraints guarantee that the state constraints are satisfied for all time and under the full nonlinear dynamics of the system (Ames, Grizzle, and Tabuada, 2014).

For the least-square tasks, we track swing leg motion with higher weight than the torso reference. This prevents that the robot exploits the leg inertia to track torso references in underactuated directions, and it ensures that the foot motion is prioritized over torso tracking when close to kinematics limits. Tracking the contact forces references with a low weight regulates the force distribution in case the contact configuration allows for internal forces.

Finally, the torque derived from the whole-body controller, $\boldsymbol{\tau}_{\text{wbc}} \in \mathbb{R}^{12}$, is computed. To compensate for model uncertainty for swing legs, the integral of joint acceleration error with gain $K > 0$ is added to the torque applied to the system:

$$\boldsymbol{\tau}_i = \boldsymbol{\tau}_{i,\text{wbc}} - K \int_{t_0^{sw}}^{t} (\ddot{\mathbf{q}}_i - \ddot{\mathbf{q}}_{i,\text{wbc}})\, \mathrm{d}t, \tag{6.39}$$

Table 6.1: WHOLE-BODY CONTROL TASKS.

Type	Task
$=$	Floating base equations of motion. No motion at the contact points.
$\geq$	Torque limits. Friction cone constraint. Joint limit barrier constraint.
$w_i^2 \| \cdot \|^2$	Swing leg motion tracking ($w_i = 100.0$). Torso linear and angular acceleration ($w_i = 1.0$). Contact force tracking. ($w_i = 0.01$).

where t_0^{sw} is the start time of the swing phase. For stance legs, a PD term is added around the planned joint configuration and contact consistent joint velocity.

6.7 Results

ANYmal is equipped with either two dome shaped Robo-Sense bpearl LiDARs, mounted in the front and back of the torso, or with four Intel RealSense D435 depth cameras mounted on each side of the robot. Elevation mapping runs at 20 Hz on an onboard GPU (Jetson AGX Xavier). Control and state estimation are executed on the main onboard CPU (Intel i7-8850H,2.6 GHz, Hexa-core) at 400 Hz, asynchronously to the MPC optimization which is triggered at 100 Hz. Four cores are used for parallel computation in the MPC optimization. A time horizon of $T = 1.0\,\text{s}$ is used with a nominal time discretization of $\delta t \approx 0.015\,\text{s}$, with a slight variation due to the adaptive discretization around gait transitions. Each multiple shooting MPC problem therefore contains around 5000 decision variables. Part (**A**) and (**B**) of perception pipeline in Fig. 6.3 are executed on a second onboard CPU of the same kind and provides the precomputed layers over Ethernet.

To study the performance of the proposed controller, we report results in different scenarios and varying levels of detail. First, results for the perception pipeline in isolation are presented in section 6.7.1. Second, we validate the major design choices in simulation in section 6.7.2. Afterward, the proposed controller is put to the test in challenging simulation, as well as hardware experiments in section 6.7.3. Finally, known limitations are discussed in section 6.7.4.

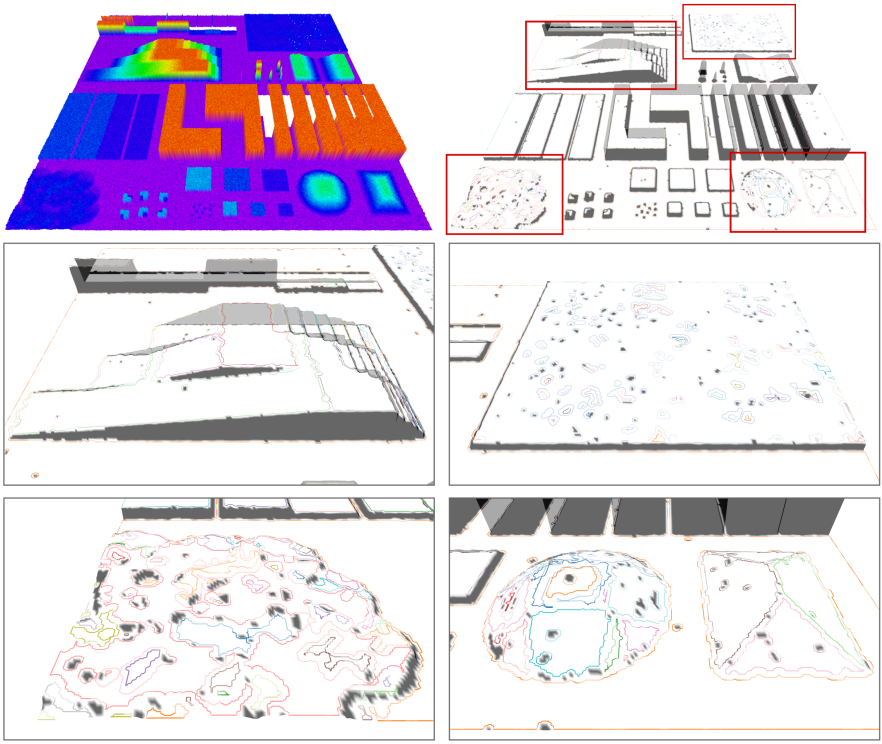


Figure 6.5: Evaluation of the plane segmentation on a demo terrain (Fankhauser and Hutter, 2016). The shown map has a true size of 20×20×1 m with a resolution of 4 cm. Top left shows the elevation map with additive uniform noise of ±2 cm plus Gaussian noise with a standard deviation of 2 cm. Top right shows the map after inpainting, filtering, steppability classification, and plane segmentation. Below, four areas of interest are shown. Their original location in the map is marked in the top right image.

6.7.1 Perception Pipeline

The output of the steppability classification and plane segmentation (part A in Fig. 6.3) for a demo terrain is shown in Fig. 6.5. This terrain is available as part of the gridmap library and contains a collection of slopes, steps, curvatures, rough terrain, and missing data. The left middle image shows that slopes and steps are, in general, well segmented. In the bottom right image, one sees the effect of the plane segmentation on a curved surface. In those cases, the terrain will be segmented into a collection of smaller planes. Finally, the rough

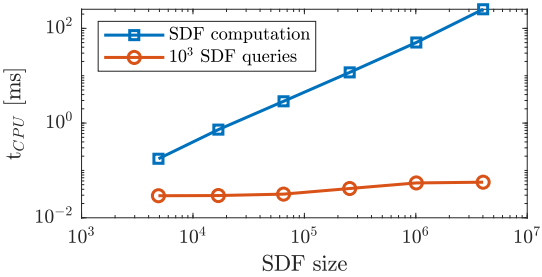


Figure 6.6: Computation time for constructing and querying the signed distance field. Submaps of the terrain in Fig. 6.5 are used. *SDF size* on the horizontal axis denotes the total amount of data points in the SDF (width×length×height). The query time is reported for the total of 10^3 random queries for the interpolated value and derivative.

terrain sections shown in the right middle and bottom left image show that the method is able to recognize such terrain as one big planar section as long as the roughness is within the specified tolerance. These cases also show the importance of allowing holes in the segmented regions, making it possible to exclude just those small regions where the local slope or roughness is outside the tolerance. A global convex decomposition of the map would result in a much larger amount of regions in these scenarios.

The computation time for the construction and querying of the signed distance field is benchmarked on sub-maps of varying sizes extracted from the demo map, see Fig. 6.6. As expected, the construction time scales linearly with the SDF size, and the query time is constant with a slight increase when the memory size exceeds a cache level. During runtime, the local SDF size is typically below 10^5 voxels, resulting in a computation time well below 10 ms. Together with the map update rate of 20 Hz, the proposed method provides the SDF at an order of magnitude faster than methods that maintain a general 3D voxel grid, with update rated reported around 1 Hz(Pankert and Hutter, 2020). Per MPC iteration, around 10^3 SDF queries are made, making the SDF query time negligible compared to the total duration of one MPC iteration.

6.7.2 Simulation

6.7.2.1 Collision Avoidance

To highlight the importance of considering knee collisions with the terrain, the robot is commanded to traverse a box of 35 cm with a trotting gait at 0.25 m/s.

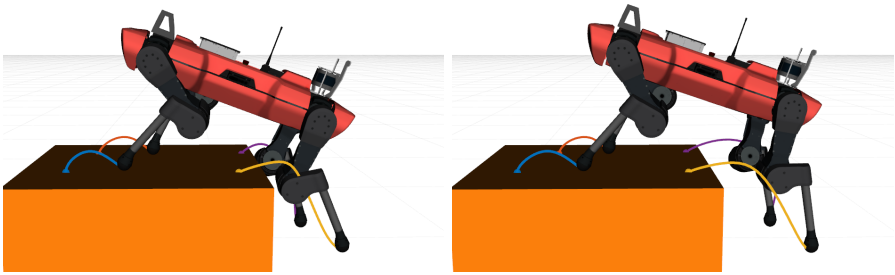


Figure 6.7: ANYmal stepping up a box of 35 cm. Left: Without considering knee collisions. Right: Knee collision included in the optimization.

Fig. 6.7 compares the simulation result of this scenario with and without the knee collisions considered. The inclusion of knee collision avoidance is required to successfully step up the box with the hind legs. As shown in the figure, the swing trajectories are altered. Furthermore, the base pose and last stepping location before stepping up are adjusted to prepare for the future, showing the benefit of considering all degrees of freedom in one optimization.

Fig. 6.8 provides insight into the solver during the motion performed with the knee collisions included. The four peaks in the cost function show the effect of the collision avoidance penalty when the legs are close to the obstacle during the step up and step down. Most of the time, the step obtained from the QP subproblem is accepted by the line-search with the full stepsize of 1.0. However, between 7 and 8 s the stepsize is decreased to prevent the constraint violation from further rising. This happens when the front legs step down the box and are close to collision. In those cases, the collision avoidance penalty is highly nonlinear, and the line-search is required to maintain the right balance between cost decrease and constraint satisfaction. We note that the line-search condition for low constraint violation is typically not achieved when using only one iteration per MPC problem.

6.7.2.2 Model Selection

In the same scenario, we compare the performance of the proposed dynamics for the base with those of the commonly used single rigid body dynamics (SRBD). To be precise, the torso dynamics in (6.8) are evaluated at a constant nominal joint configuration and with zero joint velocities, while the rest of the controller remains identical. When using the SRBD, the model does not describe the backward shift in the center of mass location caused by the leg

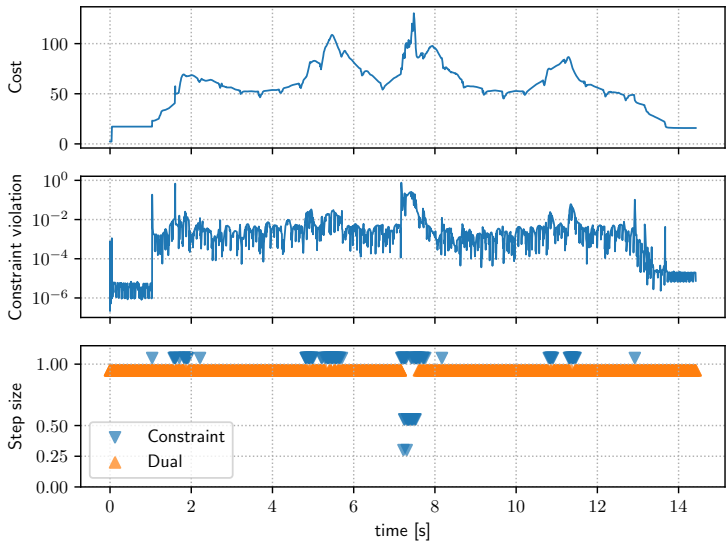


Figure 6.8: Solver status during the box traversal motion (including knee collision avoidance). The first and second plots show the total cost, and constraint violation according to (6.36), after each iteration. The bottom plot shows the stepsize and the line-search branch that led to the step acceptance. 'Constraint' refers to a step accepted in the high constraint violation branch in line 9 of Algorithm 4, 'Dual' refers to the branch where either cost or constraint decrease is accepted in line 18.

configuration. The result is that the controller with the SRBD model has a persisting bias that makes the robot almost tip over during the step up. The proposed model fully describes the change in inertia and center of mass location and therefore does not have any issue in predicting the state trajectory during the step up motion.

6.7.2.3 Contact Feedback

The reactive behavior under a mismatch in planned and sensed contact information is shown in the accompanying video. First, the sensed terrain is set to be 10 cm above the actual terrain, causing a late touchdown. Afterward, the sensed terrain is set 5 cm below the actual terrain, causing an early touchdown. The resulting vertical foot velocity for both cases is overlayed and plotted in Fig. 6.9. For the case of a late touchdown, the reactive downward accelerating trajectory is triggered as soon as it is sensed that contact is absent. For the early touchdown case, there is a short delay in detecting that contact has

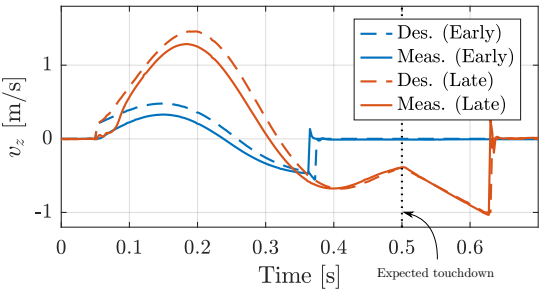


Figure 6.9: Desired and measured vertical foot velocity for the early and late touchdown scenarios shown in the accompanying video. The vertical line at 0.5 s indicated the planned touchdown time.

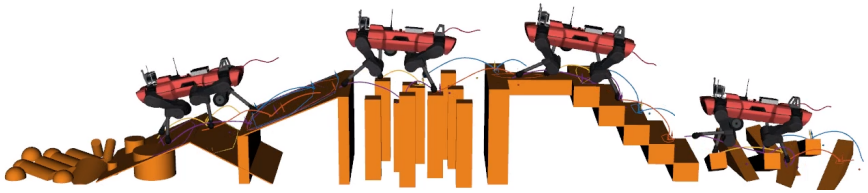


Figure 6.10: ANYmal traversing an obstacle course in simulation. Snapshots are shown for a the traversal with a trotting gait at 0.8 m/s. The MPC predictions are shown for each foot and for the torso center. For all contact phases within the horizon, the convex foot placements contraints are visualized.

happened, but once contact is detected, the measured contact is included in the MPC and the new trajectory is immediately replanned from the sensed contact location.

6.7.2.4 Stairs

The generality of the approach with respect to the gait pattern is demonstrated in the accompanying video by executing a trot at 0.25 m/s, a pace at 0.3 m/s, a dynamic walk at 0.25 m/s, and a static walk at 0.2 m/s on a stairs with 18.5 cm rise and 24 cm run. Depending on the particular gait pattern and commanded velocity the method autonomously decides to progress, repeat, or skip a step. Note that there are no parameters or control modes specific to the gait or the stair climbing scenario. All motions emerge automatically from the optimization of the formulated costs and constraints.

6.7.2.5 Obstacle Course

The controller is given a constant forward velocity command on a series of slopes, gaps, stepping stones, and other rough terrains. We traverse the terrain with a pace at 0.4 m/s, and a fast trotting gait with flight phase at 0.8 m/s. Fig. 6.10 shows the obstacle course and snapshots of the traversal with the fast trot. The supplemental video shows the planned trajectories for the feet together with the convex foothold constraints. In the right side of the screen, a front view is shown together with the elevation map and plane segmentation below. The slower gaits used in the previous section are able to complete the scenario as well, but their video is excluded as they take long to reach the end.

Finally, a transverse gallop gait is demonstrated on a series of gaps. Due to the torque limitations of the system and friction limits up the slope, this gait is not feasible on the more complex obstacle course.

6.7.2.6 Comparison against RL

We compare our method against a perceptive RL-based controller (Miki et al., 2022a) in the same obstacle course. We adapt the gait pattern of our controller to match the nominal gait used by the learned controller. The video shows that the learning-based controller can cross the unstructured terrain at the beginning and end of the obstacle course. However, it fails to use the perceptive information fully and falls between the stepping stones when starting from the left and off the narrow passage when starting from the right. This experiment highlights that current RL-based locomotion results in primarily reactive policies and struggles with precise coordination and planning over longer horizons. In contrast, using a model and online optimization along a horizon makes our proposed method generalize naturally to these more challenging terrains.

6.7.3 Hardware

6.7.3.1 Obstacle Course

The obstacle course simulation experiment is recreated on hardware in two separate experiments. First, we tested a sequence of a ramp, gap, and high step as shown in Fig. 6.11. During the middle section of this experiment, the robot faces all challenges simultaneously: While the front legs are stepping up to the final platform, the hind legs are still dealing with the ramp and gap. In a second scenario, the robot is walking on a set of uneven stepping stones, as shown in Fig. 6.12. The main challenge here is that the planes on the stepping

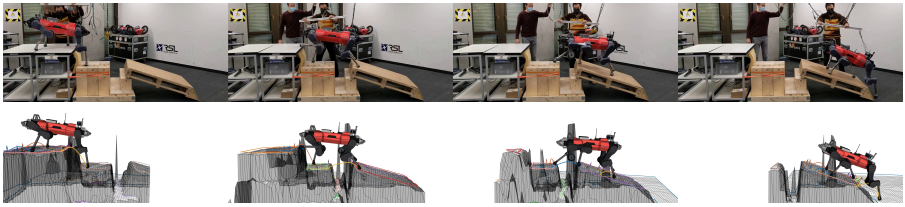


Figure 6.11: Hardware experiment where ANYmal traverses a ramp, gap, and large step (from right to left). The bottom row shows the filtered elevation map, the foot trajectories over the MPC horizon, and the convex foothold constraints.

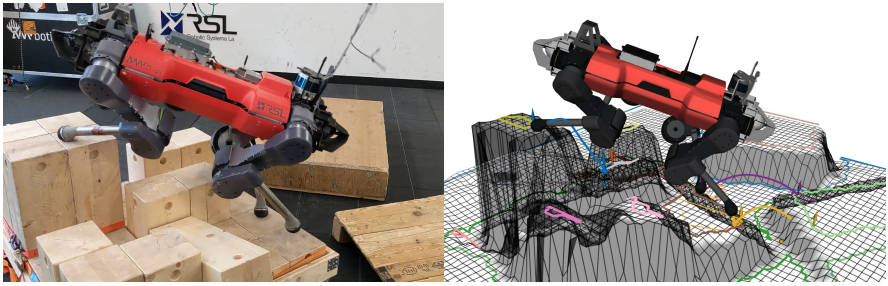


Figure 6.12: Hardware experiment where ANYmal walks on top of uneven stepping stones. Each wooden block has an area of 20x20 cm and each level of stepping stones is 20 cm higher than the previous one. The right image shows the filtered elevation map, the foot trajectories over the MPC horizon, and the convex foothold constraints.

stones are small and do not leave much room for the MPC to optimize the footholds. We found that in this scenario, the inclusion of the kinematics and reactive foothold offset during the plane selection as described in section 6.4.5 are important. A remaining challenge here is that our plane segmentation does not consider consistency over time. In some cases, the small foothold regions on top of stepping stones might appear and disappear as feasible candidates. The supplemental video shows how in this case the planned foot trajectory can fail, and the reactive contact regaining is required to save the robot.

Computation times are reported in Table 6.2. Per map update, most time is spent on terrain classification and plane segmentation. More specifically, the RANSAC refinement takes the most time and can cause a high worst-case computation due to its sampling-based nature. On average, the perception pipeline is able to keep up with the 20 Hz map updates.

Table 6.2: COMPUTATION TIMES PER MAP UPDATE AND MPC ITERATION

	Mean [ms]	Max [ms]
Classification & Segmentation	38.8	76.6
Signed distance field	1.3	7.6
LQ approximation	3.6	6.2
QP solve	2.7	4.4
Line-search	0.3	0.9
MPC iteration	6.6	9.8

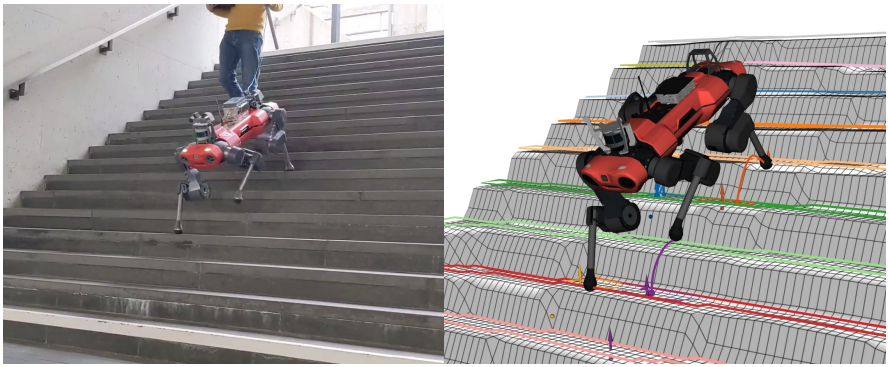


Figure 6.13: Hardware experiment where ANYmal walks up and down outdoor stairs with a 16 cm rise and 29.5 cm run. The right image shows the filtered elevation map, the foot trajectories over the MPC horizon, and the convex foothold constraints.

For the MPC computation time, the 'LQ approximation' contains the parallel computation of the linear-quadratic model and equality constraint projection (Algorithm 3, line 2 till 4). 'QP solve' contains the solution of the QP and the back substitution of the solution (Algorithm 3, line 5 and 6). Despite the parallelization across four cores, evaluating the model takes the majority of the time, with the single core solving of the QP in second place. On average, the total computation time is sufficient for the desired update rate of 100 Hz. The worst-case computation times are rare, and we hypothesize that they are mainly caused by variance in the scheduling of the numerous parallel processes on the robot. For the line-search, the relatively high maximum computation time is attained when several steps are rejected, and the costs and constraints need to be recomputed.

6.7.3.2 Stairs

We validate the stair climbing capabilities on 2-step indoor stairs and on outdoor stairs. Fig. 6.13 shows the robot on its way down the outdoor stairs. For these experiments, we obtain the elevation map from (Hoeller et al., 2022). With its learning-based approach, it provides a high quality estimate of the structure underneath the robot.

6.7.4 Limitations

A fundamental limitation in the proposed controller is that the gait pattern is externally given and only adapted during early and late touchdown. Strong adverse disturbances, for example, in the direction of a foot that will soon lift, can make the controller fail. A change in the stepping pattern could be a much better response in such cases. Together with the reactive behaviors during contact mismatch, which are currently hardcoded, we see the potential for reinforcement learning-based methods as a tracking controller to add to the robustness during execution.

Closely related to that, the current selection of the segmented plane and, therefore, the resulting foothold constraints happens independently for each leg. In some cases, this can lead to problems that could have been avoided if all legs were considered simultaneously. For example, while walking up the stairs sideways, all feet can end up on the same tread, leading to fragile support and potential self-collisions.

As with all gradient-based methods for nonlinear optimization, local optima and infeasibility can be an issue. With the simplification of the terrain to convex foothold constraints and by using a heuristic reference motion in the cost function, we have aimed to minimize such problems. Still, we find that in the case of very thin and tall obstacles, the optimization can get stuck. Fig. 6.14 shows an example where the foothold constraints lie behind the obstacle and the reference trajectory correctly clears the obstacle. Unfortunately, one of the feet in the MPC trajectory goes right through the obstacle. Because all SDF gradients are horizontal at that part of the obstacle, there is no strong local hint that the obstacle can be avoided. For future work, we can imagine detecting such a case and triggering a sampling-based recovery strategy to provide a new, collision-free initial guess. Alternatively, recent learning-based initialization could be employed (Lembono et al., 2020; Melon et al., 2021).

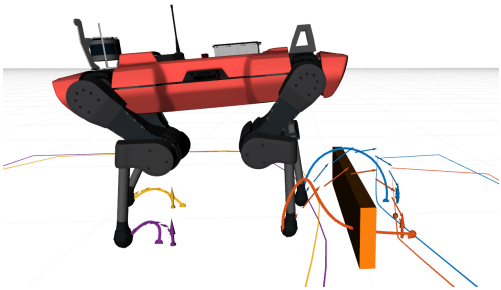


Figure 6.14: Example of the MPC optimization being stuck inside a tall and thin structure of 5 cm width and 20 cm height. The feet reference trajectories used as part of the cost function are visualized as a sequence of arrows.

6.8 Conclusion

In this work, we proposed a controller capable of perceptive and dynamic locomotion in challenging terrain. By formulating perceptive foot placement constraints through a convex inner approximation of steppable terrain, we obtain a nonlinear MPC problem that can be solved reliably and efficiently with the presented numerical strategy. Steppability classification, plane segmentation, and an SDF are all precomputed and updated at 20 Hz. Asynchronously precomputing this information minimizes the time required for each MPC iteration and makes the approach real-time capable. Furthermore, by including the complete joint configuration in the system model, the method can simultaneously optimize foot placement, knee collision avoidance, and underactuated system dynamics. With this rich set of information encoded in the optimization, the approach discovers complex motions autonomously and generalizes across various gaits and terrains that require precise foot placement and whole-body coordination.

6.9 Appendix A: Signed Distance Field Computation

This section details how a signed distance field can be computed for a 2.5D elevation map. Consider the following general definition for the squared Euclidean distance between a point in space and the closest obstacle:

$$\mathcal{D}(x,y,z) = \min_{x',y',z'} \left[(x-x')^2 + (y-y')^2 + (z-z')^2 + I(x',y',z') \right] \quad (6.40)$$

where $I(x',y',z')$ is an indicator function returning 0 for an obstacle and ∞ for empty cells.

As described in (Felzenszwalb and Huttenlocher, 2012), a full 3D distance transform can be computed by consecutive distance transforms in each dimension of the grid, in arbitrary order. For the elevation map, the distance along the z-direction is trivial. Therefore, starting the algorithm with the z-direction simplifies the computation. First, (6.40) can be rewritten as follow,

$$\begin{aligned} \mathcal{D}(x,y,z) &= \min_{x',y'} \left[(x-x')^2 + (y-y')^2 + \min_{z'} \left[(z-z')^2 + I(x',y',z') \right] \right], \\ &= \min_{x',y'} \left[(x-x')^2 + (y-y')^2 + f_z(x',y',z) \right], \end{aligned} \quad (6.41)$$

where $f_z(x',y',z)$ is a function that returns for each horizontal position, the one-dimensional distance transform in z-direction. For an elevation map, this function has the following closed form solution at a given height z.

$$f_z(x',y',z) = \begin{cases} (z - h(x',y'))^2 & \text{if } z \geq h(x',y'), \\ 0 & \text{otherwise,} \end{cases} \quad (6.42)$$

where $h(x',y')$ denotes the evaluation of the elevation map.

The same idea can be used to compute the distance to obstacle free space and obtain the negative valued part of the SDF. Adding both distances together provides the full SDF and gradients are computed by finite differences between layers, columns, and rows. However, naively taking the Euclidean distance between cell centers as the minimization of (6.41) leads to incorrect values around obstacle borders, as illustrated in Fig. 6.15. We need to account for the fact that the obstacle border is located between cells, not at the cell locations

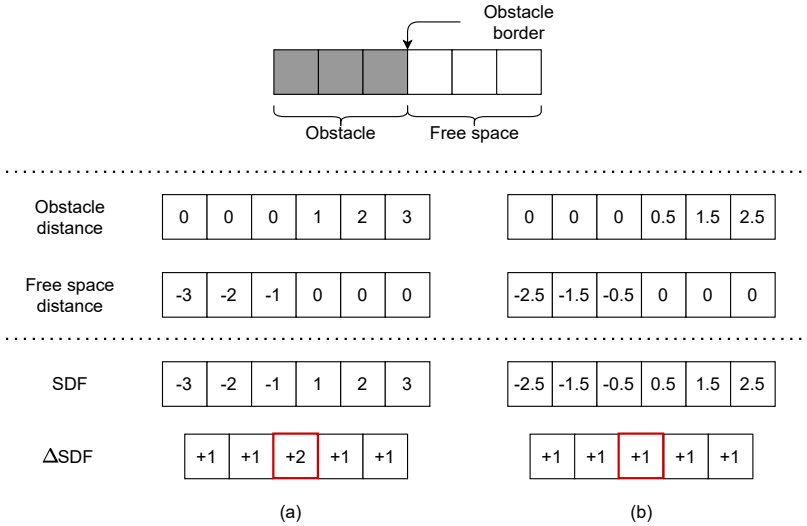


Figure 6.15: 1D example illustrating the effect of distance metric on the SDF. When taking the Euclidean distance between cell centers in (a), the SDF will have a discontinuous gradient across the obstacle border. Taking the distance between cell center and the border of an occupied / free cell as in (b), avoids this issue.

themselves. This can be resolved by adapting (6.41) to account for the discrete nature of the problem.

$$\mathcal{D}(x, y, z) = \min_{\{x', y'\} \in \mathcal{M}} \left[d\left(x, x'\right) + d\left(y, y'\right) + f_z(x', y', z) \right], \tag{6.43}$$

where $\{x', y'\} \in \mathcal{M}$ now explicitly shows that we only minimize over the discrete cells contained in the map, and $d(\cdot, \cdot)$ is a function that returns the squared distance between the center of one cell and the border of another:

$$d(x, x') = \begin{cases} \left(|x - x'| - 0.5r\right)^2 & \text{if } x \neq x', \\ 0 & \text{otherwise,} \end{cases} \tag{6.44}$$

where r is the resolution of the map. The distance transforms can now be computed based on (6.43), for each height in parallel, with the 2D version of the algorithm described in (Felzenszwalb and Huttenlocher, 2012).

7

Conclusion and Outlook

Throughout this dissertation, we have studied and developed optimization-based motion control approaches to generate dynamic and complex maneuvers for legged robots in rough terrain. In contrast to existing work, we have formulated the locomotion problem as a single optimization problem. This approach removed the need for heuristic coordination of legs and torso and has opened the door to more complex, whole-body motions operating at the robot's limits. The contributions of each individual chapter have increased our understanding of how such a complex optimization problem can be formulated to achieve the desired closed-loop performance, stability, safety, and computational complexity. While most demonstrations have been made on legged robots, the underlying theory, algorithms, and software tools are general and can be applied to other dynamical systems.

7.1 Achievements

In Chapter 2, we introduced frequency-dependent cost functions into the Model Predictive Control (MPC) formulation. Using this method to penalize control inputs in the high-frequency spectrum was shown to provide robustness against unmodeled dynamics of series elastic actuators and compliant terrain. The resulting smooth contact force profiles improved tracking performance on various terrains where the rigid contact assumption is violated, without the need to explicitly model it. Moreover, this was the first time we were able to deploy an MPC formulation that reasons about all degrees of freedom of the robot simultaneously. The computations, however, were still performed offline on a desktop machine, motivating us to focus on bridging the gap in computational efficiency.

In Chapter 3, we therefore proposed feedback MPC as an effective way to handle the low update rate associated with the computational restrictions of

mobile platforms. In this formulation, the feedback policy obtained from the Riccati backward pass of the Sequential Linear Quadratic (SLQ) algorithm was used to provide a first-order approximation to the true MPC problem at the frequency required for whole-body control. We experimentally showed that a combination with the frequency-aware approach of Chapter 2 was critical to achieving closed-loop stability on hardware. Additionally, we proposed a relaxed barrier function method to extend the SLQ algorithm to optimization problems with inequality constraints. While this was a practical consideration at that time, it turned out to be highly effective and is still used in many projects today.

From there, we turned our attention to a more theoretical side of the problem during the collaboration with the AMBER lab, resulting in the two publications in Chapters 4 and 5. In Chapter 4, we presented a novel set of approaches for unifying Control Lyapunov Functions (CLFs) and Nonlinear Model Predictive Control (NMPC) for continuous, control-affine systems. The presented CLF-NMPC methods explicitly embed stability as a constraint in the optimization problem and therefore required no further tuning of the cost function and prediction horizon to achieve closed-loop stability. At the same time, compared to purely CLF-based approaches, which are point-wise optimal in time, the addition of the prediction horizon significantly improves the long-term performance. The approach also proved excellent performance in practice and resulted in the first demonstration of a CLF-based MPC controller on robotic hardware.

Extending on this idea, we combined Control Barrier Functions (CBFs) with MPC in Chapter 5 to provide closed-loop safety guarantees. Additionally, this work proposed to embed CBFs both in the MPC formulation and the whole-body tracking controller to provide consistent safety constraints across the different execution frequencies and model complexities. We validated the viability of the approach on hardware by demonstrating dynamic locomotion on stepping-stones with safety constraints.

Finally, in Chapter 6, we propose an MPC formulation that incorporates onboard perceptive terrain information and enables dynamic locomotion across challenging terrain. An elevation map representation with steppability classification, plane segmentation, and a corresponding Signed Distance Field (SDF) are all updated online. By formulating perceptive foot placement constraints through a convex inner approximation of steppable terrain, we obtained a nonlinear MPC problem that can be solved reliably and efficiently with the presented numerical strategy. In contrast to the other works in Chapters 2, 3,

and 5 in this thesis, which relied on the single shooting algorithm SLQ, we use a multiple-shooting technique instead. This alternative solving strategy proved to be more robust for the more demanding optimization problems formulated in Chapter 6. The ultimate hardware demonstrations showed precise foot placement and whole-body coordination across gaps, slopes, and stepping stones, resulting in state-of-the-art dynamic climbing. At the time of writing, these results have not been achieved by any other real-time capable method.

7.2 Future Work

Since the start of this thesis, the field of legged robots has come a long way, and the technology is now mature enough to be deployed in commercial applications. Still, we see three specific areas in which further research can progress and build on the findings and contributions made in this thesis.

7.2.1 Autonomy

There is a need to bridge the gap between navigation and exploration modules and the dynamic locomotion controllers developed in this thesis. The controllers in this work have used a prediction horizon of typically between one and two seconds. While this has proven sufficient for most gaits and maneuvers, some scenarios might require reasoning over longer timescales. Additionally, high-level direction and velocity commands were given by a human operator, often with a good intuition of what the underlying controllers can and cannot do. When moving towards a full-stack autonomous solution in the future, the limitations of the closed-loop system need to be incorporated in a more principled way. We have started the theoretical investigation into this multi-rate view on the problem in Chapter 5, and we see an excellent opportunity for more fundamental research in this direction that has become increasingly relevant (Csomay-Shanklin et al., 2022; Rosolia and Ames, 2021).

7.2.2 Connection with Data-driven Approaches

During the time this Ph.D. research was conducted, there has been tremendous progress in applying Reinforcement Learning (RL) to legged robots. Interestingly, similar to our work on frequency-aware MPC, one of the key elements that enabled successful hardware deployment was the consideration of the imperfect actuator dynamics (Hwangbo et al., 2019). Ultimately, RL and MPC start from a similar optimization-based view on the control problem, and it

should therefore not come as a surprise that they share a number of similarities. However, the most recent work in this direction (Miki et al., 2022b) showed the incredible robustness that can be achieved with the right randomization and curriculum of terrain in simulation. This robustness with respect to inaccurate contact estimation and noise in the perceptive information is something that is still missing in the MPC controllers proposed in this work.

In future work, RL could play a role in providing individual elements within the MPC problem, for example, the cost function or system dynamics, and provide a smart filter between the noisy world and the deterministic optimization. The other way around, the methods proposed in this thesis could serve as a demonstrator during the offline training process. Currently, the training process can take a long time and often converges to a purely reactive strategy, which struggles in more complex scenarios like stepping stones. MPC could help in discovering coordinated long horizon behaviors in a more principled way.

7.2.3 Aperiodic Locomotion with Stability Guarantees

In Chapter 4, we studied the combination of CLF-based stability constraints and MPC for continuous, control-affine systems. Additional work is needed to extend the formulations in that chapter to the hybrid domain of legged robots. Here, the Hybrid Zero Dynamics (HZD) framework provides a principled way to derive CLFs for periodic gaits (Nguyen et al., 2016). In recently submitted work, we have investigated how these periodic, offline solutions can be leveraged inside an MPC formulation (Galliker et al., 2022). The theoretical properties of using HZD trajectories in the terminal components of the optimization problem will need to be studied in future work. This is a promising direction to potentially extend the stability guarantees currently available for offline, periodic motions to general, aperiodic motions optimized online.

Bibliography

Aceituno-Cabezas, B., Mastalli, C., Dai, H., Focchi, M., Radulescu, A., Caldwell, D. G., Cappelletto, J., Grieco, J. C., Fernández-López, G., and Semini, C. (2018). "Simultaneous Contact, Gait, and Motion Planning for Robust Multilegged Locomotion via Mixed-Integer Convex Optimization". *IEEE Robotics and Automation Letters* 3.3, pp. 2531–2538.

Aguiar, A. P., Bayer, F. A., Hauser, J., Häusler, A. J., Notarstefano, G., Pascoal, A. M., Rucco, A., and Saccon, A. (2017). "Constrained Optimal Motion Planning for Autonomous Vehicles Using PRONTO". In: *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*. Ed. by T. I. Fossen, K. Y. Pettersen, and H. Nijmeijer. Springer International Publishing, pp. 207–226.

Alexis, K., Papachristos, C., Nikolakopoulos, G., and Tzes, A. (2011). "Model predictive quadrotor indoor position control". In: *2011 19th Mediterranean Conference on Control Automation (MED)*, pp. 1247–1252.

Ambrose, E., Ma, W.-L., Hubicki, C., and Ames, A. D. (2017). "Toward benchmarking locomotion economy across design configurations on the modular robot: AMBER-3M". In: *Conference on Control Technology and Applications (CCTA)*. IEEE, pp. 1270–1276.

Ames, A. D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., and Tabuada, P. (2019). "Control barrier functions: Theory and applications". In: *European Control Conference (ECC)*. IEEE, pp. 3420–3431.

Ames, A. D., Galloway, K., Sreenath, K., and Grizzle, J. W. (2014). "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics". *IEEE Transactions on Automatic Control* 59.4, pp. 876–891.

Ames, A. D. and Powell, M. (2013). "Towards the unification of locomotion and manipulation through control lyapunov functions and quadratic programs". In: *Control of Cyber-Physical Systems*. Springer, pp. 219–240.

Ames, A. D., Xu, X., Grizzle, J. W., and Tabuada, P. (2017). "Control barrier function based quadratic programs for safety critical systems". *Transactions on Automatic Control* 62.8, pp. 3861–3876.

Ames, A. D., Grizzle, J. W., and Tabuada, P. (2014). "Control barrier function based quadratic programs with application to adaptive cruise control". In: *53rd IEEE Conference on Decision and Control*, pp. 6271–6278.

Anderson, B. and Mingori, D. (1985). "Use of frequency dependence in linear quadratic control problems to frequency-shape robustness". *Journal of Guidance, Control, and Dynamics* 8.3, pp. 397–401.

Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). "CasADi – A software framework for nonlinear optimization and optimal control". *Mathematical Programming Computation* 11.1, pp. 1–36.

ANYbotics (2022). *ANYmal, the autonomous legged robot for industrial maintenance tasks*. `https://www.anybotics.com/anymal-autonomous-legged-robot/`. Accessed: 2022-04-03.

Arreguit, J., Faraji, S., and Ijspeert, A. J. (2018). "Fast Multi-Contact Whole-Body Motion Planning with Limb Dynamics". In: *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 1–9.

Artstein, Z. (1983). "Stabilization with relaxed controls". *Nonlinear Analysis: Theory, Methods & Applications* 7.11, pp. 1163–1173.

Atkeson, C. G., Benzun, P., Banerjee, N., Berenson, D., Bove, C. P., Cui, X., DeDonato, M., Du, R., Feng, S., Franklin, P., et al. (2018). "What happened at the DARPA robotics challenge finals". In: *The DARPA robotics challenge finals: Humanoid robots to the rescue*. Springer, pp. 667–684.

Bajracharya, M., Ma, J., Malchano, M., Perkins, A., Rizzi, A. A., and Matthies, L. (2013). "High fidelity day/night stereo mapping with vegetation and negative obstacle detection for vision-in-the-loop walking". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3663–3670.

Bazeille, S., Barasuol, V., Focchi, M., Havoutis, I., Frigerio, M., Buchli, J., Caldwell, D. G., and Semini, C. (2014). "Quadruped robot trotting over irregular terrain assisted by stereo-vision". *Intelligent Service Robotics* 7.2, pp. 67–77.

Bellicoso, C. D., Jenelten, F., Gehring, C., and Hutter, M. (2018a). "Dynamic Locomotion Through Online Nonlinear Motion Optimization for Quadrupedal Robots". *IEEE Robotics and Automation Letters* 3.3, pp. 2261–2268.

Bellicoso, C. D., Bjelonic, M., Wellhausen, L., Holtmann, K., Günther, F., Tranzatto, M., Fankhauser, P., and Hutter, M. (2018b). "Advances in real-world applications for legged robots". *Journal of Field Robotics* 35.8, pp. 1311–1326.

Belter, D., Bednarek, J., Lin, H.-C., Xin, G., and Mistry, M. (2019). "Single-shot Foothold Selection and Constraint Evaluation for Quadruped Locomotion". In: *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7441–7447.

Belter, D., Labecki, P., and Skrzypczynski, P. (2016). "Adaptive Motion Planning for Autonomous Rough Terrain Traversal with a Walking Robot". *Journal of Field Robotics* 33.3, pp. 337–370.

Bemporad, A. and Morari, M. (1999). "Robust model predictive control: A survey". In: *Robustness in identification and control*. Ed. by A. Garulli and A. Tesi. Springer London, pp. 207–226.

Bertrand, S., Lee, I., Mishra, B., Calvert, D., Pratt, J., and Griffin, R. (2020). "Detecting Usable Planar Regions for Legged Robot Locomotion". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4736–4742.

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific.

Bjelonic, M., Grandia, R., Geilinger, M., Harley, O., Medeiros, V. S., Pajovic, V., Jelavic, E., Coros, S., and Hutter, M. (2022). "Offline motion libraries and online MPC for advanced mobility skills". *The International Journal of Robotics Research*.

Bjelonic, M., Grandia, R., Harley, O., Galliard, C., Zimmermann, S., and Hutter, M. (2021). "Whole-Body MPC and Online Gait Sequence Generation for Wheeled-Legged Robots". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8388–8395.

Bledt, G., Wensing, P. M., and Kim, S. (2017). "Policy-regularized model predictive control to stabilize diverse quadrupedal gaits for the MIT cheetah". In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4102–4109.

Bledt, G., Wensing, P. M., Ingersoll, S., and Kim, S. (2018). "Contact Model Fusion for Event-Based Locomotion in Unstructured Terrains". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4399–4406.

Blickhan, R. (1989). "The spring-mass model for running and hopping". *Journal of Biomechanics* 22.11, pp. 1217–1227.

Bock, H. G. (1983). "Recent advances in parameteridentification techniques for ode". In: *Numerical treatment of inverse problems in differential and integral equations*. Springer, pp. 95–121.

Bock, H. and Plitt, K. (1984). "A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems". *IFAC Proceedings Volumes* 17.2, pp. 1603 –1608.

Boston Dynamics (2022). *Spot - Automate sensing and inspection, capture limitless data, and explore without boundaries.* `https://www.bostondynamics.com/products/spot`. Accessed: 2022-04-03.

Bouman, A., Ginting, M. F., Alatur, N., Palieri, M., Fan, D. D., Touma, T., Pailevanian, T., Kim, S.-K., Otsu, K., Burdick, J., et al. (2020). "Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2518–2525.

Braun, D. J., Petit, F., Huber, F., Haddadin, S., Smagt, P. van der, Albu-Schäffer, A., and Vijayakumar, S. (2013). "Robots Driven by Compliant Actuators: Optimal Control Under Actuation Constraints". *IEEE Transactions on Robotics* 29.5, pp. 1085–1101.

Buchanan, R., Wellhausen, L., Bjelonic, M., Bandyopadhyay, T., Kottege, N., and Hutter, M. (2021). "Perceptive whole-body planning for multilegged robots in confined spaces". *Journal of Field Robotics* 38.1, pp. 68–84.

Budhiraja, R., Carpentier, J., Mastalli, C., and Mansard, N. (2018). "Differential Dynamic Programming for Multi-Phase Rigid Contact Dynamics". In: *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 1–9.

Carius, J., Ranftl, R., Koltun, V., and Hutter, M. (2018). "Trajectory Optimization With Implicit Hard Contacts". *IEEE Robotics and Automation Letters* 3.4, pp. 3316–3323.

Carius, J., Ranftl, R., Farshidian, F., and Hutter, M. (2022). "Constrained stochastic optimal control with learned importance sampling: A path integral approach". *The International Journal of Robotics Research* 41.2, pp. 189–209.

Carius, J., Ranftl, R., Koltun, V., and Hutter, M. (2019). "Trajectory Optimization for Legged Robots With Slipping Motions". *IEEE Robotics and Automation Letters* 4.3, pp. 3013–3020.

Caron, S., Pham, Q. C., and Nakamura, Y. (2015). "Leveraging Cone Double Description for Multi-contact Stability of Humanoids with Applications to Statics and Dynamics". In: *Proceedings of Robotics: Science and Systems*.

Carpentier, J. and Mansard, N. (2018). "Multicontact Locomotion of Legged Robots". *IEEE Transactions on Robotics* 34.6, pp. 1441–1460.

Carpentier, J., Saurel, G., Buondonno, G., Mirabel, J., Lamiraux, F., Stasse, O., and Mansard, N. (2019). "The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives". In: *IEEE International Symposium on System Integrations (SII)*.

Chang, A. H., Hubicki, C. M., Aguilar, J. J., Goldman, D. I., Ames, A. D., and Vela, P. A. (2017). "Learning to jump in granular media: Unifying optimal control synthesis with Gaussian process-based regression". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2154–2160.

Chilian, A. and Hirschmüller, H. (2009). "Stereo camera based navigation of mobile robots on rough terrain". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4571–4576.

Choi, J., Castañeda, F., Tomlin, C., and Sreenath, K. (2020). "Reinforcement Learning for Safety-Critical Control under Model Uncertainty, using Control Lyapunov Functions and Control Barrier Functions". In: *Robotics: Science and Systems (RSS)*.

Csomay-Shanklin, N., Taylor, A. J., Rosolia, U., and Ames, A. D. (2022). *Multi-Rate Planning and Control of Uncertain Nonlinear Systems: Model Predictive Control and Control Lyapunov Functions*.

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

Dantec, E., Budhiraja, R., Roig, A., Lembono, T., Saurel, G., Stasse, O., Fernbach, P., Tonneau, S., Vijayakumar, S., Calinon, S., Taix, M., and Mansard, N. (2021). "Whole Body Model Predictive Control with a Memory of Motion: Experiments on a Torque-Controlled Talos". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8202–8208.

Dario Bellicoso, C., Gehring, C., Hwangbo, J., Fankhauser, P., and Hutter, M. (2016). "Perception-less terrain adaptation through whole body control and hierarchical optimization". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE-RAS, pp. 558–564.

DARPA (2015a). *A celebration of risk (a.k.a., robots take a spill)*. https://www.youtube.com/watch?v=7A_QPGcjrh0. Accessed: 2022-04-03.

– (2015b). *DARPA Robotics Challenge*. https://www.darpa.mil/program/darpa-robotics-challenge. Accessed: 2022-04-03.

de Oliveira Kothare, S. L. and Morari, M. (2000). "Contractive model predictive control for constrained nonlinear systems". *IEEE Transactions on Automatic Control* 45.6, pp. 1053–1071.

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

Deits, R. and Tedrake, R. (2015). "Computing Large Convex Regions of Obstacle-Free Space Through Semidefinite Programming". In: *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*. Ed. by H. L. Akin, N. M. Amato, V. Isler, and A. F. van der Stappen. Springer International Publishing, pp. 109–124.

Di Carlo, J., Wensing, P. M., Katz, B., Bledt, G., and Kim, S. (2018). "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control". In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 1–9.

Diehl, M., Bock, H., and Schlöder, J. (2005). "A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control". *SIAM Journal on Control and Optimization* 43.5, pp. 1714–1736.

Doyle, J. and Stein, G. (1981). "Multivariable feedback design: Concepts for a classical/modern synthesis". *IEEE transactions on Automatic Control* 26.1, pp. 4–16.

Dyana (2022). *Dyana - a Dynamic Animatronic Robot*. https://dyana.ethz.ch/. Accessed: 2022-05-11.

Englsberger, J. and Ott, C. (2012). "Integration of vertical COM motion and angular momentum in an extended Capture Point tracking controller for bipedal walking". In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pp. 183–189.

Englsberger, J., Ott, C., and Albu-Schäffer, A. (2015). "Three-dimensional bipedal walking control based on divergent component of motion". *Ieee transactions on robotics* 31.2, pp. 355–368.

Fankhauser, P., Bjelonic, M., Bellicoso, D., Miki, T., and Hutter, M. (2018). "Robust Rough-Terrain Locomotion with a Quadrupedal Robot". In: *International Conference on Robotics and Automation (ICRA)*. IEEE.

Fankhauser, P., Bloesch, M., and Hutter, M. (2018). "Probabilistic Terrain Mapping for Mobile Robots with Uncertain Localization". *IEEE Robotics and Automation Letters (RA-L)* 3.4, pp. 3019–3026.

Fankhauser, P. and Hutter, M. (2016). "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation". In: *Robot Operating System (ROS) – The Complete Reference (Volume 1)*. Ed. by A. Koubaa. Springer. Chap. 5.

Farshidian, F., Neunert, M., Winkler, A. W., Rey, G., and Buchli, J. (2017a). "An efficient optimal planning and control framework for quadrupedal locomotion". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 93–100.

Farshidian, F., Jelavic, E., Satapathy, A., Giftthaler, M., and Buchli, J. (2017b). "Real-time motion planning of legged robots: A model predictive control approach". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 577–584.

Farshidian, F., Kamgarpour, M., Pardo, D., and Buchli, J. (2017c). "Sequential linear quadratic optimal control for nonlinear switched systems". *IFAC-PapersOnLine* 50.1, pp. 1463–1469.

Featherstone, R. (2014). *Rigid body dynamics algorithms*. Springer.

Feller, C. and Ebenbauer, C. (2017). "A stabilizing iteration scheme for model predictive control based on relaxed barrier functions". *Automatica* 80, pp. 328 –339.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2012). "Distance transforms of sampled functions". *Theory of computing* 8.1, pp. 415–428.

Fletcher, R. and Leyffer, S. (2002). "Nonlinear programming without a penalty function". *Mathematical programming* 91.2, pp. 239–269.

Focchi, M., Medrano-Cerda, G. A., Boaventura, T., Frigerio, M., Semini, C., Buchli, J., and Caldwell, D. G. (2016). "Robot impedance control and passivity analysis with inner torque and velocity feedback loops". *Control Theory and Technology* 14.2, pp. 97–112.

Freeman, R. A. and Kokotović, P. V. (1996). "Inverse optimality in robust stabilization". *SIAM journal on control and optimization* 34.4, pp. 1365–1391.

Frison, G. and Diehl, M. (2020). "HPIPM: a high-performance quadratic programming framework for model predictive control". *IFAC-PapersOnLine* 53.2. 21st IFAC World Congress, pp. 6563–6569.

Gaertner, M., Bjelonic, M., Farshidian, F., and Hutter, M. (2021). "Collision-free MPC for legged robots in static and dynamic scenes". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8266–8272.

Galliker, M. Y., Csomay-Shanklin, N., Grandia, R., Taylor, A. J., Farshidian, F., Hutter, M., and Ames, A. D. (2022). *Bipedal Locomotion with Nonlinear Model Predictive Control: Online Gait Generation using Whole-Body Dynamics.*

Galloway, K., Sreenath, K., Ames, A. D., and Grizzle, J. W. (2015). "Torque Saturation in Bipedal Robotic Walking Through Control Lyapunov Function-Based Quadratic Programs". *IEEE Access* 3, pp. 323–332.

Gangapurwala, S., Geisert, M., Orsolino, R., Fallon, M., and Havoutis, I. (2022). "RLOC: Terrain-aware legged locomotion using reinforcement learning and optimal control". *IEEE Transactions on Robotics.*

Garcia, C. E., Prett, D. M., and Morari, M. (1989). "Model predictive control: theory and practice—a survey". *Automatica* 25.3, pp. 335–348.

Gehring, C., Bellicoso, C. D., Coros, S., Bloesch, M., Fankhauser, P., Hutter, M., and Siegwart, R. (2015). "Dynamic trotting on slopes for quadrupedal robots". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5129–5135.

Gehring, C., Fankhauser, P., Isler, L., Diethelm, R., Bachmann, S., Potz, M., Gerstenberg, L., and Hutter, M. (2021). "ANYmal in the field: Solving industrial inspection of an offshore HVDC platform with a quadrupedal robot". In: *Field and Service Robotics.* Springer, pp. 247–260.

Geyer, H., Seyfarth, A., and Blickhan, R. (2006). "Compliant leg behaviour explains basic dynamics of walking and running". *Proceedings of the Royal Society B: Biological Sciences* 273.1603, pp. 2861–2867.

Ghost Robotics (2022). *Vision 60 - a mid-sized high-endurance, agile and durable all-weather ground drone for use in a broad range of unstructured urban and natural environments for defense, homeland and enterprise applications.* `https://www.ghostrobotics.io/vision-60`. Accessed: 2022-04-03.

Gill, P. E., Murray, W., and Saunders, M. A. (2005). "SNOPT: An SQP algorithm for large-scale constrained optimization". *SIAM review* 47.1, pp. 99–131.

Grandia, R., Pardo, D., and Buchli, J. (2018). "Contact Invariant Model Learning for Legged Robot Locomotion". *IEEE Robotics and Automation Letters* 3.3, pp. 2291–2298.

Grandia, R., Farshidian, F., Dosovitskiy, A., Ranftl, R., and Hutter, M. (2019a). "Frequency-aware model predictive control". *IEEE Robotics and Automation Letters* 4.2, pp. 1517–1524.

Grandia, R., Farshidian, F., Ranftl, R., and Hutter, M. (2019b). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4730–4737.

Grandia, R., Jenelten, F., Yang, S., Farshidian, F., and Hutter, M. (2022). "Perceptive Locomotion through Nonlinear Model Predictive Control". *(submitted to) IEEE Transactions on Robotics*.

Grandia, R., Taylor, A. J., Ames, A. D., and Hutter, M. (2021). "Multi-layered safety for legged robots via control barrier functions and model predictive control". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8352–8358.

Grandia, R., Taylor, A. J., Singletary, A., Hutter, M., and Ames, A. D. (2020). "Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions". In: *Proceedings of Robotics: Science and Systems*.

Griffin, R. J., Wiedebach, G., McCrory, S., Bertrand, S., Lee, I., and Pratt, J. (2019). "Footstep Planning for Autonomous Walking Over Rough Terrain". In: *International Conference on Humanoid Robots (Humanoids)*. IEEE-RAS, pp. 9–16.

Grimes, J. A. and Hurst, J. W. (2012). "The design of ATRIAS 1.0 a unique monopod, hopping robot". In: *Adaptive Mobile Robotics*. World Scientific, pp. 548–554.

Grizzle, J. W., Chevallereau, C., Sinnet, R. W., and Ames, A. D. (2014). "Models, feedback control, and open problems of 3D bipedal robotic walking". *Automatica* 50.8, pp. 1955–1988.

Grüne, L., Nešić, D., and Pannek, J. (2007). "Model Predictive Control for Nonlinear Sampled-Data Systems". In: *Assessment and Future Directions of Nonlinear Model Predictive Control*. Springer Berlin Heidelberg, pp. 105–113.

Grüne, L. and Pannek, J. (2017). "Nonlinear model predictive control". In: *Nonlinear Model Predictive Control*. Springer, pp. 45–69.

Gupta, N. K. (1980). "Frequency-shaped cost functionals-Extension of linear-quadratic-Gaussian design methods". *Journal of Guidance, Control, and dynamics* 3.6, pp. 529–535.

Gyurkovics, É. and Elaiw, A. M. (2004). "Stabilization of sampled-data nonlinear systems by receding horizon control via discrete-time approximations". *Automatica* 40.12, pp. 2017 –2028.

H. Chen and F. Allgöwer (1998). "A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability." *Automatica* 34.10, pp. 1205 –1217.

Hashimoto, K., Kang, H., Nakamura, M., Falotico, E., Lim, H., Takanishi, A., Laschi, C., Dario, P., and Berthoz, A. (2012). "Realization of biped walking on soft ground with stabilization control based on gait analysis". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2064–2069.

Hashimoto, K., Kondo, H., Lim, H.-O., and Takanishi, A. (2015). "Online Walking Pattern Generation Using FFT for Humanoid Robots". In: *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*. Ed. by G. Carbone and F. Gomez-Bravo. Springer International Publishing, pp. 417–438.

Hauser, J. and Saccon, A. (2006). "A Barrier Function Method for the Optimization of Trajectory Functionals with Constraints". In: *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 864–869.

Heidarinejad, M., Liu, J., and Christofides, P. D. (2012). "Economic model predictive control of nonlinear process systems using Lyapunov techniques". *AIChE Journal* 58.3, pp. 855–870.

Herbert, M., Caillas, C., Krotkov, E., Kweon, I., and Kanade, T. (1989). "Terrain mapping for a roving planetary explorer". In: *Proceedings, 1989 International Conference on Robotics and Automation*, 997–1002 vol.2.

Herdt, A., Diedam, H., Wieber, P.-B., Dimitrov, D., Mombaur, K., and Diehl, M. (2010). "Online walking motion generation with automatic footstep placement". *Advanced Robotics* 24.5-6, pp. 719–737.

Hereid, A., Cousineau, E. A., Hubicki, C. M., and Ames, A. D. (2016). "3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1447–1454.

Herzog, A., Schaal, S., and Righetti, L. (2016). "Structured contact force optimization for kino-dynamic motion generation". In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 2703–2710.

Hoeller, D., Rudin, N., Choy, C., Anandkumar, A., and Hutter, M. (2022). "Neural Scene Representation for Locomotion on Structured Terrain". *(submitted to) IEEE Robotics and Automation Letters*.

Hours, J.-H., Zeilinger, M. N., Gondhalekar, R., and Jones, C. N. (2015). "Constrained spectrum control". *IEEE Transactions on Automatic Control* 60.7, pp. 1969–1974.

Houska, B., Ferreau, H. J., and Diehl, M. (2011). "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range". *Automatica* 47.10, pp. 2279–2285.

Howell, T. A., Jackson, B. E., and Manchester, Z. (2019). "ALTRO: A Fast Solver for Constrained Trajectory Optimization". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7674–7679.

Hubicki, C., Grimes, J., Jones, M., Renjewski, D., Spröwitz, A., Abate, A., and Hurst, J. (2016a). "Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot". *The International Journal of Robotics Research* 35.12, pp. 1497–1521.

Hubicki, C. M., Aguilar, J. J., Goldman, D. I., and Ames, A. D. (2016b). "Tractable terrain-aware motion planning on granular media: An impulsive jumping study". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3887–3892.

Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C. D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., and Hoepflinger, M. (2016). "ANYmal - a highly mobile and dynamic quadrupedal robot". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 38–44.

Hutter, M., Gehring, C., Bloesch, M., Hoepflinger, M. A., Remy, C. D., and Siegwart, R. (2012). "StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion". In: *Adaptive Mobile Robotics*. World Scientific, pp. 483–490.

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., and Hutter, M. (2019). "Learning agile and dynamic motor skills for legged robots". *Science Robotics* 4.26.

Jacobson, D. H. and Mayne, D. Q. (1970). *Differential dynamic programming*. 24. Elsevier Publishing Company.

Jadbabaie, A. and Hauser, J. (2005). "On the stability of receding horizon control with a general terminal cost". *IEEE Transactions on Automatic Control* 50.5, pp. 674–678.

Jadbabaie, A., Yu, J., and Hauser, J. (2001). "Unconstrained receding-horizon control of nonlinear systems". *IEEE Transactions on Automatic Control* 46.5, pp. 776–783.

Jankovic, M. (2018). "Robust control barrier functions for constrained stabilization of nonlinear systems". *Automatica* 96, pp. 359–367.

Jenelten, F., Grandia, R., Farshidian, F., and Hutter, M. (2021). "TAMOLS: Terrain-Aware Motion Optimization for Legged Systems". *(submitted to) IEEE Transactions on Robotics*.

Jenelten, F., Miki, T., Vijayan, A. E., Bjelonic, M., and Hutter, M. (2020). "Perceptive Locomotion in Rough Terrain–Online Foothold Optimization". *IEEE Robotics and Automation Letters* 5.4, pp. 5370–5376.

Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., and Hirukawa, H. (2003). "Biped walking pattern generation by using preview control of zero-moment point". In: *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*. Vol. 2, 1620–1626 vol.2.

Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., and Schaal, S. (2010). "Fast, robust quadruped locomotion over challenging terrain". In: *2010 IEEE International Conference on Robotics and Automation*, pp. 2665–2670.

Kenneally, G., De, A., and Koditschek, D. E. (2016). "Design Principles for a Family of Direct-Drive Legged Robots". *IEEE Robotics and Automation Letters* 1.2, pp. 900–907.

Khalil, H. K. (2002). *Nonlinear systems; 3rd ed.* Prentice-Hall.

Khatib, O. (1985). "Real-time obstacle avoidance for manipulators and mobile robots". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2, pp. 500–505.

Kim, D., Carballo, D., Di Carlo, J., Katz, B., Bledt, G., Lim, B., and Kim, S. (2020). "Vision Aided Dynamic Exploration of Unstructured Terrain with a Small-Scale Quadruped Robot". In: *International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2464–2470.

Koenemann, J., Prete, A. D., Tassa, Y., Todorov, E., Stasse, O., Bennewitz, M., and Mansard, N. (2015). "Whole-body model-predictive control applied to the HRP-2 humanoid". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3346–3351.

Kolathaya, S. and Veer, S. (2019). "PD based Robust Quadratic Programs for Robotic Systems". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 6834–6841.

Kolter, J. Z., Rodgers, M. P., and Ng, A. Y. (2008). "A control architecture for quadruped locomotion over rough terrain". In: *International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 811–818.

Kolvenbach, H., Bärtschi, C., Wellhausen, L., Grandia, R., and Hutter, M. (2019a). "Haptic Inspection of Planetary Soils With Legged Robots". *IEEE Robotics and Automation Letters* 4.2, pp. 1626–1632.

Kolvenbach, H., Valsecchi, G., Grandia, R., Ruiz, A., Jenelten, F., and Hutter, M. (2019b). "Tactile inspection of concrete deterioration in sewers with legged robots". In: *12th Conference on Field and Service Robotics (FSR 2019)*. ETH Zurich, Robotic Systems Lab.

Kolvenbach, H., Wisth, D., Buchanan, R., Valsecchi, G., Grandia, R., Fallon, M., and Hutter, M. (2020). "Towards autonomous inspection of concrete deterioration in sewers with legged robots". *Journal of Field Robotics* 37.8, pp. 1314–1327.

Kouzoupis, D., Frison, G., Zanelli, A., and Diehl, M. (2018). "Recent Advances in Quadratic Programming Algorithms for Nonlinear Model Predictive Control". *Vietnam Journal of Mathematics* 46.4, pp. 863–882.

Krotkov, E., Hackett, D., Jackel, L., Perschbacher, M., Pippine, J., Strauss, J., Pratt, G., and Orlowski, C. (2017). "The DARPA robotics challenge finals: Results and perspectives". *Journal of Field Robotics* 34.2, pp. 229–240.

Kuindersma, S., Deits, R., Fallon, M., Valenzuela, A., Dai, H., Permenter, F., Koolen, T., Marion, P., and Tedrake, R. (2016). "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot". *Autonomous Robots* 40.3, pp. 429–455.

Layeghi, D., Tonneau, S., and Mistry, M. (2021). "Optimal Control via Combined Inference and Numerical Optimization". *arXiv preprint arXiv:2109.11361*.

Lee, J. and Yu, Z. (1997). "Worst-case formulations of model predictive control for systems with bounded parameters". *Automatica* 33.5, pp. 763 –781.

Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., and Hutter, M. (2020). "Learning quadrupedal locomotion over challenging terrain". *Science Robotics* 5.47, eabc5986.

Lembono, T. S., Mastalli, C., Fernbach, P., Mansard, N., and Calinon, S. (2020). "Learning how to walk: Warm-starting optimal control solver with memory of motion". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1357–1363.

Li, H. and Wensing, P. M. (2020). "Hybrid Systems Differential Dynamic Programming for Whole-Body Motion Planning of Legged Robots". *IEEE Robotics and Automation Letters* 5.4, pp. 5448–5455.

Li, W. C. and Biegler, L. T. (1989). "A Multistep, Newton-Type Control Strategy for Constrained, Nonlinear Processes". In: *1989 ACC*, pp. 1526–1527.

Lin, Y. and Sontag, E. D. (1991). "A universal formula for stabilization with bounded controls". *Sys. & Control Letters* 16.6, pp. 393–397.

Liniger, A., Domahidi, A., and Morari, M. (2015). "Optimization-based autonomous racing of 1:43 scale RC cars". *Optimal Control Applications and Methods* 36.5, pp. 628–647.

Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). "Applications of second-order cone programming". *Linear algebra and its applications* 284.1-3, pp. 193–228.

M. Diehl and H.G. Bock and J. P. Schlöder and R. Findeisen and Z. Nagy and F. Allgöwer (2002). "Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations". *Journal of Process Control* 12.4, pp. 577 –585.

Ma, W., Kolathaya, S., Ambrose, E. R., Hubicki, C. M., and Ames, A. D. (2017). "Bipedal robotic running with DURUS-2D: Bridging the gap between theory and experiment". In: *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*. ACM, pp. 265–274.

Magana, O. A. V., Barasuol, V., Camurri, M., Franceschi, L., Focchi, M., Pontil, M., Caldwell, D. G., and Semini, C. (2019). "Fast and continuous foothold adaptation for dynamic locomotion through cnns". *IEEE Robotics and Automation Letters* 4.2, pp. 2140–2147.

Mahmood, M. and Mhaskar, P. (2014). "Constrained control Lyapunov function based model predictive control design". *International Journal of Robust and Nonlinear Control* 24.2, pp. 374–388.

Manchester, Z. and Kuindersma, S. (2017). "Variational contact-implicit trajectory optimization". In: *Proceedings of the International Symposium on Robotics Research (ISRR), Puerto Varas, Chile*.

Mangasarian, O. and Fromovitz, S (1967). "The Fritz John necessary optimality conditions in the presence of equality and inequality constraints". *Journal of Mathematical Analysis and Applications* 17.1, pp. 37 –47.

Marcucci, T., Deits, R., Gabiccini, M., Bicchi, A., and Tedrake, R. (2017). "Approximate hybrid model predictive control for multi-contact push recovery in complex environments". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 31–38.

Marcucci, T., Gabiccini, M., and Artoni, A. (2017). "A Two-Stage Trajectory Optimization Strategy for Articulated Bodies With Unscheduled Contact Sequences". *IEEE Robotics and Automation Letters* 2.1, pp. 104–111.

Mason, S., Righetti, L., and Schaal, S. (2014). "Full dynamics LQR control of a humanoid robot: An experimental study on balancing and squatting". In: *2014 IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 374–379.

Mason, S., Rotella, N., Schaal, S., and Righetti, L. (2016). "Balancing and walking using full dynamics LQR control with contact constraints". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, pp. 63–68.

Mastalli, C., Havoutis, I., Winkler, A. W., Caldwell, D. G., and Semini, C. (2015). "On-line and on-board planning and perception for quadrupedal locomotion". In: *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pp. 1–7.

Mastalli, C., Budhiraja, R., Merkt, W., Saurel, G., Hammoud, B., Naveau, M., Carpentier, J., Righetti, L., Vijayakumar, S., and Mansard, N. (2020a). "Crocoddyl: An efficient and versatile framework for multi-contact optimal control". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2536–2542.

Mastalli, C., Havoutis, I., Focchi, M., Caldwell, D. G., and Semini, C. (2020b). "Motion Planning for Quadrupedal Locomotion: Coupled Planning, Terrain Mapping, and Whole-Body Control". *IEEE Transactions on Robotics* 36.6, pp. 1635–1648.

Mastalli, C., Merkt, W., Xin, G., Shim, J., Mistry, M., Havoutis, I., and Vijayakumar, S. (2022). "Agile Maneuvers in Legged Robots: a Predictive Control Approach". *arXiv preprint arXiv:2203.07554*.

Mayne, D. (1966). “A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems”. *International Journal of Control* 3.1, pp. 85–95.

Mayne, D. Q. (2014). “Model predictive control: Recent developments and future promise”. *Automatica* 50.12, pp. 2967 –2986.

Mayne, D., Rawlings, J., Rao, C., and Scokaert, P. (2000). “Constrained model predictive control: Stability and optimality”. *Automatica* 36.6, pp. 789 –814.

Melon, O., Geisert, M., Surovik, D., Havoutis, I., and Fallon, M. (2020). “Reliable Trajectories for Dynamic Quadrupeds using Analytical Costs and Learned Initializations”. In: *International Conference on Robotics and Automation (ICRA)*.

Melon, O., Orsolino, R., Surovik, D., Geisert, M., Havoutis, I., and Fallon, M. (2021). “Receding-horizon perceptive trajectory optimization for dynamic legged locomotion with learned initialization”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 9805–9811.

Mhaskar, P., El-Farra, N. H., and Christofides, P. D. (2005). “Predictive control of switched nonlinear systems with scheduled mode transitions”. *IEEE Transactions on Automatic Control* 50.11, pp. 1670–1680.

Mhaskr, P., El-Farra, N. H., and Christofides, P. D. (2006). “Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control”. *Systems & Control Letters* 55.8, pp. 650 –659.

Miki, T., Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., and Hutter, M. (2022a). “Learning robust perceptive locomotion for quadrupedal robots in the wild”. *Science Robotics* 7.62, eabk2822.

Miki, T., Wellhausen, L., Grandia, R., Jenelten, F., Homberger, T., and Hutter, M. (2022b). “Elevation Mapping for Locomotion and Navigation using GPU”. *(submitted to) IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Minniti, M. V., Farshidian, F., Grandia, R., and Hutter, M. (2019). “Whole-Body MPC for a Dynamically Stable Mobile Manipulator”. *IEEE Robotics and Automation Letters* 4.4, pp. 3687–3694.

Minniti, M. V., Grandia, R., Fäh, K., Farshidian, F., and Hutter, M. (2021). “Model predictive robot-environment interaction control for mobile manipulation tasks”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1651–1657.

Minniti, M. V., Grandia, R., Farshidian, F., and Hutter, M. (2022). "Adaptive CLF-MPC With Application to Quadrupedal Robots". *IEEE Robotics and Automation Letters* 7.1, pp. 565–572.

Mistry, M., Buchli, J., and Schaal, S. (2010). "Inverse dynamics control of floating base systems using orthogonal decomposition". In: *2010 IEEE International Conference on Robotics and Automation*, pp. 3406–3412.

Moore, B. (1981). "Principal component analysis in linear systems: Controllability, observability, and model reduction". *IEEE Transactions on Automatic Control* 26.1, pp. 17–32.

Mordatch, I., Todorov, E., and Popović, Z. (2012). "Discovery of complex behaviors through contact-invariant optimization". *ACM Transactions on Graphics (TOG)* 31.4, p. 43.

Morris, B. J., Powell, M. J., and Ames, A. D. (2015). "Continuity and smoothness properties of nonlinear optimization-based feedback controllers". In: *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, pp. 151–158.

Murray, R. M. et al. (2009). "Optimization-based control". *California Institute of Technology, CA*, pp. 111–128.

Nakanishi, J., Mistry, M., and Schaal, S. (2007). "Inverse dynamics control with floating base and constraints". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1942–1947.

Nakanishi, J. and Vijayakumar, S. (2012). "Exploiting Passive Dynamics with Variable Stiffness Actuation in Robot Brachiation". In: *Proceedings of Robotics: Science and Systems*.

Nešić, D. and Grüne, L. (2006). "A receding horizon control approach to sampled-data implementation of continuous-time controllers". *Systems & Control Letters* 55.8, pp. 660–672.

Nešić, D. and Teel, A. R. (2004). "A framework for stabilization of nonlinear sampled-data systems based on their approximate discrete-time models". *IEEE Transactions on Automatic Control* 49.7, pp. 1103–1122.

Nešić, D., Teel, A., and Sontag, E. (1999). "Formulas relating KL stability estimates of discrete-time and sampled-data nonlinear systems". *Systems & Control Letters* 38.1, pp. 49 –60.

Neunert, M., Farshidian, F., Winkler, A. W., and Buchli, J. (2017). "Trajectory Optimization Through Contacts and Automatic Gait Discovery for Quadrupeds". *IEEE Robotics and Automation Letters* 2.3, pp. 1502–1509.

Neunert, M., Stäuble, M., Giftthaler, M., Bellicoso, C. D., Carius, J., Gehring, C., Hutter, M., and Buchli, J. (2018). "Whole-Body Nonlinear Model Predictive Control Through Contacts for Quadrupeds". *IEEE Robotics and Automation Letters* 3.3, pp. 1458–1465.

Nguyen, Q., Da, X., Grizzle, J., and Sreenath, K. (2020). "Dynamic walking on stepping stones with gait library and control barrier functions". In: *Algorithmic Foundations of Robotics XII*. Springer, pp. 384–399.

Nguyen, Q., Hereid, A., Grizzle, J. W., Ames, A. D., and Sreenath, K. (2016). "3D dynamic walking on stepping stones with control barrier functions". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 827–834.

Nguyen, Q. and Sreenath, K. (2015). "Optimal Robust Control for Bipedal Robots through Control Lyapunov Function based Quadratic Programs." In: *Robotics: Science and Systems*.

– (2016a). "Optimal Robust Time-Varying Safety-Critical Control With Application to Dynamic Walking on Moving Stepping Stones". In: *Dynamic Systems and Control Conference (DSCC)*. Vol. 50701. American Society of Mechanical Engineers, V002T28A005.

Nguyen, Q., Agrawal, A., Martin, W., Geyer, H., and Sreenath, K. (2018). "Dynamic bipedal locomotion over stochastic discrete terrain". *The International Journal of Robotics Research* 37.13-14, pp. 1537–1553.

Nguyen, Q. and Sreenath, K. (2016b). "Exponential Control Barrier Functions for enforcing high relative-degree safety-critical constraints". In: *2016 American Control Conference (ACC)*, pp. 322–328.

Nishiwaki, K., Chestnutt, J., and Kagami, S. (2012). "Autonomous navigation of a humanoid robot over unknown rough terrain using a laser range sensor". *The International Journal of Robotics Research* 31.11, pp. 1251–1262.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. second. Springer.

OCS2: An open source library for Optimal Control of Switched Systems. [Online]. Available: `https://github.com/leggedrobotics/ocs2`.

Orin, D. E., Goswami, A., and Lee, S.-H. (2013). "Centroidal dynamics of a humanoid robot". *Autonomous Robots* 35.2-3, pp. 161–176.

Pajarinen, J., Kyrki, V., Koval, M., Srinivasa, S., Peters, J., and Neumann, G. (2017). "Hybrid control trajectory optimization under uncertainty". In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, pp. 5694–5701.

Pankert, J. and Hutter, M. (2020). "Perceptive Model Predictive Control for Continuous Mobile Manipulation". *IEEE Robotics and Automation Letters* 5.4, pp. 6177–6184.

Pardo, D., Neunert, M., Winkler, A., Grandia, R., and Buchli, J. (2017). "Hybrid direct collocation and control in the constraint-consistent subspace for dynamic legged robot locomotion". In: *Proceedings of Robotics: Science and Systems*.

Park, H.-W., Wensing, P., and Kim, S. (2015). "Online Planning for Autonomous Running Jumps Over Obstacles in High-Speed Quadrupeds". In: *Proceedings of Robotics: Science and Systems*.

Patel, A., Shield, S. L., Kazi, S., Johnson, A. M., and Biegler, L. T. (2019). "Contact-Implicit Trajectory Optimization Using Orthogonal Collocation". *IEEE Robotics and Automation Letters* 4.2, pp. 2242–2249.

Perko, L. (2013). *Differential equations and dynamical systems*. Vol. 7. Springer Science & Business Media.

Ponton, B., Herzog, A., Prete, A. D., Schaal, S., and Righetti, L. (2018). "On Time Optimization of Centroidal Momentum Dynamics". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–7.

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, pp. 69–81.

Posa, M., Kuindersma, S., and Tedrake, R. (2016). "Optimization and stabilization of trajectories for constrained dynamical systems". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1366–1373.

Pratt, J., Carff, J., Drakunov, S., and Goswami, A. (2006). "Capture Point: A Step toward Humanoid Push Recovery". In: *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pp. 200–207.

Primbs, J. A., Nevistic, V., and Doyle, J. C. (2000). "A receding horizon generalization of pointwise min-norm controllers". *IEEE Transactions on Automatic Control* 45.5, pp. 898–909.

Qin, S. J. and Badgwell, T. A. (2003). "A survey of industrial model predictive control technology". *Control engineering practice* 11.7, pp. 733–764.

Raibert, M. (1986). *Legged Robots That Balance*. MIT Press.

Rawlings, J. B., Mayne, D. Q., and Diehl, M. (2017). *Model predictive control: theory, computation, and design*. Vol. 2. Nob Hill Publishing Madison, WI.

Reher, J., Kann, C., and Ames, A. D. (2020). "An inverse dynamics approach to control Lyapunov functions". In: *American Control Conference (ACC)*. IEEE, pp. 2444–2451.

Reher, J., Cousineau, E. A., Hereid, A., Hubicki, C. M., and Ames, A. D. (2016). "Realizing dynamic and efficient bipedal locomotion on the humanoid robot DURUS". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1794–1801.

Rezazadeh, S., Hubicki, C., Jones, M., Peekema, A., Van Why, J., Abate, A., and Hurst, J. (2015). "Spring-mass walking with atrias in 3d: Robust gait control spanning zero to 4.3 kph on a heavily underactuated bipedal robot". In: *Dynamic Systems and Control Conference*. Vol. 57243. American Society of Mechanical Engineers, V001T04A003.

Rosolia, U. and Ames, A. D. (2021). "Multi-Rate Control Design Leveraging Control Barrier Functions and Model Predictive Control Policies". *IEEE Control Systems Letters* 5.3, pp. 1007–1012.

Saab, L., Ramos, O. E., Keith, F., Mansard, N., Souères, P., and Fourquet, J.-Y. (2013). "Dynamic Whole-Body Motion Generation Under Rigid Contacts and Other Unilateral Constraints". *IEEE Transactions on Robotics* 29.2, pp. 346–362.

Sastry, S. S. (1999). *Nonlinear Systems: Analysis, Stability and Control*. Springer.

Schlossman, R., Thomas, G. C., Campbell, O., and Sentis, L. (2018). "Exploiting the natural dynamics of series elastic robots by actuator-centered sequential linear programming". In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 1405–1412.

Schnabel, R., Wahl, R., and Klein, R. (2007). "Efficient RANSAC for Point-Cloud Shape Detection". *Computer Graphics Forum* 26.2, pp. 214–226.

Schultz, G. and Mombaur, K. (2010). "Modeling and Optimal Control of Human-Like Running". *IEEE/ASME Transactions on Mechatronics* 15.5, pp. 783–792.

Scokaert, P. O. M. and Mayne, D. Q. (1998). "Min-max feedback model predictive control for constrained linear systems". *IEEE Transactions on Automatic Control* 43.8, pp. 1136–1142.

Scokaert, P. O. M. and Rawlings, J. B. (1999). "Feasibility issues in linear model predictive control". *AIChE Journal* 45.8, pp. 1649–1659.

Sentis, L. and Khatib, O. (2006). "A whole-body control framework for humanoids operating in human environments". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* Pp. 2641–2648.

Seok, S., Wang, A., Chuah, M. Y., Otten, D., Lang, J., and Kim, S. (2013). "Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot". In: *2013 IEEE International Conference on Robotics and Automation*, pp. 3307–3312.

Seyde, T, Carius, J, Grandia, R, Farshidian, F, and Hutter, M (2019). "Locomotion Planning through a Hybrid Bayesian Trajectory Optimization". In: *2019 IEEE International Conference on Robotics and Automation (ICRA)*.

Shield, S., Johnson, A. M., and Patel, A. (2022). "Contact-Implicit Direct Collocation With a Discontinuous Velocity State". *IEEE Robotics and Automation Letters* 7.2, pp. 5779–5786.

Siekmann, J., Green, K., Warila, J., Fern, A., and Hurst, J. (2021). "Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning". In: *Proceedings of Robotics: Science and Systems*.

Singletary, A., Klingebiel, K., Bourne, J., Browning, A., Tokumaru, P., and Ames, A. (2021). "Comparative Analysis of Control Barrier Functions and Artificial Potential Fields for Obstacle Avoidance". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8129–8136.

Sleiman, J.-P., Carius, J., Grandia, R., Wermelinger, M., and Hutter, M. (2019). "Contact-Implicit Trajectory Optimization for Dynamic Object Manipulation". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6814–6821.

Sleiman, J.-P., Farshidian, F., Minniti, M. V., and Hutter, M. (2021). "A Unified MPC Framework for Whole-Body Dynamic Locomotion and Manipulation". *IEEE Robotics and Automation Letters* 6.3, pp. 4688–4695.

Smith, R. et al. (2005). *Open dynamics engine*. `http://www.ode.org`.

Sontag, E. D. (1989a). "A 'universal' construction of Artstein's theorem on nonlinear stabilization". *Systems & control letters* 13.2, pp. 117–123.

– (1989b). "Smooth stabilization implies coprime factorization". *IEEE transactions on automatic control* 34.4, pp. 435–443.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S. (2018). "OSQP: An operator splitting solver for quadratic programs". In: *2018 UKACC 12th International Conference on Control (CONTROL)*. IEEE, pp. 339–339.

Suzuki, S. and be, K. (1985). "Topological structural analysis of digitized binary images by border following". *Computer Vision, Graphics, and Image Processing* 30.1, pp. 32–46.

Sygulla, F., Wittmann, R., Seiwald, P., Hildebrandt, A., Wahrmann, D., and Rixen, D. (2017). "Hybrid position/force control for biped robot stabilization with integrated center of mass dynamics". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 742–748.

Takenaka, T., Matsumoto, T., Yoshiike, T., Hasegawa, T., Shirokura, S., Kaneko, H., and Orita, A. (2009). "Real time motion generation and control for biped robot -4th report: Integrated balance control-". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1601–1608.

Tassa, Y., Erez, T., and Todorov, E. (2012). "Synthesis and stabilization of complex behaviors through online trajectory optimization". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4906–4913.

Tassa, Y., Mansard, N., and Todorov, E. (2014). "Control-limited differential dynamic programming". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1168–1175.

Taylor, A. J., Dorobantu, V. D., Le, H. M., Yue, Y., and Ames, A. D. (2019). "Episodic Learning with Control Lyapunov Functions for Uncertain Robotic Systems". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6878–6884.

Tonneau, S., Prete, A. D., Pettré, J., Park, C., Manocha, D., and Mansard, N. (2018). "An Efficient Acyclic Contact Planner for Multiped Robots". *IEEE Transactions on Robotics* 34.3, pp. 586–601.

Tranzatto, M., Mascarich, F., Bernreiter, L., Godinho, C., Camurri, M., Khattak, S. M. K., Dang, T., Reijgwart, V., Loeje, J., Wisth, D., Zimmermann, S., Nguyen, H., Fehr, M., Solanka, L., Buchanan, R., Bjelonic, M., Khedekar, N., Valceschini, M., Jenelten, F., Dharmadhikari, M., Homberger, T., De Petris, P., Wellhausen, L., Kulkarni, M., Miki, T., Hirsch, S., Montenegro, M., Papachristos, C., Tresoldi, F., Carius, J., Valsecchi, G., Lee, J., Meyer, K., Wu, X., Nieto, J., Smith, A., Hutter, M., Siegwart, R., Mueller, M.,

Fallon, M., and Alexis, K. (2021). "CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge". en. *Field Robotics*.

Unitree Robotics (2022). *Unitree B1 - Industrial grae, super large load, dust-proof & waterproof*. `https://www.unitree.com/products/B1/`. Accessed: 2022-04-03.

Valsecchi, G., Grandia, R., and Hutter, M. (2020). "Quadrupedal Locomotion on Uneven Terrain With Sensorized Feet". *IEEE Robotics and Automation Letters* 5.2, pp. 1548–1555.

Verschueren, R., Duijkeren, N. van, Quirynen, R., and Diehl, M. (2016). "Exploiting convexity in direct optimal control: a sequential convex quadratic programming method". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, pp. 1099–1104.

Villa, N. A., Englsberger, J., and Wieber, P. (2019). "Sensitivity of Legged Balance Control to Uncertainties and Sampling Period". *IEEE Robotics and Automation Letters* 4.4, pp. 3665–3670.

Villarreal, O., Barasuol, V., Wensing, P. M., Caldwell, D. G., and Semini, C. (2020). "MPC-based controller with terrain insight for dynamic legged locomotion". In: *International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2436–2442.

Vukobratović, M. and Borovac, B. (2004). "Zero-moment point — thirty five years of its life". *International Journal of Humanoid Robotics* 1.01, pp. 157–173.

Wächter, A. and Biegler, L. T. (2006). "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". *Mathematical programming* 106.1, pp. 25–57.

Wensing, P. M. and Orin, D. E. (2013). "High-speed humanoid running through control with a 3D-SLIP model". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5134–5140.

Wermelinger, M., Fankhauser, P., Diethelm, R., Krüsi, P., Siegwart, R., and Hutter, M. (2016). "Navigation planning for legged robots in challenging terrain". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1184–1189.

Werner, A., Henze, B., Loeffl, F., Leyendecker, S., and Ott, C. (2017). "Optimal and robust walking using intrinsic properties of a series-elastic robot". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 143–150.

Werner, A., Turlej, W., and Ott, C. (2017). "Generation of locomotion trajectories for series elastic and viscoelastic bipedal robots". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5853–5860.

Westervelt, E. R., Grizzle, J. W., Chevallereau, C., Choi, J. H., and Morris, B. (2007). *Feedback control of dynamic bipedal robot locomotion*. CRC press.

Wieber, P.-B. (2006). "Holonomy and nonholonomy in the dynamics of articulated motion". In: *Fast motions in biomechanics and robotics*. Springer, pp. 411–425.

Winkler, A. W., Bellicoso, C. D., Hutter, M., and Buchli, J. (2018). "Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization". *IEEE Robotics and Automation Letters* 3.3, pp. 1560–1567.

Wu, K., Otoo, E., and Suzuki, K. (2009). "Optimizing Two-Pass Connected-Component Labeling Algorithms". *Pattern Anal. Appl.* 12.2, 117–135.

Wu, Z., Albalawi, F., Zhang, Z., Zhang, J., Durand, H., and Christofides, P. D. (2018). "Control Lyapunov-Barrier Function-Based Model Predictive Control of Nonlinear Systems". In: *2018 Annual American Control Conference (ACC)*, pp. 5920–5926.

Xiong, X. and Ames, A. (2022). "3-D Underactuated Bipedal Walking via H-LIP Based Gait Synthesis and Stepping Stabilization". *IEEE Transactions on Robotics*.

Yu, J., Jadbabaie, A., Primbs, J., and Huang, Y. (2001). "Comparison of nonlinear control design techniques on a model of the Caltech ducted fan". *Automatica* 37.12, pp. 1971 –1978.

Yu, W., Jain, D., Escontrela, A., Iscen, A., Xu, P., Coumans, E., Ha, S., Tan, J., and Zhang, T. (2021). "Visual-Locomotion: Learning to Walk on Complex Terrains with Vision". In: *5th Annual Conference on Robot Learning*.

Zeng, J., Zhang, B., and Sreenath, K. (2021). "Safety-critical model predictive control with discrete-time control barrier function". In: *2021 American Control Conference (ACC)*, pp. 3882–3889.

Zhou, C., Li, Z., Wang, X., Tsagarakis, N., and Caldwell, D. (2016). "Stabilization of bipedal walking based on compliance control". *Autonomous Robots* 40.6, pp. 1041–1057.

Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., and Srinivasa, S. S. (2013). "CHOMP: Covariant Hamiltonian optimization for motion planning". *The International Journal of Robotics Research* 32.9-10, pp. 1164–1193.

Curriculum Vitae

Name	Ruben Jelle Grandia
Born	October 21, 1992 in Zwijndrecht, the Netherlands
Email	rubengrandia@gmail.com

Education

2017–2022 **PhD student at the RSL, ETH Zurich, Switzerland.**
Supervisor: Prof. M. Hutter.
My research interests center around nonlinear optimal control as a general tool to achieve autonomy in modern robotic systems, with a emphasis on quadrupedal walking robots. Throughout several publications, we explored how the key challenges in locomotion can be translated into dynamics, costs, and constraints, while keeping the optimization problem tractable. This research was part of the EU funded Horizon 2020 project THING.

Core algorithms have been contributed to OCS2 - a C++ toolbox tailored for Optimal Control for Switched Systems: github.com/leggedrobotics/ocs2

Nov-Dec 2019 **Visiting Researcher at the AMBER lab, California Institute of Technology.**
During my research visit at Prof. Ames' lab, we collaborated on the unification of tools from nonlinear control theory and nonlinear MPC. By incorporating Control Lyapunov Functions (CLFs) as constraints into a general MPC formulation, closed loop stability guarantees can be provided. We developed practical methods to efficiently solve the resulting nonlinear optimization problems and ultimately realized the proposed controllers on a robotic platform.

2014–2017 **MSc in Robotics, Systems, and Control, ETH Zurich.**
Focus: Optimal Control, Machine learning, Computer Vision.
Graduated with distinction. GPA: 5.9/6

2010–2014 **BSc in Aerospace Engineering, TU Delft.**
Graduated with distinction. GPA: 9.1/10

Work Experience

2011–2021 **Various teaching assistant positions.**

ETH	2017–2021	Robot dynamics.
	2015–2016	Linear System Theory.
TU Delft	2013–2014	Statics, Calculus.
	2012–2013	Intro to Aerospace engineering.
	2011–2012	Programming in Matlab.

2017 **Research assistant at ADRL, ETH Zurich.**
After my master thesis at this lab, I continued to work on the HyQ quadruped robot, publish the results of my thesis, and improved the implementation of a direct collocation method for full body motion optimization.

2015–2017 **Data researcher (part-time) at Genscape Inc.**
Genscape offers a simulation platform for the European energy market. My part-time job at the company involved updating the database with the latest power production information. Furthermore, I worked on analysing historic data to complete missing data, create models of wind power productivity, forecast demand data, etc.

2016 **Internship at Siemens PLM in Leuven, Belgium.**
The goal of this six month internship was to extend an existing sampling based optimal control framework to higher dimensional problems. By introducing a tool for the embedding of engineering knowledge into the problem formulation, the sampling efficiency of the method was improved by several orders of magnitude. The extended method was then applied to the case of a hydraulic hybrid automotive vehicle in simulation.

2011–2014 **Member of Forze Hydrogen Racing team.**

2013–2014	Part-time electronic engineer.
2012–2013	Full-time board member.
2011–2012	Part-time suspension engineer.

Forze is a project in which approximately sixty students collaborate to build hydrogen powered race cars. I was personally responsible for the vehicle dynamics & suspension, and the ten students in that department. More information about the project can be found on www.forze-delft.nl.

Grants and Awards

2020 RA-L Best Paper Award: Valsecchi, G., Grandia, R., and Hutter, M. (2020). "Quadrupedal Locomotion on Uneven Terrain With Sensorized Feet". *IEEE Robotics and Automation Letters* 5.2, pp. 1548–1555.

2019 IEEE RAS TC on Model-Based Optimization for Robotics Best Paper Award: Grandia, R., Farshidian, F., Ranftl, R., and Hutter, M. (2019b). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4730–4737.

2018 ETH silver medal for outstanding Master thesis. The amount of awards per year is restricted to 2.5% of the graduates. *List of awardees available online*.

2014 VSBfonds grant of €10.000. This study grant is awarded to support Dutch students that have shown interest in society and supported their local community. Each year, around 150 grants are awarded for students pursuing a program abroad. www.vsbfonds.nl.

List of Publications

This section provides a list of publications that were published or recently submitted. For a more detailed and up-to-date list, please visit *Google Scholar*.

First Author Publications

- Grandia, R., Jenelten, F., Yang, S., Farshidian, F., and Hutter, M. (2022). "Perceptive Locomotion through Nonlinear Model Predictive Control". *(submitted to) IEEE Transactions on Robotics*

- Grandia, R., Taylor, A. J., Ames, A. D., and Hutter, M. (2021). "Multi-layered safety for legged robots via control barrier functions and model predictive control". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8352–8358

- Grandia, R., Taylor, A. J., Singletary, A., Hutter, M., and Ames, A. D. (2020). "Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions". In: *Proceedings of Robotics: Science and Systems*. Corvalis, Oregon, USA

- Grandia, R., Farshidian, F., Ranftl, R., and Hutter, M. (2019b). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, pp. 4730–4737

- Grandia, R., Farshidian, F., Dosovitskiy, A., Ranftl, R., and Hutter, M. (2019a). "Frequency-aware model predictive control". *IEEE Robotics and Automation Letters* 4.2, pp. 1517–1524

- Grandia, R., Pardo, D., and Buchli, J. (2018). "Contact Invariant Model Learning for Legged Robot Locomotion". *IEEE Robotics and Automation Letters* 3.3, pp. 2291–2298

Co-Authored Publications

- Miki, T., Wellhausen, L., Grandia, R., Jenelten, F., Homberger, T., and Hutter, M. (2022b). "Elevation Mapping for Locomotion and Navigation using GPU". *(submitted to) IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*

- Galliker, M. Y., Csomay-Shanklin, N., Grandia, R., Taylor, A. J., Farshidian, F., Hutter, M., and Ames, A. D. (2022). *Bipedal Locomotion with Nonlinear Model Predictive Control: Online Gait Generation using Whole-Body Dynamics*
- Minniti, M. V., Grandia, R., Farshidian, F., and Hutter, M. (2022). "Adaptive CLF-MPC With Application to Quadrupedal Robots". *IEEE Robotics and Automation Letters* 7.1, pp. 565–572
- Bjelonic, M., Grandia, R., Geilinger, M., Harley, O., Medeiros, V. S., Pajovic, V., Jelavic, E., Coros, S., and Hutter, M. (2022). "Offline motion libraries and online MPC for advanced mobility skills". *The International Journal of Robotics Research*
- Jenelten, F., Grandia, R., Farshidian, F., and Hutter, M. (2021). "TAMOLS: Terrain-Aware Motion Optimization for Legged Systems". *(submitted to) IEEE Transactions on Robotics*
- Minniti, M. V., Grandia, R., Fäh, K., Farshidian, F., and Hutter, M. (2021). "Model predictive robot-environment interaction control for mobile manipulation tasks". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1651–1657
- Bjelonic, M., Grandia, R., Harley, O., Galliard, C., Zimmermann, S., and Hutter, M. (2021). "Whole-Body MPC and Online Gait Sequence Generation for Wheeled-Legged Robots". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8388–8395
- Valsecchi, G., Grandia, R., and Hutter, M. (2020). "Quadrupedal Locomotion on Uneven Terrain With Sensorized Feet". *IEEE Robotics and Automation Letters* 5.2, pp. 1548–1555
- Kolvenbach, H., Wisth, D., Buchanan, R., Valsecchi, G., Grandia, R., Fallon, M., and Hutter, M. (2020). "Towards autonomous inspection of concrete deterioration in sewers with legged robots". *Journal of Field Robotics* 37.8, pp. 1314–1327
- Kolvenbach, H., Bärtschi, C., Wellhausen, L., Grandia, R., and Hutter, M. (2019a). "Haptic Inspection of Planetary Soils With Legged Robots". *IEEE Robotics and Automation Letters* 4.2, pp. 1626–1632
- Kolvenbach, H., Valsecchi, G., Grandia, R., Ruiz, A., Jenelten, F., and Hutter, M. (2019b). "Tactile inspection of concrete deterioration in

sewers with legged robots". In: *12th Conference on Field and Service Robotics (FSR 2019)*. ETH Zurich, Robotic Systems Lab

- Sleiman, J.-P., Carius, J., Grandia, R., Wermelinger, M., and Hutter, M. (2019). "Contact-Implicit Trajectory Optimization for Dynamic Object Manipulation". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6814–6821
- Minniti, M. V., Farshidian, F., Grandia, R., and Hutter, M. (2019). "Whole-Body MPC for a Dynamically Stable Mobile Manipulator". *IEEE Robotics and Automation Letters* 4.4, pp. 3687–3694
- Seyde, T, Carius, J, Grandia, R, Farshidian, F, and Hutter, M (2019). "Locomotion Planning through a Hybrid Bayesian Trajectory Optimization". In: *2019 IEEE International Conference on Robotics and Automation (ICRA)*
- Pardo, D., Neunert, M., Winkler, A., Grandia, R., and Buchli, J. (2017). "Hybrid direct collocation and control in the constraint-consistent subspace for dynamic legged robot locomotion". In: *Proceedings of Robotics: Science and Systems*. Cambridge, Massachusetts

List of Supervised Projects

This section provides a list of student projects that were supervised by the author as part of the doctoral studies.

Master's Theses

Six months full-time research project

- Fenglong, Song (2022): "MPC with Learned Actuator Dynamics"
- Galliker, Manuel Yves (2022): "Bipedal Locomotion with Nonlinear Model Predictive Control"
- Cavaliere, Andrea (2021): "Offset Free MPC for Quadrupeds through Disturbance Observers and MPC"
- Hombach, Paula (2021): "Foothold selection in Optimization based Control"
- Shaohui, Yang (2021): "A Trust-Region Method for Multiple Shooting Optimal Control"
- Pajović, Vuk (2021): "Dynamic Motions for Wheeled-legged Robots using Sampling-Based Initializations of Trajectory Optimizations"
- Harley, Oliver (2020): "Trajectory Optimization with Whole-body MPC for Complex Hybrid Locomotion"
- Fäh, Kevin (2020): "MPC-Based Force Control for a Dynamically Stable Robot"
- Dubath, Benoit (2018): "Design of a learning DC motor controller"
- Sleiman, Jean-Pierre (2018): "Contact-Implicit Trajectory Optimization for Dynamic Object Manipulation"
- Pietrasik, Lukasz (2018): "Hierarchical motion planning through sampling and optimization"
- Bärtschi Christian (2017): "Development of an automated single-leg testbench"
- Seyde, Tim (2017): "Learning to Optimize Single-Legged Hopping"

Semester Projects

Four months part-time research project

- Krinner, Maria (2022): "Model Hierarchy Predictive Control"
- Grzegorz, Malczyk (2020): "Perceptive MPC for Wheeled-Legged Robots"
- Ni, Chao (2020): "Trajectory Optimization and MPC for Wheeled-legged Robots"
- Adzic, Andrej (2020): "Convex Terrain Representation"
- Shaohui, Yang (2020): "Multiple Shooting Method in Model Predictive Control Deployed on Legged Robots"
- De Vincenti, Flavio (2019): "Model Predictive Control Approach for Whole Body Motion Planning of ANYmal on Rough Terrain"
- Mouritzen, Daniel (2018): "Integrating Navigation and MPC for ANYmal"

Bachelor's Theses

Four months part-time research project

- Kaufmann, Timon (2021): "Animation-based trajectory creation for quadruped robot Dyana"
- Bührer, Nick (2019): "Perceptionless locomotion for the VariLeg Exoskeleton"
- Sun, Benjamin; Zenkl, Radek (2018): "Object detection and manipulation on a mobile robot"

Focus Project

Ten months product development project

- Grimm, Andrina; Zoller, Cyrill; Gregori, Delia; Portenier, Dominique; Bähler, Jannis; Iten, Klemens; Trentini, Marco; Jans, Meret; Breuer, Peter; Müller, Raffael; Heinrich, Sophia; Wenger, Thibaut; Kaufmann, Timon (2020 - 2021): "Dyana – Dynamic Animatronic"