THE UNIVERSITY
of EDINBURGH

Optimization-based Multi-contact Motion Planning for Legged Robots

Iordanis Chatzinikolaidis

Doctor of Philosophy

Institute of Perception, Action and Behaviour
CDT Robotics and Autonomous Systems
School of Informatics
The University of Edinburgh

2021

Abstract

For legged robots, generating dynamic and versatile motions is essential for interacting with complex and ever-changing environments. So far, robots that routinely operate reliably over rough terrains remains an elusive goal. Yet the primary promise of legged locomotion is to replace humans and animals in performing tedious and menial tasks, without requiring changes in the environment as wheeled robots do.

A necessary step towards this goal is to endow robots with capabilities to reason about contacts but this vital skill is currently missing. An important justification for this is that contact phenomena are inherently non-smooth and non-convex. As a result, posing and solving problems involving contacts is non-trivial. Optimization-based motion planning constitutes a powerful paradigm to this end. Consequently, this thesis considers the problem of generating motions in contact-rich situations.

Specifically, we introduce several methods that compute dynamic and versatile motion plans from a holistic optimization perspective based on trajectory optimization techniques. The advantage is that the user needs to provide a high-level task description in the form of an objective function only. Subsequently, the methods output a detailed motion plan—that includes contact locations, timings, gait patterns—that optimally achieves the high-level task.

Initially, we assume that such a motion plan is available, and we investigate the relevant control problem. The problem is to track a nominal motion plan as close as possible given external disturbances by computing inputs for the robot. Thus, this stage typically follows the motion planning stage. Additionally, this thesis presents methods that do not necessarily require a separate control stage by computing the controller structure automatically.

Afterwards, we proceed to the main parts of this thesis. First, assuming a pre-specified contact sequence, we formulate a trajectory optimization method reminiscent of hybrid approaches. Its backbone is a high-accuracy integrator, enabling reliable long-term motion planning while satisfying both translational and rotational dynamics. We utilize it to compute motion plans for a hopper traversing rough terrains—with gaps and obstacles—and performing explosive motions, like a somersault. Subsequently, we provide a discussion on how to extend the method when the contact sequence is unspecified.

In the next chapter, we increase the complexity of the problem in many aspects. First, we formulate the problem in joint-level utilizing full dynamics and kinematics models. Second, we assume a contact-implicit perspective, *i.e.* decisions about contacts are implicitly defined in the problem's formulation rather than defined as explicit contact modes. As a result, pre-specification of the contact interactions is not required, like the order by which the feet contact the ground for a quadruped robot model and the respective timings. Finally, we extend the classical rigid contact model to surfaces with soft and slippery properties. We quantitatively evaluate our proposed framework by performing comparisons against the rigid model and an alternative contact-implicit framework. Furthermore, we compute motion plans for a high-dimensional quadruped robot in a variety of terrains exhibiting the enhanced properties.

In the final study, we extend the classical Differential Dynamic Programming algorithm to handle systems defined by implicit dynamics. While this can be of interest in its own right, our particular application is computing motion plans in contact-rich settings. Compared to the method presented in the previous chapter, this formulation enables experiencing contacts with all body parts in a receding horizon fashion, albeit with limited contact discovery capabilities. We demonstrate the properties of our proposed extension by comparing implicit and explicit models and generating motion plans for a single-legged robot with multiple contacts both for trajectory optimization and receding horizon settings.

We conclude this thesis by providing insights and limitations of the proposed methods, and possible future directions that can improve and extend aspects of the presented work.

Lay summary

Computing dynamic motions that are physically feasible for legged robots is a challenging problem. First, legged robots operate in a way that can easily lead to faults. For example, slipping during walking or placing a foot in a wrong manner can lead to undesirable falls. Second, from a computational perspective, a large number of quantities need to be computed very quickly, which is in practice very computationally demanding. These quantities can be both continuous, such as the motion of each joint of the robot, but also discrete, like selecting where to step each foot. The final important aspect is the need to take into account changing environments. Most commercial robots operate on factory settings, where everything around the robot is static or moves with precisely prescribed motions. Yet we expect legged robots to perform tasks in a wide range of outdoor, cluttered and changing environments.

Striking a balance between the need to compute precise motions and doing it very fast has proven quite challenging, given all these requirements above. Research in legged locomotion focused initially on satisfying the strict real-time constraint by imposing strong assumptions and simplifying the problem. Advances in computational power enabled us to increase the modelling complexity accordingly, and lifting parts of those assumptions. Yet we have reached a point where modelling efforts are fragmented: Motion planning is based on the connection of computational blocks, each trying to perform a specific behaviour, typically in a waterfall manner. As our requirements for legged robots that operate fluently in a variety of environments become more strict, computing motions that are general and adaptable becomes more important. The focus of this thesis is to introduce several novel computational and algorithmic methods that can compute general motions with minimal user input and assumptions.

To this end, the main theoretical contributions of this thesis are as follows. First, to formulate the motion planning problem in a unified framework. Instead of breaking the problem into multiple parts, our aim is a single, unified formulations that can be easily be configured to work in a large variety of situations. Second, we are interested in increasing the fidelity of the computed motions by performing a richer and more detailed modelling of the environment. We try to model not only hard interactions with rigid surfaces, but also slippery and soft ones, which are quite common in real environments.

Lastly, the main practical contribution is to introduce three approaches in Chapters 5 to 7, that are based on different assumptions and exhibit general motion planning capabilities. All of them are formulated based on a numerical optimization viewpoint, where the aim is to compute optimal motions subject to a number of constraints. The only requirement from the user is to quantify what they think is an optimal motion by means of a cost function. For example, this practically translates to reaching a particular goal point as close as possible, or as fast as possible, or without requiring much energy consumption. Afterwards, the presented approaches try to compute detailed motion plans that achieve these goals, while taking into account physical constraints specifically for legged robots.

The first approach focuses more on the interaction of the robot with the environment rather than with the robot itself. As a result, it uses a simple model for the robot. This allows taking into account challenging environments, for example when obstacles are involved or when a gap might require a jumping motion. The second approach includes a detailed modelling of the robot and how this can interact with the environment. It introduces simplifications on the environment, and computes more general motions than the previous approach. Further, it is able to model additional interactions with the environment, such as slippery and soft contact interactions. The last approach tries to strike a balance between the previous ones by including both detailed models and operating on general environments.

Apart from the previous contributions, this thesis also includes a through discussion about previous work in Chapter 2, with detailed discussion about theoretical prerequisites in Chapter 3. Chapter 4 includes a discussion on the relevant control problem: given a motion plan output from the previous methods, how can this be reliably executed on a real robot? This is a very important consideration when it comes to practical engineering work, but is outside of the scope of this thesis, so we only discuss briefly. Finally, our work serves as a stepping stone towards more general and adaptive motion planning for legged robots. In Chapter 8 we discuss key limitations and promising future directions.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Iordanis Chatzinikolaidis*)

Acknowledgements

First and foremost, I would like to express my gratitude to my advisor, Dr Zhibin (Alex) Li, for introducing me to the world of legged locomotion, unbeknown to me prior to my studies, and for his overall support—both research related as well as regarding my professional development. I would also like to thank my second advisor, Prof Sethu Vijayakumar, for providing me with the opportunity to do my studies in Edinburgh and for the feedback during my reviews.

Doctoral studies can be quite burdensome at times and it is important to have proper support and a sense of understanding. For fulfilling this need, I would like to thank all my colleagues at the Institute of Perception, Action and Behaviour and the Edinburgh Centre for Robotics. I would especially like to thank Eleftherios, Kai, Chris, Wouter, Quentin, Wenbin, Wanming, Qingbiao, and Chuanyu from my group, the Advanced Intelligent Robotics, and Theodoros, Jiayi, João, Chris, and Henrique from the Statistical Machine Learning and Motor Control group. I would also like to express my gratitude to Shihao Wang for being the best conference buddy.

At this point, I would also like to thank Dr Yangwei You for hosting me during my internship at the Agency for Science, Technology and Research. My six-month stay there was particularly enjoyable and I really appreciate the opportunity to experience firsthand the beautiful Singapore.

My studies in Edinburgh were particularly delightful because of the experiences and fun that I had outside of my research. For this fantastic period of my life that I will forever remember and cherish, I would like to thank all members and non-members of the "Cinderella" group (in alphabetical order): Aisha, Alex, Antreas, Argyris, Chef, Giannis, Giota, John, Maria, Marina, Michalis, Niki, Nikos, Pigi, Takis, Velanis.

My childhood and undergrad friends will always have a special place in my life and I am always looking forward to catch up with them every time I visit home. For supporting me and being with me all these years I would like to thank (again in alphabetical order): Alexandros, Anna, Antonis, Aris, George, Ilias, Janis, Kostas, Leandros, Maria, Nikos, Petros, Silent, Sotiris, Thodoris, Thomas.

Last but not least, I want to thank the most important people in my life. Their

support and love is continuous, unwavering, and the cornerstone of my life: my parents since I was born, my sister since she was born, Eirini since I first met her.

Funding

This research was supported by the Engineering and Physical Sciences Research Council as part of the Centre for Doctoral Training in Robotics and Autonomous Systems at Heriot-Watt University and The University of Edinburgh. Grant ref.: (EP/L016834/1).

Για τους αγαπημένους μου

Table of Contents

List of Acronyms xiii

List of Figures xv

List of Publications xvii

List of Symbols xix

List of Tables xxii

1 Introduction 1

1.1 Problem statement . . . 2
1.2 Outline . . . 5
1.3 Contributions . . . 7

2 Prior work 9

2.1 Motion planning by template models . . . 9
2.1.1 Zero-moment point . . . 10
2.1.2 Linear inverted pendulum model . . . 11
2.1.3 Capture point . . . 12
2.1.4 Spring-loaded inverted pendulum . . . 12
2.1.5 Summary . . . 13
2.2 Motion planning by tasks synthesis . . . 14
2.2.1 Optimization-based methods . . . 16
2.2.2 Probabilistic methods . . . 17
2.2.3 Summary . . . 18
2.3 Motion planning by holistic optimization . . . 19
2.3.1 Hybrid optimization . . . 19

2.3.2 Contact-implicit optimization . . . 20
2.3.3 Shooting methods with contacts . . . 22
2.3.4 Summary . . . 23
2.4 Machine learning . . . 25
2.4.1 Deep reinforcement learning . . . 25

3 Background **27**
3.1 Optimal control . . . 28
3.2 Mathematical programming . . . 29
3.2.1 Karush-Kuhn-Tucker conditions . . . 30
3.2.2 Sequential quadratic programming methods . . . 32
3.2.3 Interior-point methods . . . 32
3.3 Trajectory optimization . . . 33
3.3.1 Transcription . . . 33
3.3.2 Shooting, multiple shooting, and collocation . . . 34
3.3.3 Differential dynamic programming . . . 35
3.4 Rigid-body dynamics with contacts . . . 39
3.4.1 Time-stepping formulation . . . 40
3.4.2 Contact models . . . 42

4 Whole-body control **48**
4.1 Prior work . . . 49
4.1.1 Velocity-based whole-body control . . . 49
4.1.2 Torque-based whole-body control . . . 52
4.2 Optimization-based formulation . . . 53
4.2.1 Whole-body QP-based formulations background . . . 54
4.2.2 Whole-body QP-based approach . . . 55
4.3 Results . . . 56
4.3.1 Hand motion with gravity compensation for Atlas . . . 56
4.3.2 Automatic gain tuning for Valkyrie . . . 57
4.4 Limitations . . . 58

5 Contact-implicit trajectory optimization in task space **60**
5.1 Specified contact sequence . . . 61
5.1.1 Formulation of the problem . . . 64
5.1.2 Results . . . 70

5.1.3 Conclusion . . . 75
5.2 Unspecified contact sequence . . . 76
5.2.1 Kinematics constraints . . . 77
5.2.2 Dynamics constraints . . . 78
5.2.3 Results . . . 79

6 Contact-implicit trajectory optimization in joint space 82
6.1 Introduction . . . 83
6.1.1 Contributions . . . 85
6.2 Trajectory optimization formulation . . . 86
6.2.1 Optimal control problem . . . 86
6.2.2 Contact model with analytical solution . . . 87
6.2.3 Direct transcription . . . 89
6.3 Results . . . 90
6.3.1 Comparison with physics simulation . . . 92
6.3.2 Comparison with a MPCC formulation . . . 92
6.3.3 ANYmal trotting on hard and slippery surfaces . . . 96
6.3.4 ANYmal jumping on hard and soft surfaces . . . 98
6.4 Conclusion . . . 100

7 Differential dynamic programming with contacts 101
7.1 Introduction . . . 102
7.1.1 Contributions . . . 104
7.2 Prior work . . . 105
7.2.1 Differential dynamic programming . . . 105
7.2.2 Through-contact motion planning . . . 106
7.3 Implicit differential dynamic programming . . . 107
7.3.1 First-order sensitivity analysis . . . 107
7.3.2 Second-order sensitivity analysis . . . 108
7.3.3 Gauss-Newton approximation . . . 109
7.4 Acceleration-level contact dynamics . . . 109
7.5 Results . . . 112
7.5.1 Implementation details . . . 112
7.5.2 Aggregate double pendulum swing-up . . . 112
7.5.3 Single double pendulum swing-up . . . 114
7.5.4 Multi-contact stand-up . . . 116

7.5.5 Multi-contact balancing 117
7.6 Conclusion 119

8 Conclusion **121**
8.1 Summary 121
8.2 Limitations 123
8.3 Future directions 125
8.3.1 Informed initialization 125
8.3.2 Objective function definition 127
8.3.3 Sparsity in the time horizon 127
8.3.4 Higher-order methods 127

Bibliography **128**

List of Acronyms

BLCP bounded linear complementarity problem 42, 43

CICM convex and invertible contact model 42, 45, 46

CoM centre of mass xix, 10–12, 15–18, 53, 55, 58, 65, 66, 71–73, 75

CoP centre of pressure 16, 54

CP capture point 12, 61

DDP differential dynamic programming 5, 6, 8, 22, 24, 35, 37, 38, 41, 101, 103–107, 109–117, 119, 120, 122, 124, 125, 127

DoF degrees of freedom xxi, 19, 34, 35, 52, 86, 104, 116

iLQR iterative linear-quadratic regulator 105, 109, 113–116

IP interior-point 32, 70, 90, 92, 124

KKT Karush-Kuhn-Tucker 30, 31, 84, 87

LCP linear complementarity problem 43

LIP linear inverted pendulum 11–13, 61

LP linear programming 16, 30, 31, 43, 44

LQR linear-quadratic regulator 6, 54

MDP maximum dissipation principle 43, 45, 87

MP Moore–Penrose generalized inverse 50

MPC model predictive control 16, 117, 123

MPCC mathematical problem with complementarity constraints 21, 84, 85, 93–95

MRP modified Rodrigues parameters 90, 91

NCP nonlinear complementarity problem 42, 43, 87

OCP optimal control problem 86

ODE ordinary differential equation 11

PD proportional-derivative control 56, 57

PGS projected Gauss-Seidel 44, 92, 110

QCQP quadratically constrained quadratic programming 30, 31, 45, 106

QP quadratic programming 6, 30–32, 43, 45, 51, 53–55, 61, 105, 127

RMSE root-mean-square error 72–74, 93

RNE recursive Newton–Euler 54, 101

RRT rapidly-exploring random trees 17

SDP semidefinite programming 30, 31

SLIP spring-loaded inverted pendulum 12, 13

SOCP second-order cone programming 30, 31

SQP sequential quadratic programming 32

TO trajectory optimization 4, 5, 7, 8, 27, 33–35, 41, 42, 45, 46, 60, 78, 82, 84–87, 92, 93, 100, 102, 103, 116, 124, 126, 127

TPBVP two-point boundary-value problem 28, 29

WBC whole-body controller 5, 6, 45, 48, 49, 51–54, 56–58, 123, 127

ZMP zero-moment point 10, 11, 80

List of Figures

1.1 Examples of robots targeted by the topics of this thesis 3
1.2 Overview of the approaches presented in this thesis 6

2.1 Walking approximated by an inverted pendulum 11
2.2 Inverted pendulum vs. SLIP template models 13

3.1 Hierarchy of convex programming classes 31
3.2 Dynamic quantities for a quadruped robot model 40
3.3 Common linearizations of the Coulomb friction cone 44

4.1 Time-lapse snapshots of Atlas' arm motion 56
4.2 Snapshots of Valkyrie navigating different terrains 57

5.1 Dynamic maneuvers using a unified optimization framework . . . 62
5.2 Snapshots of jumping over an obstacle 71
5.3 Centre of mass position 72
5.4 Snapshots of jumping over a gap 73
5.5 Complementarity between foot position and contact force 73
5.6 Snapshots of jumping over a gap with inverse kinematics 74
5.7 Snapshots of a dynamic somersault motion 75
5.8 Orientation difference with respect to baseline 76
5.9 Smooth approximations of absolute value function 79
5.10 Snapshots of a walking motion for a bipedal model 80
5.11 Snapshots of a galloping motion for a quadruped model 80

6.1 Dynamic motions on variable grounds 83
6.2 Influence of r_t on the rigid ball dropping test 93
6.3 Comparison of normal impulses between benchmarked methods . 95
6.4 Snapshots of trotting on hard ground 96

6.5 Joint positions and torques for trotting on hard ground 97
6.6 Comparison of body positions during trotting and jumping tasks . 98
6.7 Foot height during jumping on hard and soft ground 98
6.8 Snapshots of jumping motions on hard ground 99
6.9 Snapshots of jumping motions on soft ground 99

7.1 Complex multi-contact motions of a single-leg robot model 103
7.2 Results for the cost of four DDP variants 113
7.3 Results for the number of iterations of four DDP variants 113
7.4 Cost per iteration for six implicit DDP variants 114
7.5 Time-lapse snapshots of contact-rich motions 117
7.6 Base and joint positions during the standing up task 118
7.7 Cost at start & end of each run for the MPC formulation 119

List of Publications

The research work presented in this thesis led to the following publications:

I. Chatzinikolaidis and Z. Li (2021). "Trajectory optimization of contact-rich motions using implicit differential dynamic programming". In: *IEEE Robotics and Automation Letters*. Vol. 6. 2, pp. 2626–2633. DOI: 10.1109/LRA.2021.3061341.

I. Chatzinikolaidis, Y. You, and Z. Li (2020). "Contact-implicit trajectory optimization using an analytically solvable contact model for locomotion on variable ground". In: *IEEE Robotics and Automation Letters* 5.4, pp. 6357–6364. DOI: 10.1109/LRA.2020.3010754.

I. Chatzinikolaidis, T. Stouraitis, S. Vijayakumar, and Z. Li (2018). "Nonlinear optimization using discrete variational mechanics for dynamic maneuvers of a 3D one-leg hopper". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 932–937. DOI: 10.1109/HUMANOIDS.2018.8624981.

Parts and ideas of this work contributed to the following publications:

T. Stouraitis, **I. Chatzinikolaidis**, M. Gienger, and S. Vijayakumar (2020). "Online hybrid motion planning for dyadic collaborative manipulation via bilevel optimization". In: *IEEE Transactions on Robotics* 36.5, pp. 1452–1471. DOI: 10.1109/TRO.2020.2992987. 2020 IEEE RAS TC on Model-Based Optimization for Robotics **Best Paper Award Finalist**.

J. Wang, **I. Chatzinikolaidis**, C. Mastalli, W. Wolfslag, G. Xin, S. Tonneau, and S. Vijayakumar (2020). "Automatic gait pattern selection for legged robots". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 3990–3997. DOI: 10.1109/IROS45743.2020.9340789.

K. Yuan, **I. Chatzinikolaidis**, and Z. Li (2019). "Bayesian optimization for whole-body control of high-degree-of-freedom robots through reduction of dimensionality". In: *IEEE Robotics and Automation Letters* 4.3, pp. 2268–2275. DOI: 10.1109/LRA.2019.2901308.

W. Hu, **I. Chatzinikolaidis**, K. Yuan, and Z. Li (2018). "Comparison study of nonlinear optimization of step durations and foot placement for dynamic walking". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 433–439. DOI: 10.1109/ICRA.2018.8461101.

T. Stouraitis, **I. Chatzinikolaidis**, M. Gienger, and S. Vijayakumar (2018). "Dyadic collaborative manipulation through hybrid trajectory optimization". In: *Conference on Robot Learning (CoRL)*. Vol. 87, pp. 869–878. URL: http://proceedings.mlr.press/v87/stouraitis18a.html. CoRL 2018 **Best Systems Paper Award Finalist**.

Q. Li, **I. Chatzinikolaidis**, Y. Yang, S. Vijayakumar, and Z. Li (2017). "Robust foot placement control for dynamic walking using online parameter estimation". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 165–170. DOI: 10.1109/HUMANOIDS.2017.8239552.

List of Symbols

A	inverse inertia matrix in joint space
α^*	constraint stabilizing accelerations
α^+	post-contact acceleration
α^-	pre-contact acceleration
α	orientation in quaternion
β	relative orientation between two frames
$_L$	lower bound of quantity
$_U$	upper bound of quantity
c	CoM position
dt	time step size
ϵ	small positive number
F	wrench
f	state equation
f	contact force
F	generalized force
g	inequality or path constraints vector
γ	non-negative gains
H	centrifugal, Coriolis, and gravitational forces vector

h	equality constraints vector
I	inertia matrix
i	time index
I	identity matrix
J	concatenated contact Jabobians
j	general index
$\tilde{\mathrm{J}}$	differential transformation
J	Jacobian matrix
K	feedback gains matrix
k	contact point index
κ	feedforward gains vector
L	cost-to-go function
l	cost function term
$\mathcal{L}$	Lagrangian
$\mathcal{L}_d$	discrete Lagrangian
λ	contact impulses
M	mass matrix
m	mass of rigid body
μ	friction coefficient
N	length of time horizon
$\mathcal{N}$	null space matrix
ω	angular velocity
p	end-effector position
ϕ	signed distance function
Q	Q-function
q	generalized configuration

R	diagonal positive definite matrix
r	contact model parameter
S	actuated DoF selection matrix
s	sensitivity vector or slack variables vector
t	time variable or time index
τ	torque input
u	input vector
V	value function
v	generalized velocity
v^*	constraint stabilizing velocity
v^+	post-contact velocity
v^-	pre-contact velocity
$\dot{v}$	generalized acceleration
$\mathcal{V}$	potential energy
v	task velocity
w	weighting term
x	state vector or optimization variables vector

List of Tables

2.1 Comparison between motion planning methods 23

3.1 Size of quantities for the Q-function's quadratic approximation in DDP 37

5.1 Parameters used in single rigid body's dynamic motion planning scenarios 70

6.1 Parameters for the unactuated rigid body models 91

6.2 Running time and iterations of the MPCC versus our proposed formulation 94

7.1 Effect of time step on number of iterations until convergence . . . 115

Chapter 1

Introduction

Robotics has experienced swift growth in the last couple of years. A number of sectors are upended due to the introduction of robots such as health care, agriculture, food processing, and manufacturing. Healthcare robots are expected to cater for the elderly and people with disabilities. In agriculture, large tractors will be able to work the fields autonomously, while small purpose-built robots will take over menial tasks like pruning, mowing, spraying, and harvest collection. Kitchen robots are expected to prepare healthy meals. In manufacturing, robots are already increasing productivity and efficiency while slashing costs; a trend that is expected to continue and even expand with the introduction of cobots.

Legged robots—more relevant to the content of this thesis—have started to find applications outside of research. This is evident by the spawn of a number of companies that make legged robots available for commercial solutions. Areas where legged robots show potential are disaster response, inspection, maintenance, and entertainment. In disaster response, the ability to change contact configuration and navigate discrete terrain is paramount due to the cluttered environment; a property that cannot be replicated by wheeled robots which require continuous navigation paths. In inspection, legged robots can be useful in cases where the physical presence of humans is either dangerous or costly such as offshore platforms. Maintenance from factories to space structures can benefit from the adoption of legged robots. Humanoid robots find applications in entertainment, primarily due to their physical resemblance with humans, shouldering difficult and dangerous tasks such as stunts.

A common theme in all previous cases is the need to make and break contacts. Yet dynamic interactions in multi-contact settings are still challenging. The majority of the commercially available robots are limited to repeated tasks in tightly controlled environments. Contacts with the environment are undesirable, if not outright prohibited. Thus, dynamic adaptations are extremely limited. Unless the behaviour is painstakingly planned in advance, interaction capabilities are completely missing, even in cases where contacts could be beneficial.

For legged robots, contacts pose even more fundamental complications. This is because contacts with the limbs are vital for successful navigation, while contacts with the rest of the robot's structure are considered dangerous; motion planning frameworks actively try to avoid them, except for limited, carefully pre-planned cases like tool manipulation. Nevertheless, restricting motion plans to allow contacts with the limbs only is non-trivial too. That is because both *discrete and continuous quantities* need to be harmoniously combined in real-time: among them foothold and limbs' order selections, contact timings, joint and body trajectories. Currently, in all but a handful of research results, most of these quantities are specified by human operators by trial and error approaches.

Given a definition of robotics as "the scientific and engineering discipline concerned with the creation, composition, structure, evaluation and properties of embodied artificial capabilities" (Redfield, 2019), coupled with multiple results from Embodied Intelligence (Pfeifer and Bongard, 2006; Cangelosi et al., 2015), coordinating interactions between a robot and the environment can be a fundamental skill directly linked with a range of issues: from practical, such as improving safety and enhancing autonomy, to more significant, like the emergence of intelligence. And contacts mediate the interactions between the robot and the environment. Algorithms that are capable to exploit and skilfully manipulate these interactions can lead to far-reaching outcomes.

1.1 Problem statement

From the preceding, approaches that are able to leverage contacts can have impactful and meaningful consequences. As a result, this thesis focuses on methods that can tackle aspects of *contact planning for legged robots from a holistic perspective*. Specifically, we propose optimization formulations that are able to handle general

(a) ANYmal (Hutter et al., 2016)

(b) Valkyrie (Radford et al., 2015)

(c) Talos (Stasse et al., 2017)

(d) Centauro (Kashiri et al., 2019)

Figure 1.1: Examples of robots that the motion planning approaches presented in this thesis aim to accommodate.

contact phenomena with minimum user input. Our vision is to effortlessly generate dynamic motions for legged robots that can successfully traverse complex terrains, matching or even surpassing animal capabilities. Furthermore, we underline the holistic optimization perspective, since the aim is to formulate the solution in a unified framework; a single approach that can work for arbitrary legged configurations and environments.

As this thesis is far from a complete solution to the problem, we make deliberate choices regarding the aspects of the problem that we emphasize. We prioritize the *holistic formulation* goal, which unfortunately impacts the computation rates of our methods. Furthermore, we prioritize methods that can *work for arbitrary*

legged systems, while implementing them on simpler environments than the ones we ultimately envision. Since research has not converged yet to an appropriate set of solution strategies, we tackle it through different angles, *i.e.* we propose *different solution methodologies of increasing complexity*. Yet all our methodologies respect the deliberate choices, which translates to formulations that pose the problem as a single optimization and can be applied in principle to multi-contact settings for arbitrary robot configurations, such as the ones shown in Figure 1.1.

Another important point that this research tries to address is the need to include a *wide range of environmental interactions during motion planning*. The majority of planning approaches only model rigid interactions with the environment, excluding contacts that are deformable or non-stationary. This is a limitation not only for TO methods, but also for more traditional motion planning approaches, *e.g.* probabilistic methods. In reality, most of the contacts are either non-rigid or only approximately rigid. For example, almost all legged robots have rubber feet to absorb impact forces. More importantly, many typical environments can be soft too, like soil, or slippery when for example water is spilled on the floor.

Motion planning with rigid contact interactions relies on the robustness inherent on feedback control to address this limitation. This can be relatively dangerous, as it is difficult to quantify the effect of this assumption on the overall motion execution. A main reason behind the existence of this assumption is the computational advantages. Given rigid contacts, motion planning frameworks operate on a much smaller regime of the possible contact modes. This is especially important for TO, where computational challenges can render practical implementations infeasible.

Our formulations here aim to *extend motion capabilities beyond rigid contact interactions* at the planning level. Some of the proposed methods are able to output motions that exhibit non-rigid characteristics. This requires more detailed problem formulations, introducing appropriate contact models as part of the TO formulation, with a subsequent increase in the overall modelling complexity. Yet this enables us to compute motion plans with arbitrary contacts with the environment that can include a larger set of possible modes. As a result, we are able to increase the motion planning capabilities of robots that make contacts with the environment and equip them with a larger repertoire of motions in the face of uncertain environmental interactions.

1.2 Outline

We start by classifying and discussing previous works in Chapter 2. In Chapter 3 we provide necessary background information: optimal control, mathematical programming, TO techniques, and rigid-body dynamics in the presence of impacts and friction. If a complete implementation on a robot can be thought of as a puzzle, with our work here being just a piece to it, we discuss in Chapter 4 another important puzzle piece: how to successfully track the computed motion plans. Chapter 5 introduces our analysis for contact-implicit motion planning in task space, both with specified and unspecified gait sequence. *Contact-implicit* motion planning refers to methods that do not require contact information as input and are capable to automatically compute them. Chapter 6 presents our results on joint-space motion planning, coupled with the contact modelling for soft and slippery terrain navigation. In Chapter 7 we analyze our extension to DDP for implicitly defined systems, along with applications on contact-rich motion planning. Chapter 8 concludes with a discussion on the advantages and limitations of our methods, along with some possible future research avenues.

At this point, it is worth providing a high-level description of a possible pipeline that can realize the execution of our plans on a robotic platform, which is summarized in Figure 1.2. In the beginning, the user specifies the *desired task in the form of an objective function*, as it is commonly done in the optimal control and optimization literature. This formulation accommodates more than one desired tasks by expressing them as a weighted sum of different terms. The user is responsible for providing the weights that reflect two important aspects: The relative importance between the different tasks, and homogeneity for the different physical quantities that are involved.

Next, a *choice regarding the exact motion planning framework* is made. The first option is the task-space planner presented in Chapter 5. This will output task-level quantities, *e.g.* end-effectors and body motion, and their respective timings, that will be fed to the WBC as nominal inputs; the WBC essentially translates these task-space quantities to equivalent joint-space quantities that the robot requires as input.

The second option is to use the joint-space planner described in Chapter 6. There is a noticeable increase in complexity with respect to the previous option. Yet the

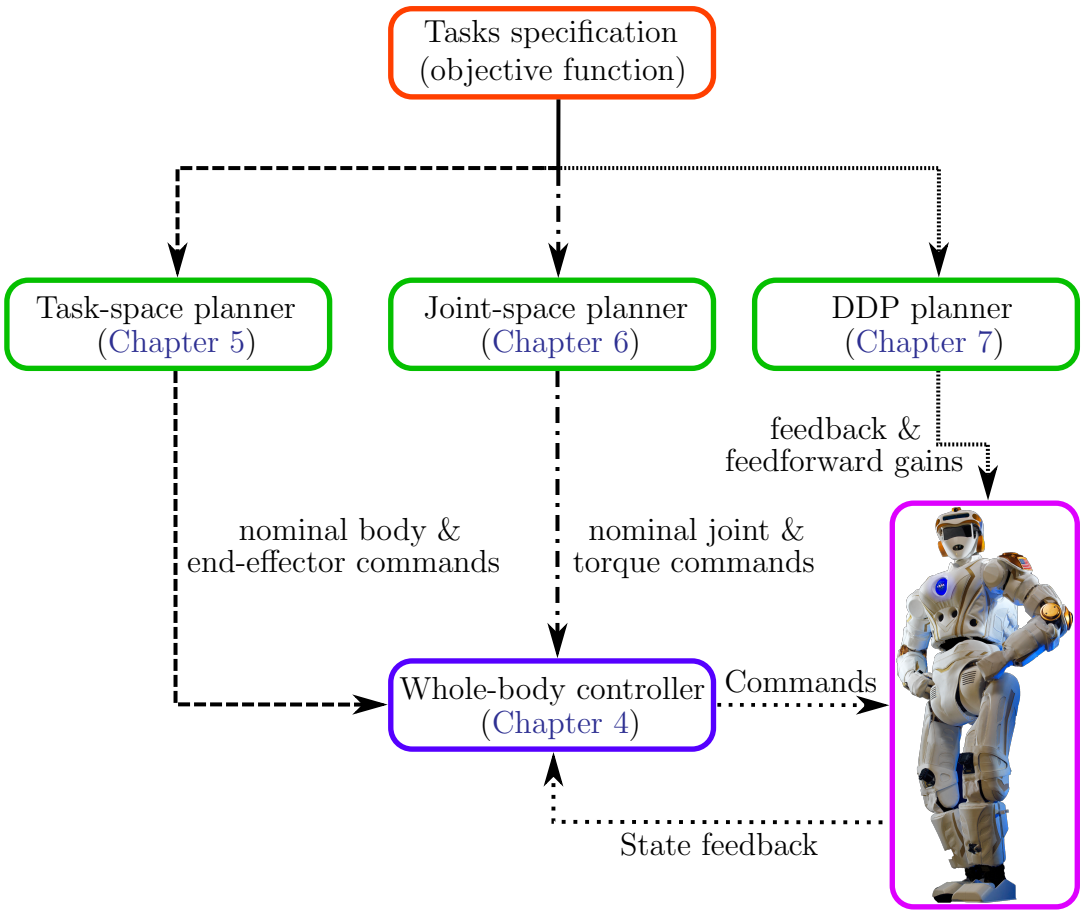


Figure 1.2: Overview of the approaches presented in this thesis.

trade-off is the descriptiveness of the formulation, which is much wider and general than the descriptiveness of the previous methods. This formulation outputs a joint-level plan, which means that in an ideal setting the WBC is not required. Practically a controller is still required due to modelling and environmental mismatches. While a QP-based controller is a viable choice for this case too, the detailed plan allows the usage of a LQR controller by linearizing around the nominal trajectory. The LQR formulation provides a cost-to-go function that can be used in QP-based controller setting (Posa, Kuindersma, and Tedrake, 2016). The advantage is that this QP formulation can also reason about future costs, becoming less greedy and myopic.

The third and final option is to select the DDP-based planner, presented in Chapter 7. This exhibits the same properties as the previous method in terms of problem descriptiveness. It lacks in terms of contact discovery capabilities, but gains in the computation speed aspect. Further, we provide a complete

description that can compute motion plans with full-body contacts, in contrast to the previous approach that requires pre-specification of the possible contact points. This approach outputs directly feedback and feedforward gains, which provide a certain robustness against perturbations about the nominal motion, and can be used directly for commanding the robot—assuming that these perturbations are sufficiently small.

1.3 Contributions

While detailed contributions and discussions about each proposed approach are provided at each chapter independently, it is worth discussing here the overall contributions of this work. These are summarized as follows:

- A *thorough discussion of prior work* in contact-implicit TO approaches for legged locomotion and their relationship with alternative methodologies in the wider legged locomotion motion planning context.
- Motion planning algorithms that *reason about both the kinematics and dynamics* of the models, significantly increasing the generality and complexity of the problem with respect to kinematics only planning.
- All proposed formulations are *expressed in a single, comprehensive form* (i.e. holistically). This simplifies practical implementation because users interact with a single component only via cost function specification. At the same time, these holistic formulations output motion plans consisting of a large number of quantities, enabling straightforward and flexible post-processing.
- A task-space formulation that exhibits *second-order integration accuracy and full translational and rotational dynamics modelling*, without singularities in the description (Chatzinikolaidis, Stouraitis, et al., 2018).
- An extension of the previous work that does not require the gait pattern as input, making it a truly contact-implicit task-space formulation.
- A novel joint-space formulation that computes motion plans with arbitrary contact events without challenging complementarity constraints, while facilitating *environments with hard, slippery, and soft contact properties in a unified manner* (Chatzinikolaidis, You, and Z. Li, 2020).

- A *novel extension of the classical DDP algorithm to systems characterized by implicit dynamics*. Moreover, a novel formulation that leverages the implicit extension for exact contact dynamics resolution in DDP, with application to TO with full-body contacts (Chatzinikolaidis and Z. Li, 2021).

Chapter 2

Prior work

In this chapter, the goal is to present a wider context of motion planning and control approaches for legged robots. More precise connections between relevant work and our contributions are included in each chapter separately.

We start by discussing motion planning approaches that are based on template models in Section 2.1. These tend to be computationally very fast but based on very restrictive assumptions, such as level terrains. Next, motion planning approaches based on task synthesis are presented in Section 2.2. These methods break down the motion planning and control problem in a cascade of sub-problems, with the aim of extending the capabilities of the template models to a larger variety of settings. In Section 2.3, we discuss motion planning methods that look at the problem from a holistic perspective and are more relevant to the approaches presented in this thesis. The focus is primarily on approaches that take into account contacts explicitly in their formulations. Finally, we provide an alternative viewpoint in Section 2.4 based on machine learning methods. These methods are not the main focus of this thesis and only a brief discussion about recent results in deep reinforcement learning is provided.

2.1 Motion planning by template models

The first successful approaches for legged locomotion focused on biped models using approximations of the mechanics of walking. They have a long history and are quite mature. Here, we only discuss the most prominent approaches.

The basic building block are template models (Full and Koditschek, 1999). Templates aim to capture in a succinct way the most important features of a biomechanical behaviour. For example, the models discussed next define a simplified behaviour for a single point with mass (where all of the other bodies' mass is compressed to that) and a single massless leg.

Template models are usually coupled with anchors, *i.e.* more detailed models that contain properties of the specific morphology at hand, for example a bipedal or a quadrupedal morphology. While templates is our focus here, anchors are used more extensively in the following chapters. Our proposed formulations utilize detailed kinematic and dynamic descriptions of the robots involved using multiple legs and joints; descriptions specific to the robot model under consideration.

Before introducing template models in detail, let us define what constitutes walking. According to the Cambridge Dictionary, to walk is

> "To move along by putting one foot in front of the other, allowing each foot to touch the ground before lifting the next."

Walking can be either static or dynamic (Kajita, Hirukawa, et al., 2014). The former means that the projection of the CoM never leaves the *support polygon*, *i.e.* the convex hull of the set of all contact points. The latter means that there exist periods when the CoM can leave the support polygon. In some cases, such as legged systems with many limbs where the support polygon covers a large surface, static walking is an effective approach. In bipedal walking, the greatest part of the walking cycle is the single support phase and as a result, the support polygon is usually smaller than the size of the foot. So the focus will be around concepts used for dynamic walking.

2.1.1 Zero-moment point

An important concept in the definition of template models for legged robots is the ZMP. The ZMP is used to judge for a given motion of a legged robot whether the contact between the ground and the sole can be maintained. It can be defined as "the point where the horizontal component of the moment of the ground reaction forces becomes zero" (Vukobratović and Borovak, 2004).

The following criterion can be used to verify dynamic stability for a legged robot:

> "The ZMP must always exist inside the support polygon."

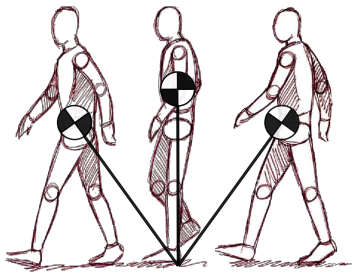


Figure 2.1: Walking approximated by an inverted pendulum.

As a result, it can be used not only to judge whether the system is stable but also the distance from instability. For motion planning, it is useful by quantifying this distance and making sure that it is always satisfied. The CoM is not required to lie inside the support polygon.

It is important to keep in mind the main assumption behind the criterion. Walking must take place in a level terrain. Application to non-level terrains is not straightforward and a number of extensions have been proposed, for example by Hirukawa et al. (2006); Caron, Pham, and Nakamura (2017).

Finally, the ZMP is mostly used for motion planning of humanoid robots. Yet it can be used for high-level stability checks in arbitrary legged structures, such as quadrupeds (Winkler, Farshidian, et al., 2017).

2.1.2 Linear inverted pendulum model

The inverted pendulum model is a model that captures efficiently the dynamics behind walking. According to this model, the stance leg behaves like an inverted pendulum moving about the stance foot (Cavagna, Thys, and Zamboni, 1976); this is graphically depicted in Figure 2.1. Moreover, it is supposed to predict accurately enough the fluctuation of the kinetic and potential energy during walking in humans (Pandy, 2003).

In robotics, the variant of the LIP is used as an efficient and low-dimensional walking pattern generator and is perhaps the most popular template pattern generator for bipeds (Z. Li, Zhou, Dallali, et al., 2014). It allows the modelling of single support phases during walking by a set of linear ODE, and enables real-time computations.

In general, LIP-based motion planning is very versatile. Minor adjustments can

facilitate double support phases and changes in orientation. Furthermore, whilst the CoM is constrained to move on a plane, this is not necessarily horizontal; it can accommodate inclination in situations like moving on stairs. An elaborate study along with more results is provided by Kajita, Hirukawa, et al. (2014).

When designing motion plans with the LIP model, different parameters need to be manually selected, something that in practice requires effort. An approach on how these parameters can be automatically updated during execution is presented by Q. Li et al. (2017). Furthermore, proper timing of the steps is very important for avoiding falls, which is usually pre-specified; setting timings as variables makes the problem nonlinear. Hu et al. (2018) present a computationally fast method for taking into account this nonlinearity.

Finally, this model has some important limitations: It assumes a constant leg length, only one foot contacts the ground at all times, it does not take into account the behaviour of the swing leg, and the CoM is constrained on a planar surface.

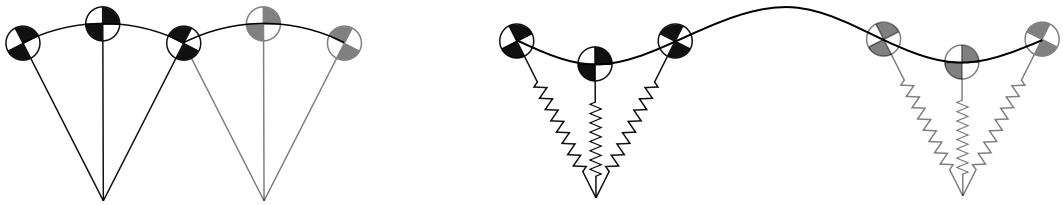
2.1.3 Capture point

The CP is introduced by Pratt et al. (2006) for humanoid push recovery. The main question that aims to address is to provide a computational framework for selecting a step location. Thus, it is defined as "the point on the ground where the robot can step to in order to come to a complete stop". The derivation of the CP for general models is not straightforward. As a result, most works focus on derivations based on the LIP model (close connection with the unstable first-order dynamics part) and variants thereof.

Multiple extensions have been proposed that aim to enhance its capabilities. The issue of taking multiple steps is elaborated in (Wight, Kubica, and D. Wang, 2007; Koolen, Boer, et al., 2012), since the original derivation is focused on a single step only. The CP can be used both as a pattern generator and stabilizer (Englsberger, Ott, Roa, et al., 2011), whilst extensions to the three-dimensional cases have been proposed too, *e.g.* by Englsberger, Ott, and Albu-Schäffer (2015).

2.1.4 Spring-loaded inverted pendulum

The focus of the previous sections was on template models that approximate the dynamics of walking. Running is better approximated by the SLIP, which



a) Inverted pendulum walking. b) SLIP running.

Figure 2.2: Intuitive comparison between: a) the inverted pendulum model, that captures the dynamics of walking, b) the SLIP model, that captures the dynamics of running.

is capable to capture energy storage and release during running cycles (Full and Koditschek, 1999).

In contrast to the LIP model, the original SLIP model can not be computed in closed-form. A closed-form approximation was proposed in (Mordatch, Lasa, and Hertzmann, 2010). Furthermore, an extension to the three-dimensional case with application to humanoid running was presented by Wensing and Orin (2013).

Finally, it is worth mentioning that traditionally walking is associated with vaulting over stiff legs, *e.g.* (Srinivasan and Ruina, 2006), while running with rebounding on compliant legs, as shown in Figure 2.2. More recent studies challenge this dichotomy, based on the observation that stiff legs cannot reproduce the stance leg dynamics during walking. Experimental data shows that compliant legs might be fundamental to walking too (Geyer, Seyfarth, and Blickhan, 2006).

2.1.5 Summary

The power behind all template models comes from the ability to summarize a behaviour in a low-dimensional and tractable manner. Yet this comes with a number of disadvantages that with the increase of computational power are becoming all the more important. A few of the most prominent points of critique are the following:

- It is difficult to quantify how inaccurate the approximations are since they compress the very large dimensionality of a robot to a small number of variables. While this provides important advantages during the motion planning stage, the difficulties are pushed to subsequent stages where anchoring, the mapping from the low-dimensional template model to the whole-body model, happens.

This is discussed further in Section 2.2.

- Each template model focuses on replicating a very specific behaviour. As the need for more agile and dexterous legged robots emerges, a number of behaviours need to be synthesized simultaneously. For example, it is difficult to formulate multi-contact interactions with these simplified models. Such problems lead to non-convex formulations, where the simplicity and computational tractability of a particular model plays a less important role.

- A number of restrictive assumptions are required for the validity of these models, such as co-planar contact locations, contact forces that do not satisfy friction cone constraints, *etc.*

- These models are based on studies that rely on human data, whilst other contact configurations (such as ones found in quadrupeds, hexapods, *etc.*) have received less attention. A swift that took place in the last couple of years demonstrates that robots with more than two legs can be practically more useful, due to their inherent stability properties. Works that focus on generating the structure of the robot and the gait pattern in a given environment automatically suggest that quadrupedal models constitute a more fundamental design for legged robot structures (Zhao et al., 2020).

2.2 Motion planning by tasks synthesis

Motion planning by tasks synthesis focuses on breaking down the locomotion problem into a sequence of tasks, with the goal of making each task simple and tractable. As already underlined, this is particularly important for legged robots due to their high dimensionality coupled with the complexity of the necessary environmental interactions.

There is a long and successful history that utilizes this approach, starting with the seminal work by Raibert (1986). A three-state cyclic approach is presented there that works in a reactive fashion. The basic building blocks are designed for a hopping robot and are partitioned as:

- *Hopping*: The first part of the system excites the cyclical hopping motion by regulating how high the robot hops. This is achieved by delivering a vertical thrust during the support period which sustains the oscillation and regulates

the amplitude.

- *Forward speed*: The second part regulates the forward running speed and acceleration. This is done by moving the leg in an appropriate forward position with respect to the body during the flight phase.
- *Posture*: The last part stabilizes the pitch angle of the body to keep it upright. This is achieved by exerting torque between the body and the leg about the hip.

The succession of these three stages is tracked by a simple state machine, while extensions to biped and quadruped robots are possible via the notion of virtual leg. A couple of drawbacks of this approach are: 1) Proper tuning of all the variables for synchronizing the timings of the state transitions is very important. 2) The foot placement rules rely either on the rapid succession of steps to deal with the mismatch between the true and approximated effect, or on the existence of additional stabilizing mechanisms. 3) Heavy actuation—especially about the hip—is required that can be primarily satisfied by hydraulic mechanisms; these are typically more complicated to operate and maintain with respect to electrically actuated. Although recent advances in electric motors have increased torque density coupled with low gear ratios or even direct drive structures, making the last point less problematic from the electric actuation standpoint.

The previous hierarchy is one of the first successful approaches for legged locomotion and has influenced multiple subsequent studies, *e.g.* by You et al. (2015). Yet the selection of each task is based on empirical observations. Other works focus on different decompositions. But the common theme is that each decomposition is usually hand designed.

For legged robots, the typical decomposition starts with the design of footholds, followed by the CoM motion design (Wieber, Tedrake, and Kuindersma, 2016). Subsequently, the CoM motion is mapped to the whole-body motion. This is usually done via inverse kinematics, but more recent torque-controllable robots allow inverse dynamics formulations too. These stages are typically formulated using either optimization or probabilistic methods, which are discussed next.

2.2.1 Optimization-based methods

2.2.1.1 Foothold selection

For the foothold selection step, a number of approaches have been proposed. In (Chestnutt et al., 2003; D. Zimmermann et al., 2015; Lin, Righetti, and Berenson, 2020), footholds are planned by discrete search-based approaches. Winkler, Mastalli, et al. (2015) propose a similar approach for a quadruped robot. An extension to multi-contact situations is proposed by Escande and Kheddar (2009), where a tree of contact transitions is incrementally built.

Continuous optimization approaches for foothold selection have also been proposed. These are typically formulated in a MPC fashion to allow fast selections and adaptations to disturbances (Herdt, Diedam, et al., 2010; Herdt, Perrin, and Wieber, 2010). Usually, only a small number of footsteps is required. Thus, exhaustive search over the number of footsteps and continuous optimization for finding locations is possible too (Fallon et al., 2015). Due to their continuous nature, these approaches are mostly applicable to flat terrains only.

Finally, approaches based on mixed-integer programming that capture directly the discrete nature of foothold selection have been proposed too. The main benefit with respect to previous approaches is that they can provide some guarantees regarding convergence and completeness even in challenging environments (Deits and Tedrake, 2014; Ponton et al., 2016). Relaxations based on LP can provide computational improvements albeit with limitations (Tonneau, Song, et al., 2020).

2.2.1.2 Body motions

After obtaining footholds (either by an automated procedure or by direct user input), the next step is to compute the motion for the CoM. More traditional approaches focus on biped robots—with fixed foothold locations—and compute the CoM motion using a simplified model (Wieber, 2006). Other approaches start with a set of fixed footholds, compute the desired motion, and then update the footholds when discrepancies between the desired and actual CoM motions happens (Feng, Xinjilefu, et al., 2016). Similarly, for quadruped robots, after the foothold selection, a simplified model is used for the CoM, where the CoP is kept within the support region (Kolter, Rodgers, and Ng, 2008; Kalakrishnan et al., 2011; Mastalli, Havoutis, et al., 2020).

More recent approaches focus on formulations that are able to handle more general multi-contact scenarios. An approach based on a B-spline parametrization of the joints and using full kinematics and dynamics is presented by Lengagne et al. (2013). Kuindersma, Deits, et al. (2016) formulate a sparse nonlinear program for humanoid robots that augments the CoM dynamics with centroidal dynamics (Orin, Goswami, and S.-H. Lee, 2013) and enables multiple non-coplanar contacts. Instead of the centroidal model, the single rigid body model provides a simpler template for multi-contact motion planning (Winkler, 2018). Especially for quadruped robots, the majority of the mass is concentrated on the body and the single rigid body approximation seems reasonable (Di Carlo et al., 2018). While inherently non-convex, a particular Bézier curve parametrization leads to a conservative linear approximation (Fernbach et al., 2020). Finally, rather than using the centroidal model separately from the whole-body model, learning of feasibility constraints (*i.e.* constraints that guarantee that the computed motion for the CoM is achievable by the whole-body inverse kinematics) is presented by Carpentier and Mansard (2018). This allows taking into account approximate whole-body feasibility without explicit evaluation, which is in practice costly.

2.2.2 Probabilistic methods

Probabilistic methods (LaValle, 2006) for motion planning of legged robots have also been proposed but their number is much smaller compared to optimization-based methods. Major factors that impede their implementation are the contact constraints and the underactuation, which complicate motion planning in configuration space (Bouyarmane, Caron, et al., 2019). Taking them into account in optimization-based methods is more straightforward.

Separate stages between foothold selection and CoM motion computation are prevalent in these methods too. The most common approach for foothold selection is RRT (Kuffner et al., 2002; Perrin et al., 2012). Zucker et al. (2013) use a Hamiltonian Monte Carlo approach for sampling from trajectory distributions, with application to computing body motions for a quadruped robot with pre-specified footholds. Bounding motions for a quadruped robot on rough terrain using RRT were computed in (Shkolnik et al., 2011).

General acyclic motion plans for multi-contact scenarios are presented by Bretl (2006) for a rock-climbing robot. An improved approach targeted to legged robots

is presented in (Hauser et al., 2008). Classical multi-contact selection is challenging due to the difficulty of obtaining valid contacts from sampling-based approaches. The samples need to be projected to contact manifolds, an operation which is in practice expensive. To mitigate this issue, Tonneau, Del Prete, et al. (2018) propose to first compute a contact reachable root path and, afterwards, sample statically stable configurations for the limbs along this path.

2.2.3 Summary

Tasks synthesis approaches provide the most common motion planning framework for legged and, in general, multi-contact scenarios. Their power comes from the fact that breaking down the problem to hierarchical levels enables viable computations even in difficult cases, such as foothold selection in challenging environments. From the previous discussion, the typical decomposition is to first plan the footholds with a probabilistic or mixed-integer programming approach, then the CoM motion with an optimization-based approach, and finally compute the joint level commands based on inverse dynamics.

There are two main points of critique for this family of methods. First, the selection of the hierarchy is ad-hoc. That means that the individual stages are selected in a plausible manner; yet the priority is efficient computation. It is difficult to provide an adequate explanation of the decomposition and the best case is to motivate a mechanism inspired by biological studies (*e.g.* as in the case of template models). More importantly, this ad-hoc selection makes it difficult to reason about the interplay of the stages in the hierarchy.

Furthermore, the hierarchical structure leads to an inevitable restriction of the solution space. The most common approach to benchmark this restriction is to compare against a full but computationally challenging model. Since providing exact and quantifiable results is rather difficult, the comparison ends up being done in a small, hand-picked number of cases.

To sum up, task synthesis approaches are typically conservative, while quantifying the degree of conservativeness is also a difficult task. Further, many approaches lack in terms of contact considerations and extending beyond non-flat and non-rigid terrain is not straightforward. As robot applications focus on more dynamic and skilful regimes, the trend is to merge individual stages to holistic and more general.

The goal is to compute a larger variety of motions with as little assumptions as possible. This is the focus of the next section.

2.3 Motion planning by holistic optimization

Holistic optimization approaches focus on tackling the motion planning problem in a more general manner. Instead of breaking down the problem into hierarchical stages, the aim is to have as little stages as possible. Moreover, there is an additional effort—if an explicit hierarchy is required—to make it less rigid, by allowing subsequent stages to modify the information provided. This means that information from previous stages only guide the solution, while the latter stages take into account more complex details of the problem.

The overwhelming majority of the approaches in this category deploy at their core an optimization formulation. Due to the large number of DoF, legged systems typically require complex decisions for motion planning and control. While it is possible to create plans manually, it is in general impractical. An optimization-based approach enables to define performance indices in the form of a single objective, which allows the computations of a continuum of solutions.

In this section, we discuss works that either focus on motion planning for a particular type of legged system (*e.g.* humanoids) or on works that can model general contact interactions (with locomotion being a specific instance). It is worth noting that the majority of research in the past focused on approaches that explicitly avoid contacts, which amounted to undesirable collisions, as for example in (Schulman et al., 2013). Planning through contact requires challenging reasoning with discontinuities due to friction and impacts.

2.3.1 Hybrid optimization

Hybrid optimization approaches usually rely on models that are based on hybrid dynamical systems (Goebel, Sanfelice, and Teel, 2009), and encompass situations where discrete and continuous decisions are combined. The discrete modes are typically pre-specified either by external input (*e.g.* by hand or a probabilistic method) or by outer levels (S. Zimmermann, Hakimifard, et al., 2020). Possible modes—in legged locomotion at least—can define the leg order assignments and possibly the foothold locations or step timings (Wampler and Z. Popović, 2009).

The main idea behind hybrid optimization is to define a mode sequence, which once fixed restricts the overall model's dynamics to a continuous and differentiable regime. This enables efficient gradient-based optimization, since the difficulties of the model (which correspond to the discontinuous decisions by the modes) are eliminated. Discontinuous events can be optimized as follows. A discontinuity corresponds to a jump in the state space. Conditions that activate these jumps are called *guard conditions*. With a fixed mode schedule, these guard conditions are pre-specified. Thus, optimization is performed at each continuous state trajectory between these guards, whilst extra constraints are added that ensure that the guard conditions are met between segments.

This approach is effective when the number of guard condition combinations is relatively small. If there are multiple such conditions, as in locomotion and manipulation, the number of possible combinations grows exponentially (Posa, Cantu, and Tedrake, 2014). Sources of guard conditions in robot motion planning can be: when a foot hits the ground, when a joint limit is reached, when there is switching between a sliding and a sticking contact, *etc.* Another issue that plagues hybrid approaches is that small changes in the control can lead to qualitative different guard activations. This complicates search and makes it particularly challenging to efficiently explore away from the initial mode sequence.

As such, most approaches in legged locomotion focused on humanoid robots, where the possible mode sequences are small and can be efficiently explored. For example, a hybrid optimization approach combined with multiple shooting for human-like running (Mombaur, 2009; Schultz and Mombaur, 2010). In the popular approach of hybrid zero dynamics control, the continuous dynamics correspond to the swing phase while contacts with the ground activate discrete guard conditions (Ames and Poulakakis, 2017). Even though the main focus of hybrid zero dynamics approaches is on the control of legged robots, applications to motion planning have been studied too (Hereid et al., 2016).

2.3.2 Contact-implicit optimization

The goal of contact-implicit optimization is to remove the need to explicitly specify the mode sequence, *i.e.* to become mode-invariant. Decisions about contacts are folded within the problem's formulation and these decisions are retrieved after the respective optimization problems are solved. In general, there are three options

for the description and solution of the problems in this class:

- *Penalization*: The problem is transformed into a continuous one by introducing additional parameters. At every iteration, these parameters are adapted, so that this limiting process converges to a solution of the original problem.
- *Lifting*: Additional variables and suitable constraints (typically in inequality form) are introduced, with the aim of making the optimization solver handle directly the switches via the activation and deactivation of the constraints.
- *Mixed-integer*: Discrete decisions are directly modelled by the introduction of integer (binary) variables that activate and deactivate modes.

Among the first works that proposed solutions based on this approach is the work by Yunt and Glocker (2006). They introduce a penalization scheme with sequential sub-problems, where a distinct set of parameters is updated at every iteration. Another important breakthrough is the idea of contact invariant optimization (Mordatch, Todorov, and Popović, 2012; Mordatch, Popović, and Todorov, 2012). This corresponds to a mixture of the penalization and lifting approaches, where a set of auxiliary variables that affect both the objective and the dynamics of the system is introduced. At the same time, these auxiliary variables are penalized, forcing the optimizer to converge to solutions that violate physics as little as possible. Applications to a humanoid robot by solving for multiple perturbed models simultaneously is presented by Mordatch, Lowrey, and Todorov (2015). A method that relaxes the contact model and automatically adapts it at each iteration to suppress the relaxation is proposed by Önol, Corcodel, et al. (2020).

Posa, Cantu, and Tedrake (2014) present a lifting approach that introduces contact forces as explicit variables, along with constraints that enforce complementarity between normal forces and distances. They pose the problem of contact planning as an MPCC and describe constraints that produce hard contacts at the solution, whilst the selection of contacts is performed by the optimizer. While the focus was on 2D cases, results for complex contact interactions in 3D where presented by Dai, Valenzuela, and Tedrake (2014). Inherent in this formulation is the first-order discretization that captures non-smooth interactions. Approaches that try to introduced higher accuracy have been proposed too: Combining contact-implicit dynamics with variational integrators that enable a structured way of deriving

higher-order integrators (Manchester, Doshi, et al., 2019), and an approach reminiscent of event-driven simulation based on the smoothing of the contact impulses (Patel et al., 2019). Xi, Yesilevskiy, and Remy (2016) use this approach to automatically synthesize motions for a planar biped and quadruped robot, while verifying biological findings in terms of locomotion pattern selection for different speeds.

Finally, mixed-integer formulations that directly handle foothold location and possibly gait selection aspects have been proposed too. The focus is primarily on convex formulations of the problem by introducing integer variables for the decomposition of the non-convex constraints (Valenzuela, 2016; Aceituno-Cabezas et al., 2018). Koolen (2020) utilizes a mixed-integer non-convex formulation that tries to reduce the number of integer variables.

2.3.3 Shooting methods with contacts

Shooting methods with contacts provide a middle ground with respect to the previous approaches. They require neither prior specification of the guard conditions for the activation and deactivation of contacts, nor are constrained to pre-specified surfaces for contacts. This is done by treating contact phenomena implicitly by means of full physical simulations, which are typically required to be at least locally differentiable; with extension to control applications if the additional property of invertibility is met. Due to the diverse nature of the possible motions, carefully designed cost functions are needed that encode the right behaviours for the specified task. Classical approaches utilize evolutionary optimization methods that are performed offline (Al Borno, de Lasa, and Hertzmann, 2013).

Methods that achieve fast computational rates—in real-time or close—are based on DDP. Tassa, Erez, and Todorov (2012) demonstrate results for a humanoid model experiencing full-body contacts. Application to a humanoid robot is discussed in (Koenemann et al., 2015), while results for a quadruped robot in multiple different scenarios is shown in (Neunert et al., 2017). Since contact discovery is limited due to reasons explained in the next section, more recent works focus on combining DDP-based motion planning with additional information such as contact locations and sequence (Mastalli, Budhiraja, et al., 2020) or pre-specification of phases similar to hybrid methods (H. Li and Wensing, 2020).

Table 2.1: Comparison between motion planning methods.

	Hybrid	Contact-implicit	Shooting
Contact discovery	●	●	●
Formulation generality	●	●	●
Small mode efficiency	●	●	●
Solution complexity	●	●	●

● Unfavourable ● Moderate ● Favourable

2.3.4 Summary

Compared to task synthesis approaches, holistic optimization provides a more straightforward and general formulation for motion planning problems. Problem decomposition is usually unnecessary; only hybrid optimization requires one due to the pre-specification of the mode sequence. As a result, formulations tend to be much more general and principled, able to describe a wide range of cases usually by switching objective functions only.

Yet this switching of objective functions introduces two extra sources of complexity. First, these objective functions need to be carefully defined, and there is no automated mechanism for the selection of the individual terms, which tends to be task-specific. Second, due to the generality of the formulation, the solution of the problem is quite sensitive to changes of the objective. This typically translates to the need for careful tuning of the individual cost terms, whilst requiring informed initialization sufficiently close to the desired local minimum.

In terms of their comparative characteristics, hybrid methods tend to be very efficient when prior mode enumeration is possible. As a result, their generality is limited with respect to the other approaches, and their contact discovery capabilities suffer. Subsequently, the solution complexity can be quite good, since the problem description can be well-defined. Formulations based on hybrid optimization can leverage efficient optimization between the discrete transitions.

Contact-implicit formulations demonstrate excellent contact discovery characteristics. This is evident by their ability to fundamentally change the initial contact pattern to fit better the cost function specifications; quite frequently they can be initialized without a valid pattern and converge to one. Their formulation gener-

ality is quite good, although the need for distance calculations practically means that only a few pre-defined surfaces can be considered. Penalization formulations require the solution of a sequence of non-convex problems, while lifting methods require the specification of complementarity constraints; the latter enables the inclusion of logical constraints in a continuous setting.

Mixed-integer formulations do not need distance calculations but are again restricted by the number of integer variables introduced, which is in practice limited too. They are also quite efficient for small modes—although they perform worse than the hybrid ones—without requiring pre-specification. Regarding the solution complexity aspect, mixed-integer approaches tend to perform even worse due to the combinatorial explosion. Convex formulations can be currently utilized on template models only, where convexification of the dynamics and kinematics is possible by hand. Mixed-integer non-convex formulations typically result to brute-force mode enumeration as the decisions that can be eliminated by the branch and bound algorithm are quite limited. This property renders them more challenging than a single nonlinear optimization problem.

Finally, shooting-based approaches excel in terms of the formulation generality. Contact phenomena are resolved implicitly by the underlying simulator. The optimization formulation does not need to explicitly model them, so there is no restriction in terms of the number of contact surfaces and interactions that can be considered. Their performance does not change when small modes are possible, since this property does not affect their structure. While theoretically the contact discovery capabilities are quite general, this would require the full inversion of a physical simulator, which is implausible (Toussaint, Allen, et al., 2018). Thus, the selected initialization and objective act as guides towards valid solutions, as in the case of contact-implicit formulations. In practice, their contact discovery capabilities are less powerful than the contact-implicit formulations, as they need to "bump" upon contacts to take them into account.

Nevertheless, the solution complexity for DDP-based methods has proven in practice quite good, enabling many efficient implementations and real-time results. Their computational advantages stem from the fact that they inherently take into the sparsity of the problem in time, enabling structure-exploiting implementations (Frison, 2015). In contrast to contact-implicit formulations, they are easily extensible to receding horizon settings too. This enables application to diverse

settings, where computation of a single nominal motion plan is not adequate. For example, this can happen in situations where the environment is changing considerably.

We summarize the previous points on Table 2.1.

2.4 Machine learning

Finally, it is worth pointing out that machine learning techniques gained increased interest in the context of legged locomotion the past couple of years. They provide an orthogonal approach to the one discussed before, since they are primarily data-driven methods rather than model-driven. The most prominent approaches try to compute motions in an end-to-end fashion, *e.g.* from pixels to torques, and fit better to the holistic motion planning perspective that was discussed before.

One of their attractive features is their ability to circumvent the real-time implementation problem: once properly trained, they are executed in rates that are acceptable for the dynamics of legged locomotion. Their biggest obstacle for wider acceptance and application is their need for large quantities of data, which is in practice difficult to obtain. Thus, the majority of applications are implemented in simulation only, where data is easier to obtain. Yet real-world applications become all the more frequent, especially for quadruped robots, and they are expected to take a more prominent role in legged locomotion research.

Since machine learning incorporates a wide-range of methods—and discussing all of them is outside of the scope of this thesis—we will briefly discuss approaches based on reinforcement learning using deep neural networks. Such methods were recently proposed and generated successful and impressive results.

2.4.1 Deep reinforcement learning

Reinforcement learning approaches aim to solve motion planning and control problems directly, in an end-to-end fashion. This parallels the holistic optimization perspective; yet they are easier to implement, as holistic optimization approaches require complex design and implementation that is done by experts (Hwangbo et al., 2019).

An approach trained by an adaptive terrain curriculum demonstrates a single policy

of robust trotting that traverses a variety of indoor and outdoor unstructured environments (J. Lee et al., 2020). A structure of multiple neural networks, with reward design for the task and imitation rewards, blends behavioural features to achieve imitation learning for human-style locomotion using motion capture data (Yang, Yuan, Heng, et al., 2020). A novel architecture of multi-expert reinforcement learning is able to extend single-skill capabilities to multi-skill and multi-modal locomotion with coherent fall recovery, trotting, and all dynamic transitions in-between these modes (Yang, Yuan, Zhu, et al., 2020). These neural network-based feedback policies are trained in simulation, and then deployed on real robots, but still demonstrate robustness under scenarios that are never encountered during training.

However, these neural network policies act as reactive feedback control that responds to the proprioceptive state feedback. It is hard to incorporate future target objectives for long-term temporal planning. Even though they are computationally fast to run in real-time, it is difficult to guarantee the long-term stability and optimality of motions, *i.e.* whether the robot will fall in the future or whether a successful sequence of motions is more optimal in terms of energy efficiency, with sufficient stability margins against uncertainties. Recent works aim to mitigate these issues, *e.g.* by Gangapurwala, Mitchell, and Havoutis (2020).

Further, for safety-critical applications, such approaches are not able to provide verifiable validity before execution, especially for hard physical constraints that must be respected in real-world applications. To overcome these limitations, more mathematically principled approaches can take into account knowledge about the constraints of the robot and environments, and provide verifiable long-term stability and feasibility.

Chapter 3

Background

In this chapter, we present the necessary material that form the backbone of the ideas discussed afterwards. The topics that will be presented are:

- *Optimal control* (Section 3.1): It provides a wide range of tools for designing trajectories for the mechanical systems that we are interested in. While important theoretical approaches are discussed, we also explain why TO based on nonlinear programming is the most appropriate approach for legged robots.
- *Mathematical programming* (Section 3.2): TO problems are typically solved via standard nonlinear optimization techniques. In this section, we summarize the basic approaches to mathematical programming.
- *Trajectory optimization* (Section 3.3): It is a mature research field that aims to compute locally optimal solutions for dynamical systems. All our results formulate the contact planning problem as a TO problem, which is subsequently solved as a nonlinear program.
- *Rigid-body dynamics with contacts* (Section 3.4): Since contacts are of fundamental importance to the ideas discussed, we present how contacts are typically resolved in simulation. This provides a standardized and accepted framework for computing contact forces that we incorporate and improve in our formulations.

3.1 Optimal control

The goal of optimal control is to compute an admissible control $u(t)$ so that a cost function is optimized (either minimized or maximized). The ingredients that are typically necessary to do that are the following:

- A scalar cost function that evaluates the possible trajectories according to a performance criterion.
- The state equations of the model, either in continuous time $\dot{x} = f(x, u)$ or in discrete time $x_{i+1} = f(x_i, u_i)$.
- Boundary conditions on the state variables at a fixed initial and possibly fixed final time.
- A set of constraints on the state and control variables.

There are three prominent theoretical approaches to solve optimal control problems:

- The calculus of variations (Liberzon, 2012).
- The maximum principle (Pontryagin, 1987).
- Dynamic programming (Bellman and Dreyfus, 1962).

The solution of optimal control problems by the *calculus of variations* leads to the Euler-Lagrange equations, which must be solved subject to boundary conditions. The main difficulty is that the boundary conditions are specified at the two endpoints. The resulting TPBVP is difficult to solve analytically (NP-hard), except in simplified cases.

The *maximum principle*, which can be regarded as a special case of the calculus of variations, provides a set of locally necessary conditions for optimality. The conditions are analogous to the necessary conditions for the minimum of a smooth function, where the first partial derivative must be zero and the second non-negative. Variables analogous to Lagrange multipliers are introduced, which are called co-state or adjoint variables. These co-state variables lead to transversality conditions which—coupled with the initial value conditions—result again to a TPBVP. Additional state and control constraints increase the complexity further.

The *dynamic programming* approach is based on the *principle of optimality* which states that

> "An optimal strategy has the property that, whatever the initial state and control, the remaining decision must form an optimal control strategy with respect to the state resulting from the initial decision."

For continuous systems, dynamic programming leads to a nonlinear partial differential equation which is called the Hamilton-Jacobi-Bellman equation. The solution of this differential equation provides the value function, which describes the future cost from any particular state and control action. Dynamic programming has been most impactful for discrete systems, where numerical solutions can be more conveniently applied. Its biggest disadvantage is the excessive computer memory requirements. For example, if the state vector is n-dimensional and each state variable can obtain M discrete values, a table of size M^n needs to be maintained, assuming one input only. Having more control input values and a larger state space can worsen things dramatically.

The approach pursued in this thesis is somewhat different from the ones described before. It utilizes techniques from mathematical programming, which focuses on finding the extrema of multivariable functions subject to equality and inequality constraints. The basic approach is to reformulate the optimal control problem in the a typical mathematical programming form. With respect to TPBVP and dynamic programming approaches, the two fundamental advantages are:

- Equality and inequality constraints can be handled straightforwardly both for state and control variables.
- Problems formulated this way do not require the excessive computer storage of dynamic programming.

Next, we present a brief exposition to mathematical programming.

3.2 Mathematical programming

Mathematical programming is concerned with the solution of a multi-variable extremization problem. The variables considered are used to express a certain functional relationship that has to be minimized or maximized. This relationship is referred as the *objective function*. Apart from the extremization of the objective

function, the variables need to satisfy additional equalities and inequalities, called *constraints*. Any set of variables which satisfy all of the constraints constitutes a *feasible solution*. A feasible solution that minimizes (or maximizes) the objective function is an optimal solution to the problem. We can express a mathematical program as

$$\begin{aligned} \min_{x} \quad & l(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & h(x) = 0. \end{aligned} \tag{3.1}$$

An important concept in mathematical optimization is convexity. A set $S \in \mathbb{R}^n$ is a *convex set* if a straight line segment connecting any two arbitrary points of the set lies entirely within the set. Similarly, a function g is a *convex function* if its domain is a convex set and if for any two points in the domain it holds

$$g(\alpha x + (1-\alpha)y) \leq \alpha g(x) + (1-\alpha)g(y), \forall \alpha \in [0,1].$$

The fundamental property is that if the objective function and the feasible region are convex, then any local solution to (3.1) is a global solution. Alternatively, the mathematical problem (3.1) is convex if l and g are convex, and h is linear. In optimal control, the distinction between linear and nonlinear systems is fundamental; linear systems can be more deeply analyzed, while their solution provides a larger set of guarantees. "The great watershed in optimization isn't between linearity and nonlinearity, but convexity and non-convexity" (Rockafellar, 1993).

Mathematical programs are typically partitioned into different classes according to the properties of the functions involved. The main reason for doing so is that different classes can be solved efficiently by exploiting different properties. While this is standard for convex optimization problems, non-convex problems—the primary focus of this thesis—do not exhibit this rich tapestry of classes. Yet, since QP and QCQP formulations are utilized frequently, we underline their hierarchy in Figure 3.1 with respect to other convex classes, *i.e.* LP, SOCP, and SDP.

3.2.1 Karush-Kuhn-Tucker conditions

The KKT conditions are a set of first-order necessary conditions, fundamental in mathematical programming, which form the basis of many computational algorithms. They can be seen as a generalization of the Lagrange multipliers

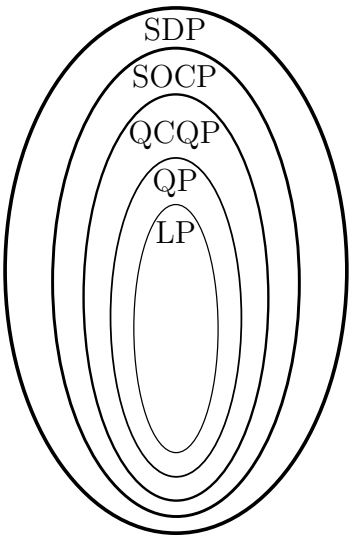


Figure 3.1: Hierarchy of convex programming classes.

method in the presence of inequality constraints. By introducing a set of variables μ_i and λ_j, the Lagrangian of the problem (3.1) is

$$L(x, \mu, \lambda) = l(x) + \sum_i \mu_i g_i(x) + \sum_j \lambda_j h_j(x).$$

The KKT conditions stipulate that if x^* is an optimal point, then there exist multipliers μ_i^* and λ_j^* such that the following conditions hold

$$\nabla L(x^*, \mu^*, \lambda^*) = \nabla l(x^*) + \sum_i \mu_i^* \nabla g_i(x^*) + \sum_j \lambda_j^* \nabla h_j(x^*) = 0 \tag{3.2a}$$

$$g_i(x^*) \leq 0 \tag{3.2b}$$

$$h_j(x^*) = 0 \tag{3.2c}$$

$$\mu_i^* \geq 0 \tag{3.2d}$$

$$\mu_i^* g_i(x^*) = 0. \tag{3.2e}$$

Condition (3.2a) specifies that (x^*, μ^*, λ^*) should form a stationary point of the Lagrangian, conditions (3.2b) and (3.2c) specify that the constraints should be satisfied at the optimal solution, while (3.2d) and (3.2e) enforce that either an inequality will be inactive ($\mu_i = 0$) or that the optimal solution will lie on the boundary of the inequality constraint ($\mu_i \geq 0$).

The conditions relevant to the inequality constraints (3.2b), (3.2d), and (3.2e) are in a complementarity form. Complementarity formulations are in practice difficult to solve. As such, system (3.2) cannot be solved in a closed-form, except

for very specific cases, and optimization methods try to compute approximate numerical solutions. The contact planning problem can also be formulated in a complementarity form, which again is difficult to solve in practice. In Chapter 6 we investigate an alternative and smoothed formulation of this problem that is solved more efficiently.

3.2.2 Sequential quadratic programming methods

SQP methods leverage the fact that it is practically efficient to solve QP problems. At each iteration k, SQP methods approximate (3.1) as (Nocedal and Wright, 2006)

$$\begin{aligned} \min_{p} \quad & \frac{1}{2} p^T \nabla^2_{xx} L(x_k, \lambda_k, \mu_k) p + \nabla_x l(x_k)^T p \\ \text{s.t.} \quad & \nabla_x g_i(x_k)^T p + g_i(x_k) \leq 0 \quad \forall i \geq 1, \\ & \nabla_x h_j(x_k)^T p + h_j(x_k) = 0 \quad \forall j \geq 1, \end{aligned}$$

where p is the step size $x_{k+1} = x_k + p$. Typically, a trust-region constraint is required to control the length and quality of the computed step.

3.2.3 Interior-point methods

IP or barrier methods reformulate (3.1) as an equality constraint problem by replacing inequalities with slacks (Nocedal and Wright, 2006)

$$\begin{aligned} \min_{x,s} \quad & l(x) - \mu \sum_i \log(s_i) \\ \text{s.t.} \quad & g_i(x) = s_i \quad \forall i \geq 1, \\ & h_j(x) = 0 \quad \forall j \geq 1. \end{aligned}$$

This reformulation pushes the combinatorial complexity of identifying active inequalities to the selection of parameter μ that must be appropriately updated per iteration. Intuitively, IP methods generate steps that stay away from the boundaries of the nonlinear constraints for $\mu > 0$, and approach the boundary of the active constraints and the local optima for $\mu \to 0$.

We typically solve the nonlinear programming problems that we present here using an IP solver. This is motivated by the following two points. First, a large number of the inequalities are inactive at the solution. We expect that methods that traverse the interior of the constraints achieve faster convergence. Second, it is desirable for some of the inequalities not to be active at the solution (*e.g.* joint limits) and IP methods inject a bias towards solutions that avoid these limits.

3.3 Trajectory optimization

The goal of TO is to address optimal control problems without the computational difficulties of the methods discussed in Section 3.1. This is achieved by sacrificing global optimality and leads to locally optimal solutions. In practice, TO is more suitable for high-dimensional systems and the solutions are less difficult to compute (Kelly, 2017).

A broad description of the TO problem that we are interested in is

$$\begin{aligned} \min_{t_f, x(t), u(t)} \quad & l_f\left(t_f, x(x_f)\right) + \int_0^{t_f} l\left(x(t), u(t)\right) dt \\ \text{s.t.} \quad & \text{Model's dynamics:} \quad \dot{x}(t) = f\left(x(t), u(t)\right) \\ & \text{Path constraints:} \quad g\left(x(t), u(t)\right) \leq 0 \\ & \text{State bounds:} \quad x_L \leq x \leq x_U \\ & \text{Control bounds:} \quad u_L \leq u \leq u_U \\ & \qquad\qquad\quad t \in [0, t_f]\,. \end{aligned} \tag{3.3}$$

This is is an infinite-dimensional optimization problem, since the variables involved are continuous functions of time, *i.e.* trajectories. Thus, direct solution of this problem is rather difficult. Practical approaches transform it first to finite-dimensional—that is, with a finite set of variables and constraints—by a suitable transcription process.

3.3.1 Transcription

The process of expressing the continuous infinite-dimensional TO problem (3.3) to a finite-dimensional approximation is called *transcription*. Its formulation involves three steps (Betts, 2010):

1. Discretizing (3.3) into a problem with a finite number of variables.
2. Solving this finite-dimensional problem using mathematical programming techniques (Section 3.2).
3. Assessing the characteristics of the solution in terms of quality and accuracy and, if necessary, repeating the previous steps.

While these steps are equally important, much of this thesis is focused on the first and second step only. For reasons that will be clarified further later, contacts

introduce non-smooth aspects in the problem. Thus, most approaches are limited to first-order discretization methods only, where the non-smooth phenomena can be properly captured. Apart from the discussion in Chapter 5, the most straightforward approach to improve the accuracy of the solution is to perform a denser discretization.

3.3.2 Shooting, multiple shooting, and collocation

Single shooting is among the simplest TO techniques. An early application of the method involved the computation of the angle of a cannon in order to hit a target, hence its name (Betts, 2010). The idea is to select the control input and use it to simulate the system. After the trajectory is computed, the next step is to compare the final condition with the desired one. The control input is then improved by changing its value in a way that will decrease the final condition error in the next run, and the whole process is repeated. This method is more suitable for small problems with simple structure and no path constraints, which are outside of the scope of this thesis. Furthermore, single shooting approaches tend to be very sensitive to initial conditions and the control input.

Multiple shooting is an extension of single shooting with the following property: the original problem is broken down to multiple small single-shooting problems, which are solved in parallel. Additional constraints are added at the end of each segment to match the beginning of the next one (defect constraints). With respect to single shooting, this approach reduces sensitivity (Betts, 2010). Its biggest drawback is the difficulty to handle path constraints. While it can be applied to systems with many DoF, a large number of path constraints pose a significant challenge. Collocation can handle them more easily.

Collocation assumes a different perspective and does not require the simulation of the dynamical system. Instead, the trajectory is approximated using piecewise polynomials. Additional constraints are needed to make sure that the polynomials match the system's dynamics at collocation points (knots). Path constraints can be easily integrated by imposing them at the collocation points, while implementation to high-dimensional state spaces is practical (Hargraves and Paris, 1987). Thus, collocation is more appropriate for the problems that are discussed in this thesis. We focus on direct collocation, which approximates the system's dynamics using a large number of low-order polynomials. In Chapter 5, trapezoidal collocation is

utilized, while the rest of the chapters utilize piecewise linear approximations.

3.3.3 Differential dynamic programming

In Chapter 7, we study an extension of DDP for implicitly defined systems. Here, we present a concise discussion on the classical DDP algorithm. DDP is a shooting method. As such, the only variables of the problem are the control inputs, while simulation is required to obtain the state of the model.

DDP was introduced as a second-order optimization method for optimizing discrete-time systems by Mayne (1966). Rather than optimizing the value function throughout the spate space, it focuses on improving the cost-to-go function along the local trajectory. As a result, DDP is not affected by the curse of dimensionality and can be applied to systems with many DoF. Unfortunately, this means it computes a locally optimal trajectory rather than a global one, in contrast to dynamic programming.

Furthermore, it exploits the stage-wise nature of the decision problems at hand. Instead of solving a large optimization problem with a complexity of $O\left(N^3\right)$ for N time periods, DDP solves a small quadratic programming problem at each time step. Thus, its complexity scales linearly $O\left(N\right)$ with the time horizon. In general, it is considered a superior choice if the problem can be described in the DDP framework (Liao and Shoemaker, 1992). A reason why that might not be the case is if complicated path constraints are present.

Another desirable characteristic of DDP is that it demonstrates quadratic convergence similar to the stage-wise Newton method (Murray and Yakowitz, 1984); the latter generates exactly the same steps as Newton's method but with complexity $O\left(N\right)$. Nevertheless, its numerical performance is generally considered superior and we utilize it for efficient TO in the presence of contacts.

A standard discrete dynamical system can be described in explicit form as

$$x' = f(x, u), \tag{3.4}$$

where $x \in \mathbb{R}^n$ denotes the state of our system, $u \in \mathbb{R}^m$ the control, and we use the notation $\cdot'$ to denote the quantity at the next time step, *e.g.* the next state in our context.

Next, we define the cost that we wish to minimize. We utilize the standard additive cost objective function formulation to leverage iterative solutions. The total cost is the sum of the intermediate costs l_i and the final cost l_f, given by

$$L(x_0, U_0) = l_N(x_N) + \sum_{i=0}^{N-1} l_i(x_i, u_i).$$

Here, U_t is the sequence of controls starting from time step t. Further, L is called the *cost-to-go function* and measures the cost from a given instance t

$$L(x_t, U_t) = l_N(x_N) + \sum_{i=t}^{N-1} l_i(x_i, u_i),$$

which is essentially the total cost as measured by starting from an intermediate state. Subsequently, the function that computes the optimal cost from that instance t is central in the dynamic programming literature and is called *value function*, given by

$$V(x_t) = \min_{U_t} L(x_t, U_t).$$

The value function is the total of the cost at a state x_t, once we apply the optimal control sequence or policy U_t; essentially the optimal cost from this state. At the final time step N, the value function is

$$V(x_N) = l_N(x_N).$$

Alternative, we can specify the value function recursively with respect to the next step's value function V' as

$$\begin{aligned} V(x) &= \min_u l(x, u) + V'(x') \\ &= \min_u l(x, u) + V'(f(x, u)). \end{aligned} \tag{3.5}$$

Rather than trying to compute exactly the value function—as in the dynamic programming case—we perform an approximation of the value function along the current trajectory and, afterwards, successive improvements. We approximate the value function using the *Q-function*

$$Q(x, u) = l(x, u) + V'(f(x, u)) \tag{3.6}$$

by performing a quadratic approximation around the current point (x_i, u_i) as

$$Q(x, u) \approx Q(x_i, u_i) + Q_x(x_i, u_i)\delta x + Q_u(x_i, u_i)\delta u$$

Table 3.1: Size of quantities in the quadratic approximation of the Q-function, where n is the number of states and m the number of controls.

1^{st} order expansion	Size
l_x	$1 \times n$
l_u	$1 \times m$
$V'_{x'}$	$1 \times n$
f_x	$n \times n$
f_u	$n \times m$

2^{nd} order expansion	Size
l_{xx}	$n \times n$
l_{xu}	$n \times m$
l_{uu}	$m \times m$
$V'_{x'x'}$	$n \times n$
f_{xx}	$n \times n \times n$
f_{xu}	$n \times n \times m$
f_{uu}	$n \times m \times m$

$$+ \tfrac{1}{2} \begin{bmatrix} \delta x^T & \delta u^T \end{bmatrix} \begin{bmatrix} Q_{xx}(x_i, u_i) & Q_{xu}(x_i, u_i) \\ Q_{ux}(x_i, u_i) & Q_{uu}(x_i, u_i) \end{bmatrix} \begin{bmatrix} \delta x \\ \delta u \end{bmatrix}, \tag{3.7}$$

where $\delta x = x - x_i$ and $\delta u = u - u_i$ are state and input perturbations, respectively. The terms in (3.7) are computed by expanding and matching same power terms in (3.6) as

$$\begin{aligned} Q_x &= l_x + V'_{x'} f_x, \\ Q_u &= l_u + V'_{x'} f_u, \end{aligned} \qquad \begin{aligned} Q_{xx} &= l_{xx} + f_x^T V'_{x'x'} f_x + V'_{x'} f_{xx}, \\ Q_{xu} &= l_{xu} + f_x^T V'_{x'x'} f_u + V'_{x'} f_{xu} = Q_{ux}^T, \\ Q_{uu} &= l_{uu} + f_u^T V'_{x'x'} f_u + V'_{x'} f_{uu}. \end{aligned} \tag{3.8}$$

At this point, it is worth summarizing in Table 3.1 the sizes of the previous quantities for clarity.

The Q-function depends on terms of the current and next time step. The DDP algorithm leverages that by performing at every iteration (i) a *backward pass* starting from the end of our current trajectory to compute the optimal change in the control δu^* that will decrease the Q-function, and (ii) a *forward pass* to compute the new control sequence U_0 and corresponding state.

3.3.3.1 Backward pass

During the backward pass, the optimal control change δu^* is first computed by minimizing the unconstrained quadratic equation (3.7) as

$$\delta u^* = u^* - u_i = \underset{u}{\operatorname{argmin}}\, Q(x, u) - u_i = \underbrace{-Q_{uu}^{-1} Q_u^T}_{\kappa} \underbrace{-Q_{uu}^{-1} Q_{ux}}_{K} \delta x = \kappa + K \delta x. \tag{3.9}$$

Next, the solution is plugged in (3.7) to obtain the quadratic approximation of the value function at the current time step as

$$\delta V = V(x) - Q(x_i, u_i) = \tfrac{1}{2} Q_u \kappa, \tag{3.10a}$$

$$V_x = Q_x + Q_u K, \tag{3.10b}$$

$$V_{xx} = Q_{xx} + Q_{xu} K. \tag{3.10c}$$

This will serve at the next step of DDP as the approximation of the next step value function for computing the quantities in (3.8), since we are iterating backwards in time.

For the initial step of the backward pass, the boundary values are given by

$$V_x^N = l_x^N,$$
$$V_{xx}^N = l_{xx}^N.$$

3.3.3.2 Forward pass

Once the feedforward and feedback terms κ_i and K_i for each time step are computed, the forward pass is run to compute the updated control sequence as

$$\hat{x}_0 = x_0, \tag{3.11a}$$

$$\hat{u}_i = u_i + \delta u^* = u_i + \kappa_i + K_i(\hat{x}_i - x_i), \tag{3.11b}$$

$$\hat{x}_{i+1} = f(\hat{x}_i, \hat{u}_i), \tag{3.11c}$$

for $i \in [0, N-1]$. At the end of the forward run, the total cost of the updated trajectory is computed and compared with the cost from the previous step. If the decrease is smaller than a selected tolerance or if a stationary point is found, DDP is complete. Otherwise, a new step is computed, *i.e.* a backward and forward pass sequence is performed.

3.3.3.3 What can go wrong

There are two things that can fail during a DDP iteration (Yakowitz, 1989):

1. The computed policy does not improve the total cost. This happens because the quadratic approximation of the model is not accurate enough and the resulting trajectory falls outside of the region of validity (Tassa, 2011).

2. The matrix Q_{uu} is not positive definite, which is necessary for computing a descent direction.

Regarding the first issue, the standard approach is to use a line search procedure to compute the updated control sequence during the forward pass as

$$\hat{u}_i = u_i + \alpha\kappa + K(\hat{x}_i - x_i),$$

for decreasing values of $0 < \alpha \leq 1$, until the total cost is improved.

The second issue is more involved. A standard approach is to modify Q_{uu} as $Q_{uu} + \mu\mathrm{I}$, to become positive definite. This corresponds to adding an extra quadratic cost around the current control sequence, making changes to control more conservative. Alternatively, Tassa (2011) proposed to modify Q_{uu} and Q_{xu} in (3.8), by replacing $V'_{x'x'}$ with $V'_{x'x'} + \mu\mathrm{I}$. This corresponds to adding an extra quadratic cost around the current state sequence, making changes to the state more conservative. In our experience, performance depends on the system at hand and none is strictly better than the other.

As in the Levenberg–Marquardt algorithm, the damping factor μ should be adjusted at each iteration. A quadratic scheduling approach works well across different models (Tassa, Erez, and Todorov, 2012).

Given that both the line search procedure and the regularization require modification of the original algorithm, the new value update is computed as

$$\begin{aligned}
\delta V &= \frac{\alpha^2}{2}\kappa^T Q_{uu}\kappa + \alpha Q_u^T\kappa, \\
V_x &= Q_x + K^T Q_{uu}\kappa + Q_u^T K + Q_{xu}\kappa, \\
V_{xx} &= Q_{xx} + K^T Q_{uu} K + Q_{xu} K.
\end{aligned}$$

3.4 Rigid-body dynamics with contacts

Since contact phenomena are paramount to the topics discussed in the thesis, we present how contacts can be taken into account for rigid-body models undergoing contact interactions. Two approaches are in general popular for incorporating contact phenomena in robotic simulation:

- *Hybrid*: The model evolves in a continuous regime of the state-space until a guard condition (contact) is met. At this point, a discrete transition (jump) takes place and the model starts evolving from the new point. Typically, they require specification of the transitions beforehand. More details in the context of hybrid optimization are provided in Section 2.3.1.

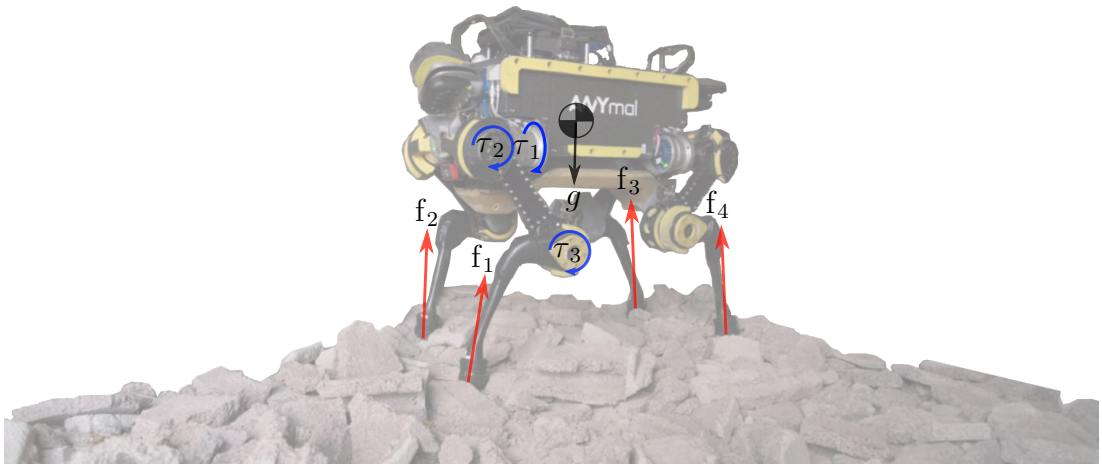


Figure 3.2: Gravitational force, contact forces, and joint torques of a single leg for the quadruped robot ANYmal (Hutter et al., 2016).

- *Time-stepping*: They discretize the dynamics expressing the problem in the velocity–impulse level. Contact events that take place between the discretizations are lumped together at the end of each time step. These approaches allow experiencing multiple contacts simultaneously.

A particular hybrid approach that is suitable for legged robots is discussed in Chapter 5 in more detail. But our main focus is contact-implicit approaches that do not require a priori specification of the contact activation pattern and are based on time-stepping. Thus, a brief description of the impulse-based time-stepping approach by Stewart and Trinkle (2000); Stewart (2000) is presented, which forms the basis of the methods discussed in Chapters 6 and 7.

3.4.1 Time-stepping formulation

The equations of motion of a typical robot model are

$$M(q)\dot{v} + H(q, v) = S\tau + \sum_k J_k^T(q)\mathrm{f}_k, \tag{3.12}$$

where M is the mass matrix, H the vector of nonlinear forces (*e.g.* Coriolis, centrifugal, and gravitational), S is a selection matrix that maps actuated joint torques τ to generalized coordinates, and J_k denotes the Jacobian of the k-th contact and f_k the corresponding force. We simplify the notation by dropping explicit dependence on quantities and write M instead of $M(q)$, *etc.* Inputs to this equation are depicted in Figure 3.2.

Next, equation (3.12) is discretized using an explicit Euler numerical integration method

$$M_i\,(v_{i+1} - v_i) = dt(S\tau_i - H_i) + J^T\lambda_i,$$

where dt is the size of the time step i, and λ is the concatenation of all contact impulses. The discrete Euler dynamics are projected in contact space by multiplying first with the inverse inertia matrix and, afterwards, by multiplying with the concatenated Jacobian matrix J

$$\begin{aligned} \underbrace{Jv_{i+1}}_{\mathrm{v}^+} &= \underbrace{Jv_i + dtM_i^{-1}(S\tau_i - H_i)}_{\mathrm{v}^-} + \underbrace{JM_i^{-1}J^T}_{A}\lambda_i \Rightarrow \\ \mathrm{v}^+ &= A\lambda_i + \mathrm{v}^-, \end{aligned} \tag{3.13}$$

where v^+ is the next step velocity in contact space, v^- is the velocity in the absence of contacts, and A is the inverse inertia matrix in contact space.

Furthermore, the contact impulses λ_i must satisfy additional inequality constraints, as specified by the selected contact model. Three very common contact models in robotics are discussed in Section 3.4.2.

At this point, it is worth mentioning two alternative views on (3.13) depending on the available quantities (Todorov, 2014):

- *Forward dynamics with contacts*: Quantities A and v^- are known (joint torques τ_i are given), while the goal is to compute λ_i and v^+.
- *Inverse dynamics with contacts*: Quantities A and v^+ are known, while the goal is to compute λ_i and v^- (the joint torques are unknown).

The first approach is typically utilized in simulation, and also in most TO motion planning approaches. The second viewpoint has received much less attention, mainly because the contact model needs to be invertible. In Chapters 6 and 7 we present two approaches based on the inverse dynamics with contacts viewpoint in the context of collocation and DDP, respectively.

Finally, in Chapter 7 we present an approach that avoids the time-stepping discretization and is formulated in the acceleration–force space directly. As we discuss there too, by formulating in this space there are two distinct advantages. First, a constant contact Jacobian J is not required, as it is assumed in the velocity–impulse formulation. Second, high-order integration is straightforward

to implement, while time-stepping approaches are confined to a first-order from their design.

3.4.2 Contact models

We discuss three contact models of interest in robotics simulation and control (Horak and Trinkle, 2019):

- NCP (Nonlinear complementarity problem).
- BLCP (Bounded linear complementarity problem).
- CICM (Convex and invertible contact model).

The first model corresponds to the classical complementarity-based formulation of rigid body dynamics with non-smooth contacts. The second model is important because it is the most commonly used in simulation engines, such as the Open Dynamics Engine (ODE) and Bullet. Our focus will be on the CICM model, which is invertible and can be used for formulating inverse dynamics with contacts.

3.4.2.1 Nonlinear complementarity problem

The NCP formulation augments (3.13) with the following constraints

$$\text{Friction cone:} \quad \sqrt{\lambda_k^t + \lambda_k^o} \leq \mu \lambda_k^n \tag{3.14a}$$

$$\text{Impulse/gap distance complementarity:} \quad 0 \leq \lambda_k^n \perp \phi_k^+ \geq 0, \tag{3.14b}$$

where $\lambda_k = [\lambda_k^t \;\; \lambda_k^o \;\; \lambda_k^n]^T$ is the impulse at the k-th contact, and ϕ_k^+ is the next step contact normal gap distance. Constraint (3.14b) is a shorthand notation for

$$\begin{aligned} \lambda_k^n &\geq 0 \\ \phi_k^+ &\geq 0 \\ \phi_k^+ \lambda_k^n &= 0, \end{aligned}$$

which have the same form as the complementarity conditions that were discussed in Section 3.2.1. Their difficulty is even more important for simulation; their implicit form requires solving systems of nonlinear equations. This is impractical for the rates that simulations typically operate. For TO this is not such an important issue since the problem formulation is already in implicit form.

To avoid the nonlinear complementarity form in simulation, the next step gap distance is approximated by a Taylor expansion around the gap distance of the current time step

$$\phi^+ = \phi + dt\dot{\phi} + O(dt^2) \approx \phi + dt\dot{\phi},$$

where $\dot{\phi}$ is the gap velocity. By combining (3.13), constraint (3.14b) is expressed as

$$0 \leq \lambda_k^n \perp \phi_k + dt\dot{\phi}_k \geq 0 \Rightarrow 0 \leq \lambda_k^n \perp (A\lambda_k + \mathrm{v}^-)_n + \frac{\phi_k}{dt} \geq 0. \tag{3.15}$$

While (3.15) is in a LCP form, (3.14a) makes the whole problem nonlinear.

Additionally, the contact impulse should satisfy the MDP, which specifies for a single contact point that (Moreau, 2011):

> "The friction forces should maximize the dissipation of kinetic energy at the contact."

This practically means that the contact impulse should assume a value that prohibits sliding when it lies inside the friction cone. If sliding occurs, the contact impulse should lie on the boundary of the cone opposing the sliding velocity. This can be mathematically expressed as

$$\begin{aligned} \min_{\lambda^f} \quad & v^T J_f^T \lambda^f \\ \text{s.t.} \quad & \left\|\lambda^f\right\| \leq \mu\lambda^n, \end{aligned} \tag{3.16}$$

where $\lambda^f = [\lambda^t \ \lambda^o]^T$ are the friction components of the contact impulse, and J_f is the Jacobian mapping the friction components to joint space.

In terms of solution, this is typically done using iterative solution approaches. Solvers first compute the normal component by (3.14a) and (3.15), and subsequently the tangential components are computed by solving (3.16).

3.4.2.2 Bounded linear complementarity problem

The BLCP model is the most commonly used contact model in robotics simulation. The difference with respect to the NCP model is the linearization of the friction cone. This turns the problem into an LCP which can be solved more efficiently, even though it is NP-complete too.

LCP constitutes a generalization of LP and a special case of QP. Two are the standard approaches for the solution of an LCP:

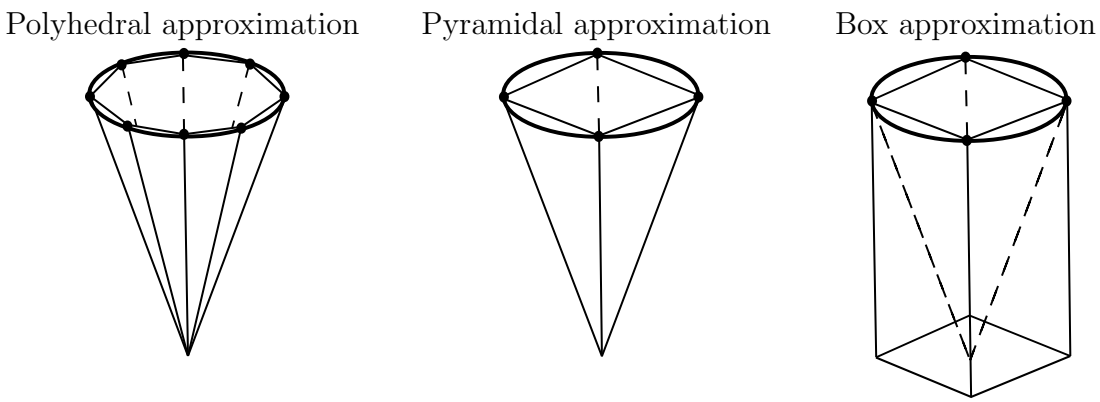


Figure 3.3: Common linearizations of the Coulomb friction cone: (a) A polyhedral cone approximation becomes arbitrarily close with the original cone by increasing the number of edges. (b) The most commonly used approximation is the four-sided pyramidal. (c) The simplest approximation is achieved using a box.

- Lemke's algorithm is a pivoting algorithm similar to the Simplex algorithm for LP problems (Murty, 1997).
- The PGS algorithm is a generalization of the standard Gauss-Seidel algorithm. Each iteration computes first the contact impulse using the Gauss-Seidel method without the linearized friction cone constraint, and afterwards the solution is projected back to the friction cone (Horak and Trinkle, 2019).

The linearization of the friction cone is done using a polyhedral approximation. In general, there are two flavours:

- Circumscribed approximation: The polyhedral cone is an outer approximation of the friction cone.
- Inscribed approximation: The polyhedral cone is an inner approximation of the friction cone. This case is shown in Figure 3.3.

As an example, the constraints for the pyramidal approximation of the friction cone are

$$\begin{aligned} \left|\lambda^t\right| &\leq \tilde{\mu}\lambda^n \\ |\lambda^o| &\leq \tilde{\mu}\lambda^n, \end{aligned}$$

where $\tilde{\mu} = \mu$ for the circumscribed and $\tilde{\mu} = \frac{\mu}{\sqrt{2}}$ for the inscribed approximation, for a total of four inequalities. This model is usually the preferred choice since it

provides a good balance of accuracy and number of inequalities.

Finally, these approximations are also used as constraints for the contact forces in a WBC based on QP, which is discussed in Chapter 4.

3.4.2.3 Convex and invertible contact model

The CICM contact model removes the complementarity formulation altogether. It was first introduced by Todorov (2011) and further refined and expanded by Todorov (2014). It forms the backbone of the popular physics engine MuJoCo.

The model possesses some important properties that differentiate it from the previous two models. First, it is convex, which implies that there is a unique solution for each contact configuration. This means that the rigid contact assumption of the previous two models no longer holds. More importantly for TO settings, this model is invertible, with the inverse having a closed-form solution. As far as we know, this is the only contact model that possesses such properties, and the latter two are exploited in Chapters 6 and 7.

In terms of formulation, instead of assuming the MDP standpoint, this model seeks to minimize the next step velocities in contact space. This leads to a complementarity-free formulation, which is convex. The problem is a QCQP, if the Coulomb friction cone is assumed, or a QP, if a linearized version is used. The resulting optimization problem has the following form

$$\begin{aligned} \min_{\lambda_i} \quad & \frac{1}{2}\lambda_i^T(A+R)\lambda_i + \lambda_i^T(\mathrm{v}^- - \mathrm{v}^*) \\ \text{s.t.} \quad & \lambda_i \in \mathbb{F}_{\mu_i}, \end{aligned} \tag{3.17}$$

where $\mathbb{F}_{\mu_i} = \left\{ \lambda_k^{(i)} \mid \lambda^{n(i)} \geq 0, \sqrt{\lambda^{t(i)} + \lambda^{o(i)}} \leq \mu_i \lambda^{n(i)} \right\}$ is the set of impulses that satisfies the Coulomb friction model, R is a positive definite matrix (typically diagonal) that renders the problem convex, and v^* is a reference velocity for stabilization. While a general Baumgarte stabilization form (Baumgarte, 1972) can be valid,

$$\mathrm{v}^* = k_d\phi + k_v\dot{\phi}, \tag{3.18}$$

for the cases examined in this thesis a velocity damping with $k_d = 0$ and $k_v = \frac{1}{dt}$ is adequate (Horak and Trinkle, 2019).

Algorithm 1: Project impulse to friction cone.

Input: Unconstrained impulse λ, friction coefficient μ.

Output: Friction impulse projected on friction cone λ_P.

$n := \lambda^n$
$t_f := \left\|\lambda^f\right\|$
if $t_f \leq \mu n$ **then**
    $\lambda_P := \lambda$
else if $t_f \leq -\frac{n}{\mu}$ **then**
    $\lambda_P := 0$
else
    $p_n := \frac{\mu t_f + n}{\mu^2 + 1}$
    $\lambda_P := \begin{bmatrix} \frac{\mu p_n \lambda^t}{t_f} & \frac{\mu p_n \lambda^o}{t_f} & p_n \end{bmatrix}$

The inverse contact dynamics case is computed by

$$\begin{aligned} \min \lambda_i \quad & \frac{1}{2}\lambda_i^T R \lambda_i + \lambda_i^T (\mathrm{v}^+ - \mathrm{v}^*) \\ \text{s.t.} \quad & \lambda_i \in \mathbb{F}_{\mu_i}. \end{aligned} \tag{3.19}$$

With a diagonal R matrix this problem can be solved in closed form. In (Todorov, 2014), a closed-form solution is provided for frictionless contact; contact with sliding friction; contact with sliding and torsional friction; contact with sliding, torsional, and rolling friction. For legged robots, contacts with sliding friction are typically adequate. In this case, each contact impulse is given by

$$\lambda_i = P_\mu\{-R^{-1}(\mathrm{v}^+ - \mathrm{v}^*)\},$$

where P_μ projects the unconstrained solution into the friction cone. In Algorithm 1, a procedure to achieve this is described, as defined in (Tasora and Anitescu, 2011).

Finally, it is worth discussing its differences with other prominent contact models. With respect to complementarity-based models, the differences were already explained. Another important family is spring–damper contact models (Haddadi and Hashtrudi-Zaad, 2012). These models specify the contact force independently for each contact. Their main drawback is that they lead to stiff differential equations that require small time steps. This can make them impractical for TO applications since small time steps lead to large problems. The CICM avoids this issue due to matrix A, which couples contact impulses during forward dynamics.

Conceptually, this model corresponds to multiple springs that are adapted to produce a consistent solution. As a result, it enjoys large step sizes similar to time-stepping approaches.

Chapter 4

Whole-body control

This chapter presents a discussion on WBC approaches for legged robots. These are among the most successful approaches for controlling high-dimensional robots due to their flexibility. Specifically, their main advantage is their generality that enables their application in a different number of settings, while keeping salient aspects of their formulation fixed. For example, WBC is deployed for locomotion in humanoid and quadruped robots, but also for manipulation, and combinations thereof. The DARPA Robotics Challenge provided the starting point for multiple WBC approaches that are now mature. Furthermore, most of the robots that participated in the challenge utilized some form of WBC, showcasing the advantages of this approach.

WBC approaches typically demonstrate compliant full-body mobility with the following characteristics:

- They are based on a small set of rules, such as stability, joint-limit avoidance, self-collision avoidance, *etc.*
- This rules are combined to successfully execute a single (but not specific) task or multiple tasks simultaneously.
- The goal is to exploit all the degrees of freedom of the high-dimensional robots involved, since redundancy is a common property. This is especially important for floating-base legged robots.

WBC approaches were originally developed to address the limitations of position control methods, which are based on inverse kinematics. While position control

approaches lead to simpler and faster implementations, their main drawback is that they typically lead to stiff behaviours. For legged robots—if excellent knowledge of the environment is assumed—then position control is a viable approach. Yet in practice there is an uncertainty both in terms of the environment and the contact information (*e.g.* uncertainty whether a contact is activated or not), which favours compliant approaches, such as WBC.

4.1 Prior work

The starting points for the majority of the WBC approaches is the impedance control work by Hogan (1985) and the operational space formulation by Khatib (1987), which form the theoretical basis of multiple subsequent approaches. One of the first approaches to explicitly apply WBC to humanoid robots is the work by Kajita, Kanehiro, et al. (2003). The idea is to provide a reference in the form of momentum, which is then mapped back to the joint-level by generating a whole-body motion.

WBC approaches are classified in two large categories (Moro and Sentis, 2019):

1. *Velocity-based WBC*, where the output command is joint-level velocity signals.
2. *Torque-based WBC*, where the output command is joint-level torque signals.

Next, both approaches are discussed, although the primary focus are torque-based approaches which form the basis of our controller implementation.

4.1.1 Velocity-based whole-body control

Velocity-based approaches can be broadly classified again into two categories (Moro and Sentis, 2019):

1. *Closed-form solutions*, where the generalized velocity command is computed by means of algebraic manipulations.
2. *Optimization-based solutions*, where the generalized velocity command is computed via numerical optimization.

4.1.1.1 Closed-form solutions

Closed-form solutions compute the generalized velocities in the form

$$v = \tilde{\mathrm{J}}_1^{\dagger}\mathrm{v}_1 + \ldots + \tilde{\mathrm{J}}_N^{\dagger}\mathrm{v}_N, \tag{4.1}$$

where v is the desired task in velocity form, $\tilde{\mathrm{J}}_j$ a differential transformation that maps task velocities to joint velocities for the j-th task, and N is the number of tasks. Notation $(\cdot)^{\dagger}$ denotes a generalized inverse of a matrix. This is typically a MP pseudo-inverse, but any G-inverse matrix is suitable (Udwadia and Kalaba, 1996). The MP pseudo-inverse for a real matrix $\tilde{\mathrm{J}}$ with linearly independent rows is defined as

$$\tilde{\mathrm{J}}^{\dagger} = \tilde{\mathrm{J}}^T \left(\tilde{\mathrm{J}}\tilde{\mathrm{J}}^T\right)^{-1}. \tag{4.2}$$

In case of no preference about the N tasks, meaning that equal significance is assigned, then the differential transform is equal to the Jacobian matrix of the task $\tilde{\mathrm{J}} = \mathrm{J}$. For more than one tasks ($N > 1$), there is no guarantee whether all or any of the tasks will be fully satisfied, especially if the tasks are conflicting.

Alternatively, if a hierarchical structure is specified, it is possible to enforce it by making sure that tasks lower in the hierarchy are projected to the null space of the tasks higher (Nakamura, Hanafusa, and Yoshikawa, 1987; Dietrich, 2016). In that case, the differential transformation is defined as

$$\tilde{\mathrm{J}}_j = \mathrm{J}_j \prod_{k=j+1}^{N} \mathcal{N}_k, \tag{4.3a}$$

where $\mathcal{N}_j$ is the null space projector of the j-th task, with $\mathcal{N}_j = \mathrm{I} - \mathrm{J}_j^{\dagger}\mathrm{J}_j$. Another option is to introduce weighting directly in (4.1) by properly selecting weights w that directly define the hierarchy as (Bouyarmane and Kheddar, 2018)

$$\tilde{\mathrm{J}}_j = w_j \mathrm{J}_j, \tag{4.3b}$$

with w_j specifying the relative importance of the j-th task. The first option has the advantage that less tuning in required, although a strict order needs to be defined which does not provide flexibility for better task satisfaction. The second option is more computationally efficient and straightforward to implement, although it requires careful weight tuning. Thus, the choice depends on the specifics of the application considered.

Furthermore, it is common to introduce weighting at the task level. The pseudo-inverse matrix with weighting is defined as (Gienger, Toussaint, and Goerick, 2010)

$$\tilde{\mathrm{J}}^{\dagger} = R^{-1}\tilde{\mathrm{J}}^{T}\left(\tilde{\mathrm{J}}R^{-1}\tilde{\mathrm{J}}^{T}\right)^{-1}. \tag{4.4}$$

The most common weighting matrix is the inertia matrix, which leads to the inertia-weighted generalized inverse for $R = M$. This specific choice tries to fulfill the task while minimizing the kinetic energy (De Sapio, Khatib, and Delp, 2006).

It is worth pointing out that incorporating inequality constraints is challenging with this approach, like collision avoidance. While there exist approaches for unilateral constraints (Mansard, Khatib, and Kheddar, 2009), more general inequalities can be handled by optimization-based formulations better.

4.1.1.2 Optimization-based solutions

Optimization-based solutions provide a more general approach to WBC since the closed-form case can be alternatively expressed as a least-squares problem subject to linear equality constraints. Their main drawback is that they require an iterative solution strategy; the increased iterations can cause a computational bottleneck. Since optimization problems are typically more computationally intensive than their closed-form counterparts, this is partially mitigated by focusing on implementations based on QP, leveraging its speed and attractive properties.

For a single task or tasks of the same importance, a linear equality or inequality can be selected for each task. The resulting system is expressed as

$$A_j v = b_j, \tag{4.5a}$$

$$C_j v \leq d_j. \tag{4.5b}$$

This system can be expressed by a soft optimization problem as (Kanoun, Lamiraux, and Wieber, 2011)

$$\begin{aligned} \min_{v} \quad & \sum_{j} \|A_j v - b_j\| + \|w_j\| \\ \text{s.t.} \quad & C_j v - w_j \leq d_j. \end{aligned} \tag{4.6}$$

In the case of a hierarchy of tasks, a sequence of QP problems needs to be solved, where the number of QP problems is equal to the the number of hierarchical levels. The general idea is to solve the lower hierarchy tasks by strictly enforcing the

optimal solutions of the higher priority tasks, which are solved first. More details can be found in (Escande, Mansard, and Wieber, 2014).

4.1.2 Torque-based whole-body control

Torque-based WBC is similar to the velocity-based one; that is, closed-form and optimization-based solutions exist too. While the solution techniques remain the same, torque-based approaches rely on different quantities for the definition of the tasks, *i.e.* generalized accelerations, joint torques, and wrenches.

4.1.2.1 Closed-form solutions

In the case of a closed-form solution approach, the task now is described in terms of generalized accelerations/joint torques/wrenches that need to be translated to joint-torques. This assumes a similar form as in (4.1), which is

$$\tau = \tilde{\mathrm{J}}_1^T F_1 + \ldots + \tilde{\mathrm{J}}_N^T F_N. \tag{4.7}$$

Again, if all tasks are assigned equal priority then $\tilde{\mathrm{J}} = \mathrm{J}$. In case of a hierarchy, either of the approaches presented in (4.3) is again valid.

An important issue in legged locomotion is that the robot models involved are underactuated. As such, projecting the computed torque in the actuated DoF is critical. Satisfaction of the constraints due to underactuation can be achieved by proper projection matrices (Mistry and Righetti, 2011).

Finally, passivity-based control is an alternative approach to achieve compliant torque-based task satisfaction in closed-form (Hyon, Hale, and Cheng, 2007; Hyon, 2009). There are a number of attractive properties of passivity-based approaches. First, they do not require a precise kinematic and dynamic models. Furthermore, they do not require inverse kinematics and dynamics computations, and no measurement of the contact forces. Additionally, whole-body compliance is achieved in the presence of unknown contact forces in arbitrary links of the robot. The biggest disadvantage of passivity-based approaches is the quasi-static dynamics assumption. As a result, dynamic motions are difficult to execute, since the quasi-static model is valid on the low velocity regime only.

4.1.2.2 Optimization-based solutions

Similarly to the velocity-based WBC, optimization-based formulations are capable to handle tasks that are expressed as inequalities in the torque level. This is particularly desirable for legged robots due to the contact forces: According to the widely used Coulomb contact model, apart from unilaterality (which can be integrated to closed-form approaches), the friction cone constraint is more difficult to handle in closed-form approaches—even when using a linearized approximation, excluding the box model (Figure 3.3). This provides a salient advantage to optimization-based formulations because QP formulations can handle linear inequalities straightforwardly (Saab et al., 2013). In the passivity-based control context, this property is used by Henze, Roa, and Ott (2016) to distribute the desired CoM wrench to the available contacting end-effectors, while taking into account the Coulomb friction model constraints. A similar approach for computing the ground reaction forces subject to friction cone constraints for quadrupedal motion execution on a challenging terrain is used by Focchi et al. (2017).

Furthermore, another important advantage of optimization-based formulations is that they do not require special projection matrices to handle underactuation. This happens because the dynamics equation (3.12) already accounts for this via the $S\tau$ term. Moreover, this term is necessary because it ensures that the whole-body dynamics are satisfied.

4.2 Optimization-based formulation

As already discussed, optimization-based WBC approaches are particularly suitable for robots interacting with the environment in the presence of inequality constraints and underactuation. This was especially evident during the DARPA Robotics Challenge, where the majority of teams used some form of QP-based control in order to translate task space plans to joint space torque commands. This leverages the fact that the instantaneous dynamics with contacts can be expressed linearly (Kuindersma, Permenter, and Tedrake, 2014). Among the DARPA entries that used robots with purely legged locomotion capabilities (Krotkov et al., 2017), the best performing ones used a variant of this approach. Next, more details about the first three legged locomotion entries are provided.

4.2.1 Whole-body QP-based formulations background

In Team IHMC's entry (Koolen, Bertrand, et al., 2016), desired motion tasks are expressed as accelerations (joint-space, spatial, and point accelerations) that are mapped to momentum and desired external wrenches (ground reaction forces and grasping forces). The resulting joint accelerations and wrenches that achieve the desired momentum specifications are computed through a QP, subject to the momentum balance equations (Orin, Goswami, and S.-H. Lee, 2013) and contact forces constraints. Afterwards, these quantities are mapped to commanded joint torques via the RNE inverse dynamics algorithm.

In a similar spirit, the approach by Team MIT (Kuindersma, Deits, et al., 2016) starts by specifying a desired trajectory computed with a simplified model. Afterwards, a time-varying linearization is done and a stabilizing controller is computed using LQR for time-varying linear systems. This computes optimal feedback and feedforward gains. But instead of using these gains directly, the computed optimal cost function is used as the cost function of a QP problem, which provides a value function approximation aspect and mitigates the instantaneous greediness inherent in QP formulations. As commonly done, the QP formulation allowes the incorporation of contact forces constraints, the full-body dynamics, and additional goals in the form of accelerations for parts of the desired motion not captured by the simplified model.

Finally, according to Team's WPI–CMU entry (Feng, Whitman, et al., 2015), they utilize a combined inverse kinematics and dynamics approach—both expressed as QPs—to better leverage the joint-level servos command structure. For the inverse dynamics part, the cost function corresponds to the weighted approach defined in (4.3b). Terms included are Cartesian space desired accelerations, a CoP tracking term, a weight distribution term for double support, a rate of torque term to preclude high frequency oscillations, and regularization terms for the problem's variables, which were the generalized accelerations, torques, and forces, per the torque-based WBC convention. Finally, constraints include friction cone and rectangular CoP limits.

4.2.2 Whole-body QP-based approach

Our implemented QP-based whole-body control approach roughly follows the formulation proposed in the last discussed work by Feng, Whitman, et al. (2015). The general formulation of our optimization framework is described as

$$\min_{\dot{v},\tau,F,s} \quad \sum_{j=1}^{N} w_j l_j(\dot{v},\tau,F,s) \qquad \text{Cost function} \tag{4.8a}$$

$$\text{s.t.} \quad M\dot{v} + H = S\tau + \sum_k J_k^T F_k \qquad \text{Lagragian dynamics} \tag{4.8b}$$

$$\mathrm{J}\dot{v} + \dot{\mathrm{J}}v = s \qquad \text{Cartesian acceleration of feet} \tag{4.8c}$$

$$F \in \mathcal{WC} \qquad \text{Contact wrenches constraints} \tag{4.8d}$$

$$|\tau| \leq \tau_U \qquad \text{Maximum torque limit} \tag{4.8e}$$

Specifically, the cost function (4.8a) includes N terms in total: spatial acceleration terms for hand end-effectors, torso, pelvis, and CoM (including linear and angular quantities), and regularization terms and rate regularization terms for the optimization variables.

The term (4.8c) defines the constraints for the Cartesian acceleration of the feet. Ideally, this should be set to zero during relevant support phases. But setting it equal to a slack variable and penalizing the slack variable's magnitude in the cost function empirically improves the numerical stability of the whole scheme and avoids infeasible solutions.

Constraint (4.8d) specifies the friction cone for the contact wrenches. In the above formulation, one wrench per end-effector of the limbs is specified, rather than four contact points at the edge of the feet support area. This slightly improves the problem's size, both in terms of variables and constraints. According to Caron, Pham, and Nakamura (2015), not only the friction cone and unilateral constraints are required, but also additional terms to avoid unwanted yaw rotation.

Finally, constraint (4.8e) bounds the computed joint torque within the maximum torque bounds specified by the robot's manufacturer.

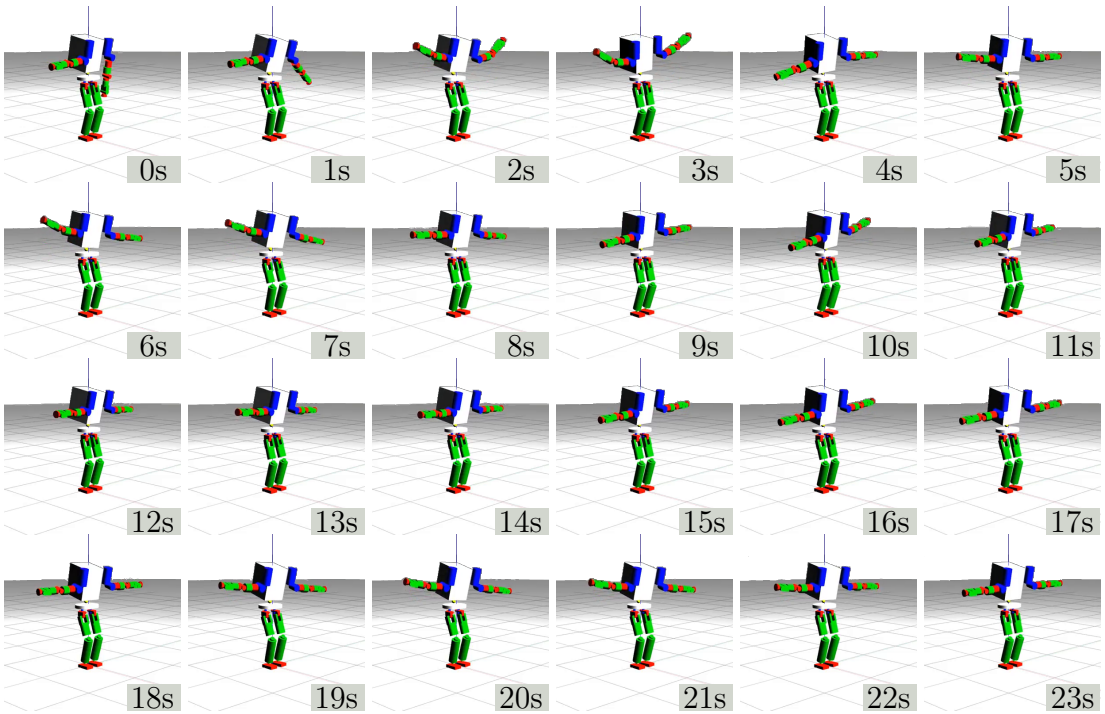


Figure 4.1: Time-lapse snapshots of the arm motion and gravity compensation tasks for the Atlas robot model.

4.3 Results

The above WBC formulation is applied to Atlas and Valkyrie (Radford et al., 2015) humanoid robots, mainly in the Gazebo simulator (Koenig and Howard, 2004). The same optimization-based formulation is applied for both robots (4.8); the only differences are found in the cost function term weighting.

4.3.1 Hand motion with gravity compensation for Atlas

The first task has the following desired goal: Compute joint torques that will cancel the gravity applied to the robot's structure while moving the left arm end-effector to a desired Cartesian pose. Spatial acceleration tasks for other links of interest (right arm, torso, *etc.*) should ideally be set to zero through a PD control scheme. Yet the cost function allows for uncontrollable violation, which is evident also from the resulting motion. Time-lapse snapshots of this computed motion are shown in Figure 4.1.

A video of the resulting motion is available at youtu.be/LZTXg7_K2Lc.

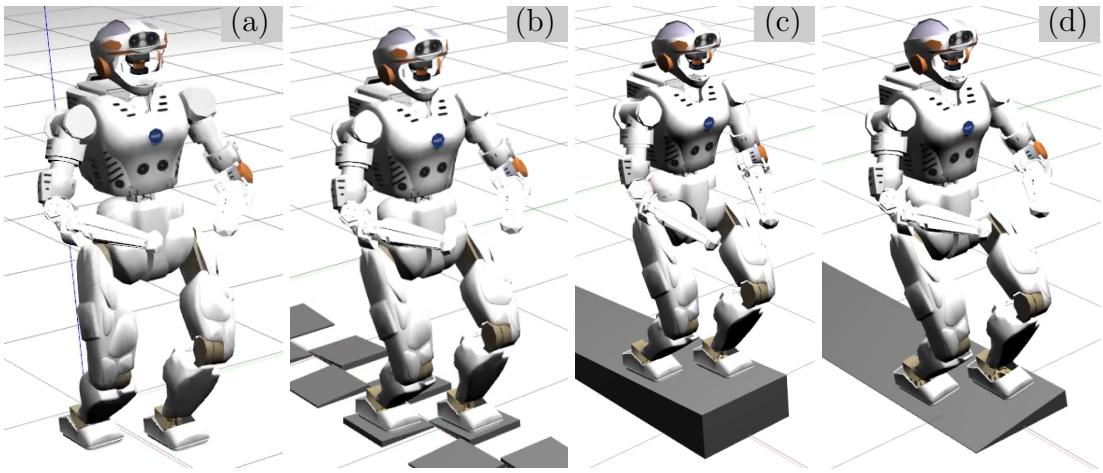


Figure 4.2: Snapshots of Valkyrie commanded by the automatically tuned WBC on different terrains: (a) Walking on flat terrain. (b) Discrete walking on slaps with 5° inclination and random orientation. (c) Walking on an inclined slab with a 10° rotation about the roll axis. (d) Walking on an inclined slab with a 5° rotation about the pitch axis. Adapted from (Yuan, Chatzinikolaidis, and Z. Li, 2019).

The resulting motion is distinctively oscillatory. This can be partially explained due to the gravity compensation term. It can become less oscillatory by introducing more damping for the D part of the spatial accelerations' PD gains. Yet the selection of these gains is complicated because many terms are involved and are in practice conflicting, as further elaborated in (4.3b). Next, a method that mitigates this issue is discussed.

4.3.2 Automatic gain tuning for Valkyrie

Bayesian optimization can be used to successfully tune the large number of parameters that are required for the WBC of high-dimensional robots (Yuan, Chatzinikolaidis, and Z. Li, 2019). This happens because Bayesian optimization does not require explicit knowledge of the function optimized, neither its gradients, and effectively treats it as a black-box function. It builds an approximation using Gaussian processes by selecting appropriate samples, in a sample-efficient manner. As a result, it can be readily applied to the WBC tuning problem.

This approach is applied to tune the cost function weights and the PD spatial

A video of the motions is available at ieeexplore.ieee.org/ielx7/7083369/8668830/8651385/18-1053_04_VI.mp4.

acceleration parameters for walking tasks across a number of non-level terrains in simulation. In summary, starting from a set of suboptimal parameter values, the optimizer iteratively improves their values to maximize a reward function. The reward function is designed in a way that encouraged better walking motions, including terms like CoM, foot, torso, pelvis, and hand motions.

Snapshots from computed motions are shown in Figure 4.2. The proposed approach successfully computes an improved set of weights and gains on terrains of varying difficulty. The computed gains are obtained for each terrain separately. Generalization across different terrains with the same set of gains requires further investigation.

4.4 Limitations

Since WBC approaches are a topic of intense and flourishing research, there are still a number of difficult challenges that need to be tackled. Addressing these issues will increase the robustness and facilitate wider adoption in a larger variety of real-world situations:

- *Lack of stability proofs and analysis*: Usually, a hierarchy of tasks is specified. The majority of the formulations based on the operational space approach guarantee exponential stability of the main task only, while the exact behaviour of the null space dynamics can not be determined easily.
- *External wrenches*: External wrenches are either not considered in the control law, leading to non-compliant behaviour, or compliance is achieved by means of measurements using force–torque sensors. Since the external loads are applied on the end effectors only, wrenches applied on other parts of the robot cannot be straightforwardly identified.
- *Objectives*: The objectives considered are usually conflicting. If the hierarchy is strict, then there is no straightforward way to choose the exact order, which changes depending on the situation. Alternatively, if relative weighting of the objectives is used, then choosing the appropriate gains and their update law is an equal complex problem without a clear solution yet.
- *Experimental validation*: Because these approaches are usually model-based, their extension in real-world situations is not straightforward. If the discrepan-

cies between the ideal and real responses are sufficiently large, that could lead to control degradation and very poor performance—or even instabilities.

Chapter 5

Contact-implicit trajectory optimization in task space

THIS chapter presents a motion planning framework that is capable to output through-contact motion plans in task space. To achieve this, Section 5.1 presents a TO framework that requires as input the desired task and the contact sequence, which implicitly defines the number of steps. Contact sequence means the pattern by which the end-effectors contact the ground. For example, for a humanoid robot, this typically is a double support phase, followed by a single support, followed by a double support and so on. For a quadruped robot, this can be a specific gait pattern, like a galloping gait. While the focus on Section 5.1 is on a single leg model—with a gait pattern of hopping, contact, hopping, and so forth—the same framework can be used in arbitrary legged configurations.

The task space model used corresponds to a single rigid body coupled with massless legs that are modelled through end-effector contact points. Incorporation of more detailed models, like full rigid body dynamics models or centroidal models, is discussed in Chapters 6 and 7. The necessary translation of the task space plan to joint space, where the robot is commanded, can be done in two ways:

- *Inverse kinematics*: Inverse kinematics can be used to translate the commands to joint space by ignoring all the information regarding contact forces. Such a motion planning pipeline is demonstrated in Section 5.1.2. In practice, being able to track such a motion for a hopping robot is quite improbable. For a more conservative and stable motion—in a humanoid, quadruped, *etc.*—this is

a viable approach, although the next option is more suitable.

- *Inverse dynamics*: For underactuated robots, inverse dynamics approaches have proven a more powerful alternative. Chapter 4 discusses this in more detail. Thus, a QP-based inverse dynamics control approach is much more appealing. Additionally, the motion plans with second-order accuracy outputted by the proposed framework can be coupled with the control approach proposed by Posa, Kuindersma, and Tedrake (2016). The authors underline that high-accuracy motion plans are a necessary prerequisite of their proposed framework.

Finally, in Section 5.2, an extension of the previous work is presented which does not require the contact sequence as an input by automatically computing it.

5.1 Specified contact sequence

Ground reference points (M. Popović, Goswami, and Herr, 2005), *e.g.* divergent component of motion (Englsberger, Ott, and Albu-Schäffer, 2015), instantaneous CP (Koolen, Boer, et al., 2012), and zero-moment point (Vukobratović and Borovak, 2004) are useful physical quantities for planning and control of discrete contact motions in legged locomotion. Such representations are intuitive and allow straightforward planning of references while being effective for generating stable motions (Yuan and Z. Li, 2018).

However, as the motions become more complex, *e.g.* involving contact points on non-coplanar surfaces or no contact points for a short period of time (in situations like running and hopping), the ground reference points—usually developed on a 2-dimensional projection basis—become less effective. As a result, they might require further extensions (the on-line adaptation presented by Zhou et al. (2017) is required to match the LIP model dynamics with that of the robot) or their assumptions might be invalidated. Such extensions can incur larger computational costs, while speed is one of the most attractive characteristics of these methods (Hu et al., 2018). Moreover, the projected quantities have reduced dimensionality, which can not fully represent the 6-dimensional information of either spatial

This section is published as: I. Chatzinikolaidis, T. Stouraitis, et al. (2018). "Nonlinear optimization using discrete variational mechanics for dynamic maneuvers of a 3D one-leg hopper". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 932–937. DOI: 10.1109/HUMANOIDS.2018.8624981

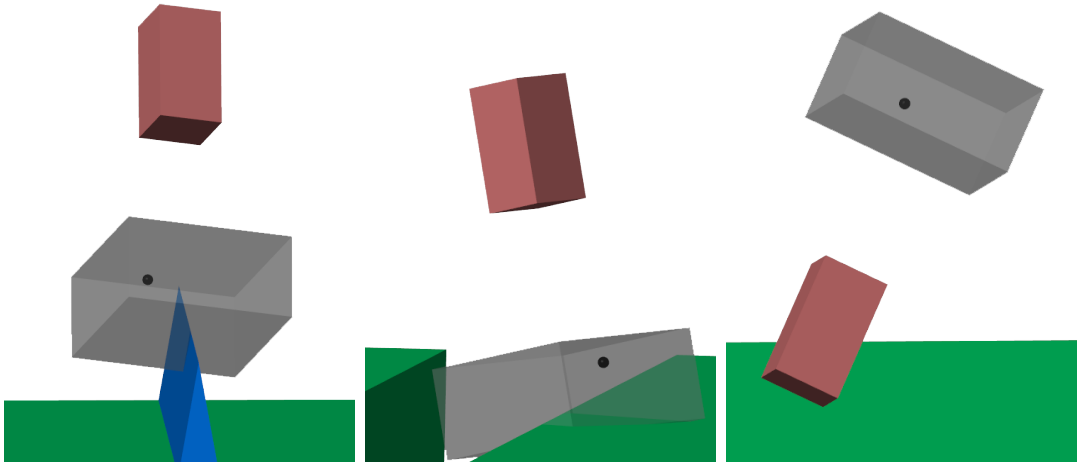


Figure 5.1: A variety of dynamic maneuvers using a unified optimization framework: (left) Jumping over an obstacle; (middle) Leaping over a gap; (right) Performing a somersault. We denote the body by a red box, the contact point by a black dot, and the workspace of the leg by the light-gray box.

motion or contact wrench.

Therefore, a more sensible approach is to directly compute ground reaction forces and the resulting physical motions. This is applicable not only in walking scenarios but also in running and hopping, non-periodic motions, *etc.* Equally important is that angular momentum is usually neglected or enforced to be zero; this heavily restricts the regime of possible motions. Examples of motions not easily planned with ground reference point methods are shown in Figure 5.1, computed by our optimization-based framework.

Still, reasoning first for the kinematic part and afterwards for the dynamics in a hierarchical fashion is used to obtain a variety of motions. Tonneau, Del Prete, et al. (2018) first find a root path using a sampling-based algorithm and then generate a discrete sequence of whole-body configurations. Due to the kinematic formulation, only static stability was enforced at the discrete configurations. More importantly, as with every hierarchical approach, it is not clear how solutions of the former stages restrict the solution of the latter stages of the pipeline.

As the models become more complicated non-convexity is introduced, which makes the problem hard to solve efficiently. Aceituno-Cabezas et al. (2018) try to absorb the discrete and non-convex aspects of the problem using a mixed-integer convex optimization formulation. The mixed-integer formulation is used to absorb the

multiple non-convexities present in the problem: related to gait (the footstep sequence, kinematic constraints, *etc.*) and the non-convexities of the terrain. Rather than computing individual motions for each leg, they use a predefined set of gait sequences. Furthermore, the convex segmentation of the terrain assumes that it can be approximated by a small number of convex polygons; otherwise, the number of variables increases dramatically.

A related line of research focuses on tackling the problem using continuous methods. The initial approach explicitly using this rationale is presented by Mordatch, Todorov, and Popović (2012). In their work everything is treated as part of an objective, *i.e.* all constraints are expressed as soft. Weighting the different constraints is tedious and requires much fine-tuning. Furthermore, the inclusion of soft constraints and forces by distance can result in non-plausible motions.

Another way to express the intermittent nature of contact is by using complementarity constraints as shown by Posa, Cantu, and Tedrake (2014). Complementarity problems are tackled with difficulty by continuous solvers due to the ill-posed nature of the constraints, so the authors resort to relaxations to make them more amenable. Another important issue that plagues continuous optimization approaches is the trade-off between problem size and accuracy. Manchester, Doshi, et al. (2019) proposes methods from discrete mechanics to make integration accuracy better, without increasing significantly the problem size.

The phase-based parametrization introduced by Winkler, Bellicoso, et al. (2018) allows the authors to avoid solving the complementarity problem explicitly, but with the trade-off of introducing the number of steps as parameter. Thus, the approach is less general than the complementarity formulation but computationally faster. But dynamic and kinematic constraints are enforced on regular intervals, which is not straightforward to select because more dynamic motions require finer computations. Also, quantities like force, limp position, and centroidal position and orientation are parametrized with a specific structure (*i.e.* splines), and as a result, only smooth motions and contacts can be represented.

We propose a constrained nonlinear optimization framework that calculates centroidal motion, limb motion, contact forces, contact timings and locations in a unified manner, given initial state, desired final state, and information about the environment. The dynamics of our system are derived using discrete variational

mechanics, with the associated geometric structure-preserving properties. The contributions of our work are the following:

- The problem is formulated as one unified optimization, rather than a hierarchy of cascade optimizations that limits the solution space.
- The use of discrete variational mechanics allows us to express the dynamics with a minimal representation while maintaining good numerical integration accuracy.
- Hard constraints are used to enforce physical plausibility while avoiding a piecewise parametric motion segmentation with splines. This allows us to compute a wider range of *dynamically feasible* motions because we do not enforce smoothness on the computed solutions and more accurately represent the discontinuity of contact phenomena.

5.1.1 Formulation of the problem

5.1.1.1 Modelling approach

We use a single rigid body that is able to describe the principles and include all the quantities associated with locomotion; translational and angular momentum, contact forces, body torques, orientation, *etc.* are well defined. At the same time, the complexity does not reach the levels of the full dynamics that a model of a humanoid robot typically exhibits, with a large number of degrees of freedom. Furthermore, current approaches in both planning and control prevalently work with the centroidal dynamic (Orin, Goswami, and S.-H. Lee, 2013), through the lens of which we can view the dynamic model as a single rigid body with mass equal to the total mass of the robot and configuration-varying inertia.

We augment the kinematic model of the rigid body with a contact point. The contact point (or limb) should always lie inside simplified kinematic limits, as shown in Figure 5.1, while its motion should exhibit continuity.

5.1.1.2 Problem formulation

Ideally, we would like to compute state and control trajectories which satisfy at every instant all the constraints that we would wish to impose—both nonlinear equality and inequality constraints. As this is currently impossible for general

formulations, we follow the paradigm of "first discretize then optimize" prevalent in the trajectory optimization literature (Betts, 2010). Specifically, we discretize both the state and the control. Hereafter, we refer to these discretization points as *knots*. In order to formulate the optimization problem, we strive in principle to express as many constraints as possible implicitly and not explicitly. That is, we try to include constraints in the formulation itself rather than explicitly enforcing them.

The problem can be described as:

$$\begin{aligned} \min_{x} \quad & l(x) \\ \text{s.t.} \quad & \dot{x} = f(x) \\ & h(x) = 0 \\ & g(x) \leq 0 \\ & x_L \leq x \leq x_U, \end{aligned} \tag{5.1}$$

which in our case is a non-convex optimization problem. The variables x of our optimization problem, denoted by capital letters, are the following:

(A) The global position $c \in \mathbb{R}^3$ and the change of orientation $\beta \in \mathbb{R}^3$ of the CoM.

(B) Time step $dt \in \mathbb{R}$ between two successive knots.

(C) The non-negative gains $\gamma \in \mathbb{R}^+$ for obtaining the vertex form of the linearized friction cone.

(D) The limb's position in the global frame $p \in \mathbb{R}^3$.

(E) The limb's velocity $\dot{p} \in \mathbb{R}^3$.

All constraints are denoted by small letters. The nonlinear equality constraints are the following:

(a) The dynamics of the CoM. An in-depth discussion about this constraint is given in Section 5.1.1.3.

(b) The initial translational velocity $\dot{c}_0 \in \mathbb{R}^3$ of the CoM and the initial angular velocity $\omega_0 \in \mathbb{R}^3$ in body coordinates. The initial position $c_0 \in \mathbb{R}^3$, and the initial unit quaternion $\alpha_0 \in \mathbb{H}, \|\alpha_0\| = 1$ for the orientation are implicitly enforced.

(c) The desired final position $c_f \in \mathbb{R}^3$ of the CoM. Final constraints for the rest of the quantities with initial values can be straightforwardly incorporated.

(d) We assume that we have access to a height map that describes the terrain's elevation. The contact point must be on the ground:

$$p^z = \text{heightmap}(p^x, p^y). \tag{5.2}$$

(e) The contact point does not contribute to the dynamics of the system. Thus, its kinematics are enforced through a forward Euler approximation:

$$p_{i+1} = p_i + \dot{p}_i dt_i. \tag{5.3}$$

(f) Unilateral and friction cone constraints for a contact force are enforced by the vertex form of the linearized friction cone (Kuindersma, Permenter, and Tedrake, 2014):

$$\mathrm{f} = \sum_{j=1}^{4} \gamma_j \left(\mathrm{f}^n + \mu \mathrm{f}^{t_j} \right), \tag{5.4}$$

where $\mathrm{f} \in \mathbb{R}^3$ is the generated contact force, and the extreme rays of the friction cone are obtained from the surface normal $\mathrm{f}^n \in \mathbb{R}^3$, the surface tangents $\mathrm{f}^{t_j} = \{\pm \mathrm{f}^{t_1}, \pm \mathrm{f}^{t_2}\} \in \mathbb{R}^3$, and the friction coefficient μ.

The nonlinear inequality constraints are:

(g) The contact point should lie above the height map:

$$p^z \geq \text{heightmap}(p^x, p^y). \tag{5.5}$$

(h) Kinematic box-type limits for the limb:

$$L \leq {}^b p - {}^b c \leq U, \tag{5.6}$$

where L, U represent the lower and upper box bounds of the relative position between the limb and the CoM, and ${}^b p$ and ${}^b c$ are the limb and CoM position in the body frame, respectively.

Finally, the following lower bounds are defined:

(i) The time duration between successive knots should be non-negative, $dt \geq 0$.

(j) The non-negative gains $\gamma \geq 0$. We also use an upper bound for these gains to bound the contact force.

We start by defining the number of contacts per limb and the number of knots per phase. By phase we mean a situation when the limb is either in rigid contact with the terrain or not (Winkler, Bellicoso, et al., 2018). Other phases in our formulation do not exist. The number of knots per phase is equivalent to the accuracy that we wish to achieve. The knots are divided into three sets:

- Knots where the limb is in contact with the terrain belong to the contact set.
- Knots where the limb is not in contact belong to the flight set.
- Knots between a flight and a contact phase belong to the landing set.

This division is inspired by the different states that can describe two contacting rigid bodies, as explained by Featherstone (2008).

There exist quantities that are defined in all knots, but also quantities defined in knots of certain sets only. Specifically, quantities (A) and (B) are defined in all knots, (C) are defined at knots belonging to the contact set, (D) are defined for knots in the flight and landing sets, and (E) are defined during flight phases only. Furthermore, we assume that the position is fixed during contact—that is zero limb velocity—and this is implicitly enforced.

These quantities are accompanied by constraints in order to enforce physical plausibility. As in the case of quantities, different constraints are enforced between different sets of knots. Equality constraints (a), (b), (c), inequality constraint (h) and bounds (i) are enforced for quantities defined in all knots. Equality (d) is defined during landing phases, while (e) during the flight phases. Finally, inequality (g) is defined during flight phases and bound (j) and equality (f) during contact phases.

5.1.1.3 Discrete variational mechanics

The main idea underlying discrete mechanics is to discretize the action and afterwards obtain the equations of motion for a mechanical system, rather than the "classical" approach of discretizing directly the equations of motion. The main advantage of this is that the obtained integrators automatically respect conservation of quantities like momentum and energy, and symplectic form, while exhibiting very good long term numerical behaviour. Here, only the necessary parts for our formulation will be presented while more information can be found

in (Marsden and West, 2001).

The starting point of the forced case is the discretization of the Lagrange–d'Alembert principle (Junge, Marsden, and Ober-Blöbaum, 2005) that seeks discrete curves $q|_{i=0}^{N}$, where $q_i \in \mathbb{Q}$ is a discrete configuration in the configuration space, satisfying

$$\delta \sum_{i=0}^{N-1} \mathcal{L}_d(q_i, q_{i+1}) + \sum_{i=0}^{N-1} \left(\mathrm{F}_i^- \cdot \delta q_i + \mathrm{F}_i^+ \cdot \delta q_{i+1} \right) = 0, \tag{5.7}$$

where $\mathcal{L}_d : \mathbb{Q} \times \mathbb{Q} \to \mathbb{R}$ is the discrete Lagrangian, and F_i^+, F_i^- are the right and left discrete forces, respectively, for all variations $\delta q_i|_{i=0}^{N}$ vanishing at the endpoints. This is equivalent to the forced discrete Euler–Lagrange equations

$$D_{q_i}\mathcal{L}_d(q_{i-1}, q_i) + D_{q_i}\mathcal{L}_d(q_i, q_{i+1}) + \mathrm{F}_{i-1}^+ + \mathrm{F}_i^- = 0. \tag{5.8}$$

In our case, the initial state is not two consecutive configurations but an initial configuration and an initial generalized velocity. We can use the left discrete Legendre transform $\mathbb{F}^-\mathcal{L}_d : \mathbb{Q} \times \mathbb{Q} \to T^*\mathbb{Q}$ obtaining

$$D_{\dot{q}_0}\mathcal{L}(q_0, \dot{q}_0) + D_{q_0}\mathcal{L}_d(q_0, q_1) + \mathrm{F}_0^- = 0, \tag{5.9}$$

where $\mathcal{L}$ is the system's continuous Lagrangian, from which we get the next state given our initial condition.

We now proceed to make the previous discussion more specific to our situation. The continuous Lagrangian $\mathcal{L} : TSE(3) \to \mathbb{R}$ of a single rigid body is

$$\mathcal{L} = \mathcal{T} - \mathcal{V} = \frac{1}{2} m \dot{c}^T \dot{c} + \frac{1}{2} \omega^T I \omega - \mathcal{V}, \tag{5.10}$$

where $\mathcal{T}$ is the kinetic energy, $\mathcal{V}$ is the potential energy, m is the mass of the rigid body, and I is the inertia matrix. The translational and rotational dynamics can be decomposed, and as a result, we study each part separately.

Translational dynamics We follow a similar derivation as presented in (Shen and Leok, 2017), the main differences being the inclusion of forces (*i.e.* the discretization of the Lagrange–d'Alembert principle rather than the principle of stationary action), and the use of non-constant time intervals due to the formulation of our optimization problem. In a similar fashion, we approximate the

translational velocity as $\dot{c} = \frac{c_{i+1}-c_i}{dt_i}$ and the relevant integrals using the midpoint rule approximation. The left and right discrete forces are approximated then as

$$\mathrm{F}_i^- = \mathrm{F}_i^+ = \frac{dt_i}{4}(\mathrm{f}_{i+1} + \mathrm{f}_i). \tag{5.11}$$

Thus, by virtue of (5.8) we have that

$$\frac{m}{dt_{i-1}}(c_i - c_{i-1}) + \frac{m}{dt_i}(c_i - c_{i+1}) - \frac{dt_{i-1} + dt_i}{2}\frac{\partial \mathcal{V}}{\partial c}(c_i) + \frac{dt_{i-1}}{4}(\mathrm{f}_{i-1} + \mathrm{f}_i) + \frac{dt_i}{4}(\mathrm{f}_i + \mathrm{f}_{i+1}) = 0, \tag{5.12}$$

while the initial condition is calculated by

$$m\dot{c}_0 + \frac{m}{dt_0}(c_0 - c_1) - \frac{dt_0}{2}\frac{\partial \mathcal{V}}{\partial c}(c_0) + \frac{dt_0}{4}(\mathrm{f}_0 + \mathrm{f}_1) = 0. \tag{5.13}$$

Rotational dynamics The orientation is parametrized using unit quaternions due to the small number of parameters (4 contrary to rotation matrices that require 9) and the lack of gimbal lock (as opposed to Euler angles). In order to avoid explicitly enforcing the unit norm for the quaternions using Lagrange multipliers, we formulate the problem using a variational integrator that preserves the Lie group structure of the unit quaternions (Manchester and Peck, 2016; Shen and Leok, 2017). Again, we use non-constant time steps which leads us to the constraint:

$$\frac{dt_{i+1}}{4}(\tau_i + \tau_{i+1}) + \frac{2}{dt_{i+1}}\left(\alpha^s_{i+1} I \alpha^v_{i+1} + \alpha^v_{i+1} \times I\alpha^v_{i+1}\right) = \frac{2}{dt_i}\left(\alpha^s_i I \alpha^v_i - \alpha^v_i \times I\alpha^v_i\right) + \frac{dt_i}{4}(\tau_{i-1} + \tau_i), \tag{5.14}$$

where $\alpha_i = \begin{bmatrix}\alpha^s_i & \alpha^v_i\end{bmatrix}^T$ is the unit quaternion of the relative orientation between knots i and $i+1$, and τ_i is the body torque at knot i. In (5.14), it is assumed that α is a unit quaternion. To implicitly enforce the unit norm, Manchester and Peck (2016) use the parametrization $\alpha = \begin{bmatrix}\sqrt{1-\phi^T\phi} & \phi\end{bmatrix}^T$, which holds for $\|\phi\| < 1$, which is not unconstrained and can lead the optimizer to compute complex values, while Shen and Leok (2017) use the exponential map, which has a singularity at zero. A more appropriate parametrization for our case is using the Cayley map, which is defined as:

$$\alpha = \begin{bmatrix}\frac{2}{1+\beta^T\beta} - 1 \\ \frac{2}{1+\beta^T\beta}\beta\end{bmatrix}, \tag{5.15}$$

Table 5.1: Parameters used in all simulation scenarios.

Type of rigid body	Cuboid
Dimensions (l_x, l_y, l_z)	$0.3 \times 0.3 \times 0.55$m
Mass (m)	80kg
Principal moments of inertia (I)	$2.6167, 2.6167, 1.2\text{kg} \cdot \text{m}^2$
Static friction coefficient (μ)	0.7
Height (h_b)	1.1m
Kinematic limits (l_x^k, l_y^k, l_z^k)	$0.6 \times 0.6 \times 0.2$m
Initial position (c_0)	$[0, -1.4, 1.1]$m
Initial velocity $(\dot{c}_0)$	$[0, 0, 0]^{\text{m}}/_{\text{s}}$
Initial orientation (α_0)	$[\sqrt{2}/2, 0, 0, \sqrt{2}/2]$
Initial angular velocity (ω_0)	$[0, 0, 0]^{\text{rad}}/_{\text{s}}$
Final position (c_f)	$[0, 0.9, 1.1]$m
Maximum constraint violation	$5 \cdot 10^{-5}$
Objective	0

where $\beta \in \mathbb{R}^3$. Finally, as with the translational dynamics, the constraint for the initial condition is defined as:

$$I\omega_0 + \frac{dt_0}{4}(\tau_0 + \tau_1) = \frac{2}{dt_0}\left(\alpha_0^s I \alpha_0^v + \alpha_0^v \times I\alpha_0^v\right). \tag{5.16}$$

5.1.2 Results

In order to demonstrate the results of our formulation, we will focus on 3 different situations: jumping over an obstacle, leaping over a gap, and performing a somersault. These are representative cases where classical approaches encounter large difficulties to generate dynamic motions. Table 5.1 summarizes the parameters that are used across all simulation scenarios.

We validated the feasibility of the produced plans by accurate numerical simulation with MATLAB (MathWorks Inc., 2018), using a fine step size for integration. MATLAB's non-linear optimization solver `fmincon` (IP algorithm) is used to solve

A video demonstrating the computed trajectory optimization motion results is available at ieeexplore.ieee.org/ielx7/8596719/8624912/8624981/0153_mm.zip.

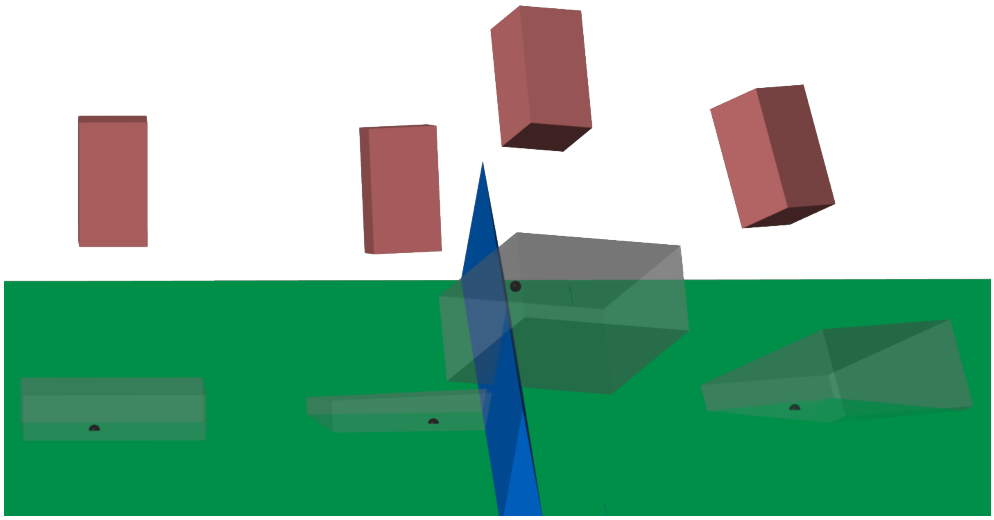


Figure 5.2: Time-lapse snapshots of the solution in Case 1.

the optimization problem. For constraints, for which analytic gradients are not available, automatic differentiation using CasADi is used (Andersson et al., 2018). All cases are executed on an `Intel(R) Xeon(R) CPU E3-1505M V6, 3.00 GHz` with `32GB RAM`.

The terrain and the obstacles are parameterized by B-splines (de Boor, 1978). B-splines provide a smooth and differentiable representation, suitable for non-linear optimization purposes. These were hand-tuned to provide two different environment models: A flat surface with an obstacle and a flat surface with a gap. A more complete approach, where the surface approximation is done using splines based on point cloud information, is provided by Pandey (2019).

Case 1: Jumping over an obstacle

In the first case, we place a triangular extrusion with a height equal to half of the model's height and span of 0.2m. Since the optimizer computes solutions at discrete knots, the interpolated motion afterwards might intersect the obstacle. In order to avoid such situations, we give a sketch of the desired solution by providing linear spaced positions of the CoM and limb as a starting point for the optimizer. We select 3 steps and 10 knots per phase.

Snapshots of the resulting motion are shown in Figure 5.2. As a quantitative measure of the difference between the optimizer's solution and that of MATLAB for

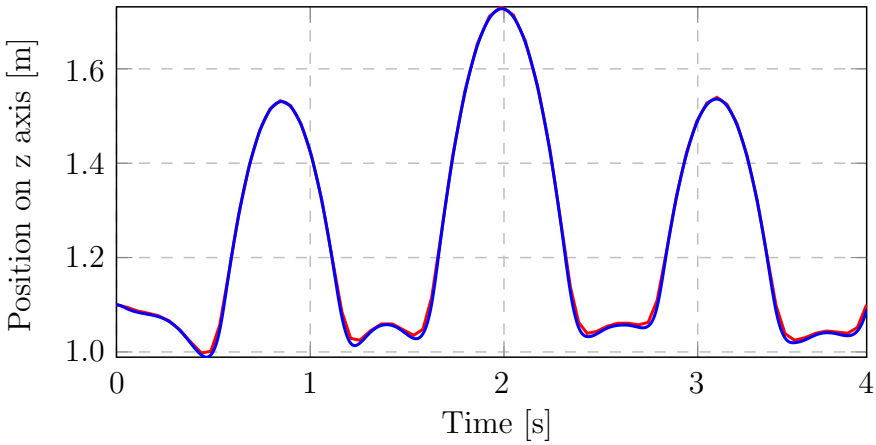


Figure 5.3: Centre of mass position in Case 1: In blue is the position along the z-axis as computed by MATLAB, while in red is the linear interpolated result of our optimizer.

the CoM position we use the RMSE, which is $[6.6216 \cdot 10^{-4}, 7.9697 \cdot 10^{-4}, 0.0066]$m. In Figure 5.3, we show the two outputs for the position in z-axis only, since the other two axes have very small errors that make the plots almost indistinguishable. To measure the difference between the computed quaternions (Huynh, 2009), we use the following metric

$$\theta = 2 \arccos(|\alpha_1 \cdot \alpha_2|), \tag{5.17}$$

with θ the angle required to get from one orientation to the other, and $\cdot$ the dot product between two quaternions. In this case, $\theta = 0.2941$rad is the maximum angle.

Case 2: Leaping over a gap

In this case, shown in Figure 5.4, we place the model at a terrain with 3m height. At the same time, we place a gap of 1m span from -0.5m to 0.5m along the y-axis.

Our scheme is only able to find local solutions. Thus, unless we model the gap in the formulation, it is very difficult to converge. Even though in the terrain representation we use a smooth surface interpolated using cubic splines, the gradients near the edges still change in an abrupt manner. Furthermore, for points inside the gap, the optimizer is unable to find solutions because very large forces

A video of the resulting motion is available at youtu.be/u9Sr9o0bLLw.

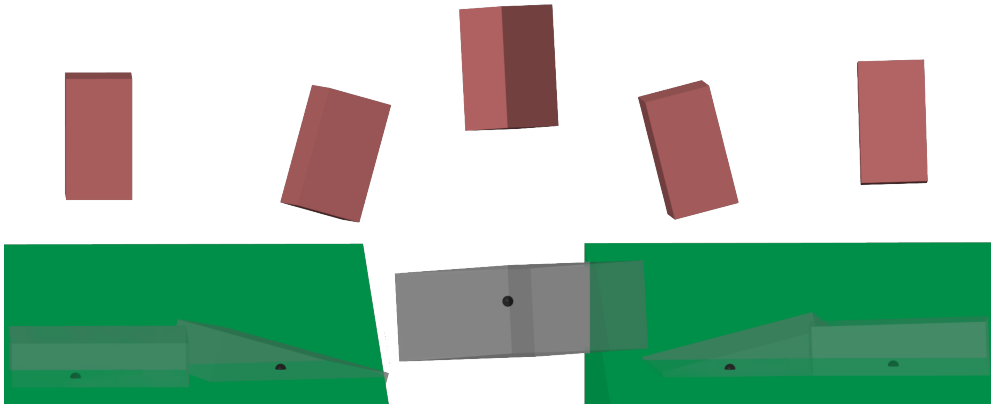


Figure 5.4: Time-lapse snapshots of the solution in Case 2.

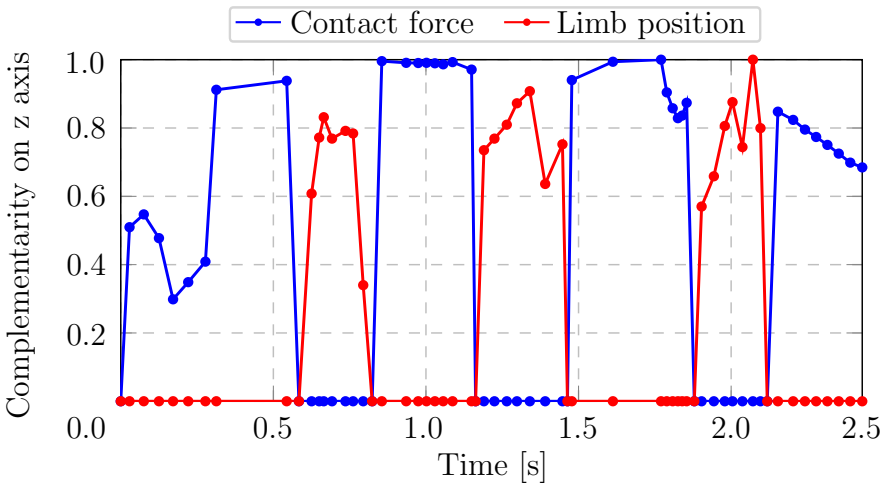


Figure 5.5: Complementarity condition in Case 2: Force only exists when the limb is in contact with the terrain, while the limb moves freely only when force is 0. All quantities are normalized by their maximum values.

would be required in order to escape from it. As a result, we select the desired flight phase where the jump takes place. Then we add a lower bound for the position of the limb at the last knot of the landing set before the jump and an upper bound for the first knot of the landing set after the jump.

Here we select 3 steps and 8 knots per phase. The RMSE for the position of the CoM is $[0.0042, 0.0121, 0.0129]$m, while the maximum angle is $\theta = 0.1633$rad. In order to stress out the complementarity inherent in the formulation, we show in Figure 5.5 the contact forces and limb positions in the same graph. Since we are comparing different physical quantities, we normalize each one with their maximum value to get a qualitative comparison.

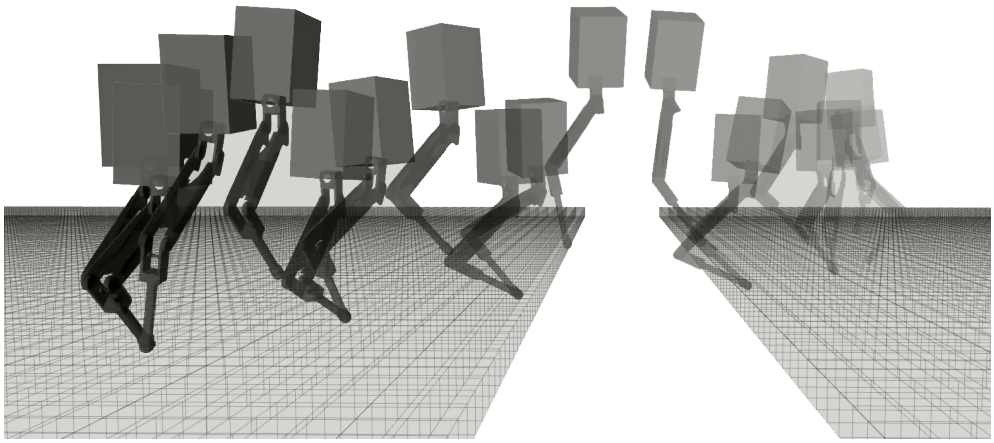


Figure 5.6: Time-lapse snapshots of the solution in Case 2 mapped a hopper model using inverse kinematics.

Further, we compute a second motion plan with the gap, where we add the additional task of reaching an opposite orientation along the z-axis at the end of the motion. We select now 5 steps with a similar number of knots per phase. With respect to the previous case, we use inverse kinematics to translate the task space motion to joint space. Snapshots of the computed motion are shown in Figure 5.6.

Case 3: Somersault

To show the modularity of the framework we include an orientation constraint. Specifically, we select an orientation of πrad with respect to the x-axis in the middle of the first flight phase. The situation is depicted in Figure 5.7. There are two potential ways to implement that: either via a suitable initialization or via an equality constraint. Here, we implement the second approach.

We select 4 steps and we use a larger number of knots per phase, *i.e.* 15, to get a sufficiently accurate approximation of the orientation. This shows why being able to choose the number of knots per phase is an important factor; different motions may require different approximations.

The translational part, due to the fine mesh, has a RMSE of $[1.4259 \cdot 10^{-4}, 2.4917 \cdot 10^{-4}, 0.0015]$m, while the rotational part has a maximum angle of $\theta = 0.3127$rad. The rotational part has a larger error in this case due to the relative large time steps for a midpoint rule approximation. This error can be seen in the quaternion components presented in Figure 5.8.

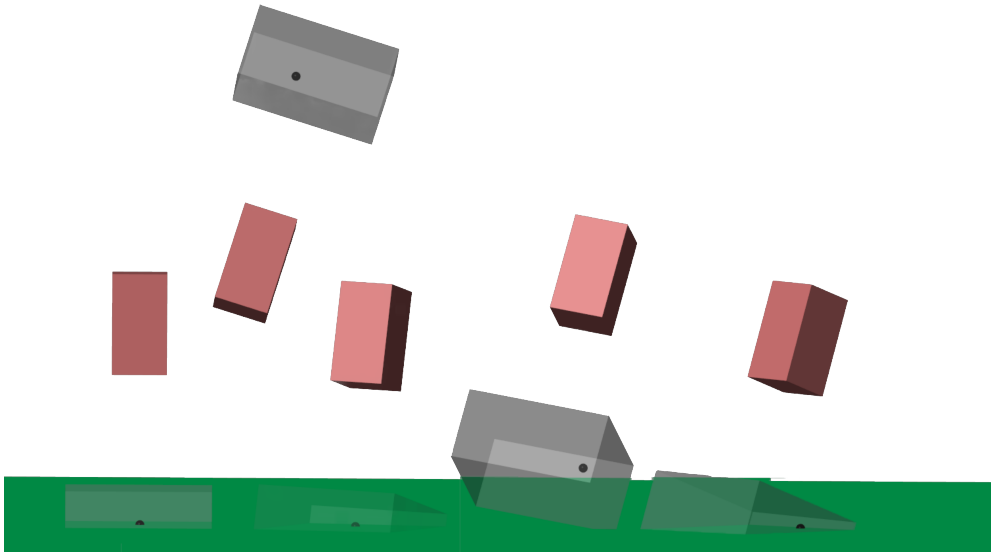


Figure 5.7: Time-lapse snapshots for part of the solution in Case 3. The desired final position of the CoM, as specified in Table 5.1, remains the same.

5.1.3 Conclusion

Our study focuses on a unified nonlinear optimization formulation that is capable of producing a wide range of dynamic motions in challenging scenarios as demonstrated by our simulation study. The scope of this work focuses on the formulation of feasible solutions. Thus, there are still a number of important questions to be answered as future work: initialization, computation time, objective function, and the extension to multiple limbs.

The optimization problem in this work is able to converge to local solutions only. As a result, seeding with appropriate initializations is very important in terms of the quality of the solutions and computational speed, while the initialization requires specific terrain information. Besides, the convergence speed depends also on the characteristics of the terrain. Computation time can vary from seconds to minutes depending on these two factors. A specific objective function is not yet given due to the additional computational cost, although it is necessary to quantitatively differentiate between feasible solutions.

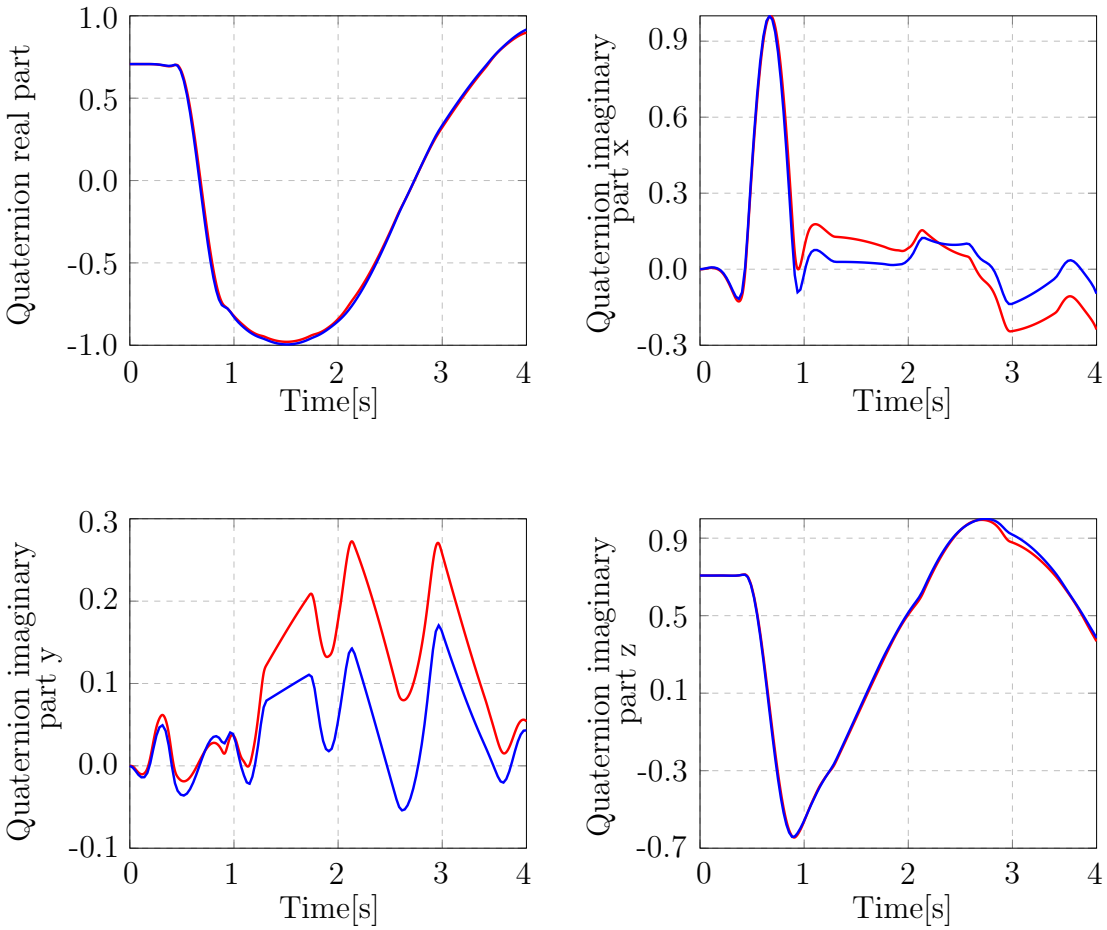


Figure 5.8: Orientation for Case 3: In red is the MATLAB's output as baseline, while in blue is the output of our method.

5.2 Unspecified contact sequence

For extending to an unspecified contact sequence, multiple approaches have been proposed. While a complete discussion can be found in Section 2.3, like the complementarity approach (Posa, Cantu, and Tedrake, 2014), a natural extension is the phase-based parameterization approach by Winkler, Bellicoso, et al. (2018). According to this parameterization, each potentially contacting end-effector assumes the fixed pattern of contact, non-contact, contact and so forth, with an individually optimized duration. This constitutes a general representation in the context of legged locomotion. Yet a number of motions are excluded, like slipping motions; Chapters 6 and 7 propose more general representations which include such motions. The question here is how to properly model the kinematic and dynamic constraints in accordance with the phase-based parametrization.

Winkler, Bellicoso, et al. (2018) propose the following solution to this problem: They interpolate the rigid body's and end-effectors motions and contact forces separately. For example, each limbs' position is parametrized using a cubic Hermite spline. Similarly, contact forces are parametrized using Hermite splines during contact phases. Then kinematic and dynamic constraints are enforced through outer sampling: The user specifies the kinematics and dynamics sampling frequencies, and these are enforced as constraints. This provides the benefit that the constraints remain sparse; for each sampled time, only a handful of variables affect the values of the relevant quantities. In order to make sure that the samples are drawn from valid timings, the total duration of the motion is pre-specified.

There are a number of drawbacks to this approach. First, the selection of all these quantities is not straightforward. The user is required to select the number of steps, the total duration of the motion, and sampling frequency of the constraints. Their selection is not decoupled from the resulting motion. Second, due to this decoupling, the quality of the computed motions is not guaranteed. For example, none or sparse sampling might happen during dynamic motion parts, leading to unphysical motions. Yet increasing the sampling frequency to mitigate this leads to a corresponding increase in the number of variables and constraints.

Here, the focus is on an alternative strategy that mitigates some of the issues. In summary, the samples are drawn from all the segments of the motion. Thus, constraints are enforced in all the motion's segments, improving the accuracy for dynamic motions. Further, by focusing the sampling on each segment, there is no need to pre-specify the total duration of the motion, and this is automatically computed by the optimizer. As a result, the user is relieved from selecting the parameters regarding the sampling frequency and total motion's duration. To achieve this, kinematic and dynamic constraints are treated slightly differently, as we explain next.

5.2.1 Kinematics constraints

Due to the task space model, kinematic constraints between the body and each end-effector are formulated independently. As a result, the body and each end-effector have their own separate discretization, both in the space and the time domain. A required constraint is that the total time duration of the body's and end-effectors' motions equal, *i.e.* the sum of all discrete time steps per timeline are

equal. Yet the total value is not pre-specified and is computed by the optimizer. We focus on the case of the body with one end-effector, but the same formulation is applied to all k body/end-effector pairs.

From the previous, each knot of the body's and end-effector's motion has its own timing. Thus, to enforce the kinematic constraints it is necessary to find a common timeline, *i.e.* similar time steps when the position of the body and the end-effector are sampled. Winkler, Bellicoso, et al. (2018) perform sampling based on an outer loop time step selection, as already discussed. Our selected approach is to perform sampling at the union of the two trajectories' knots. Thus, it is guaranteed that there is a constraint for each knot, as typically done in TO approaches, plus an additional set of sampled points. This means that a more dense constraint satisfaction is performed, albeit doubling the number of constraints with respect to classical TO approaches.

To perform this sampling, it is necessary to interpolate the knot values. Instead of cubic splines, we opt for a linear interpolation. This has the additional benefit that smoothness is not enforced to the solution—as cubic splines do. A piecewise linear function with N breakpoints $\sigma_1 < \sigma_2 < \ldots < \sigma_N$ can be expressed as

$$y(t) = a + bt + \sum_{i=1}^{N} c_i |t - \sigma_i|, \tag{5.18}$$

where $b = \frac{s_1 + s_N}{2}$, $c_i = \frac{s_{i+1} - s_i}{2}$, $\alpha = y(0) - \sum_{i=1}^{N} c_i |\sigma_i|$, and s_i denotes the slope of the i-th linear segment. If the slope is not available for the first and last breakpoint, it can be set to zero. This means that $b = 0$.

In practice, some user-controlled smoothness is required to avoid discontinuities between knots, which are not compatible with nonlinear optimization solvers. The source of discontinuity in the previous equation is the absolute value terms. There are a couple of options on how to smooth absolute values; some possible options are shown in Figure 5.9.

5.2.2 Dynamics constraints

For enforcing the dynamics constraints, the approach is very similar with the kinematics constraints case. Instead of the body's and end-effectors' positions, the focus now is on the force and torque on the body as a result of the contact forces by the end-effectors. In contrast to the kinematic case, the dynamics constraints are

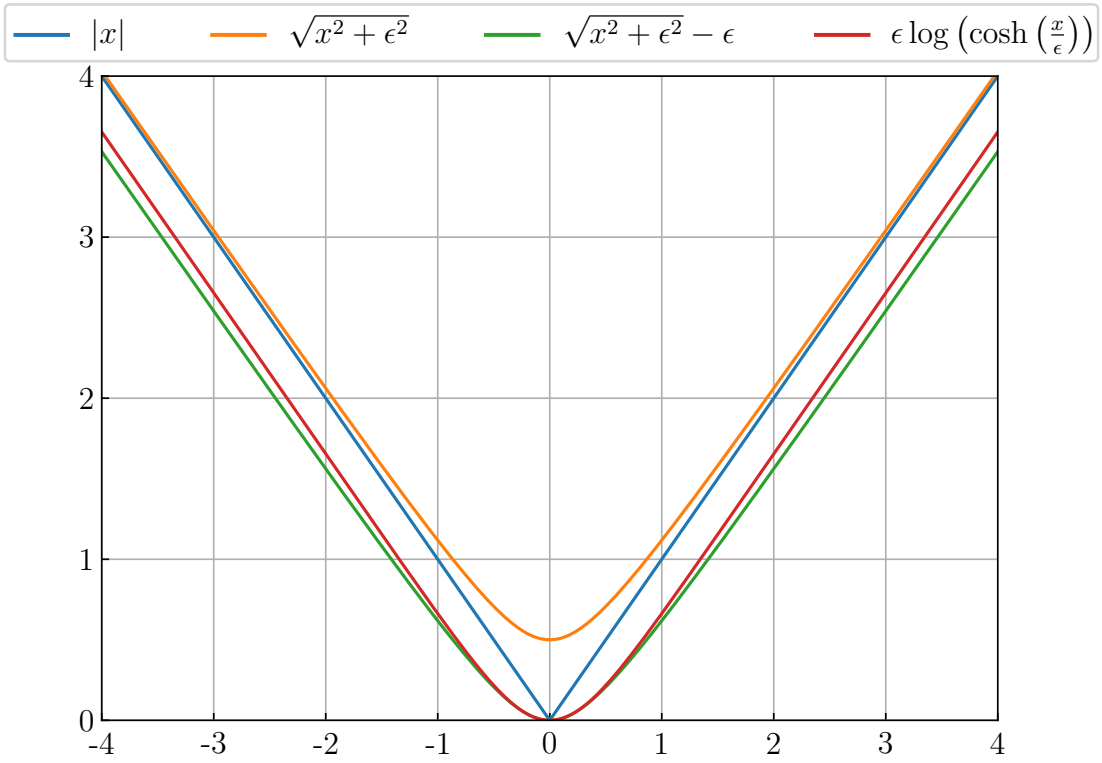


Figure 5.9: Smooth approximations of the absolute value function. Here $\epsilon = 0.5$.

formulated by taking into account all the force contributions from the end-effectors and it is not possible to separate them into pairs.

Thus, sampling is done from all contact forces at the same time instants. The equivalent wrench from each contact point at the body of the model is computed, and subsequently all contributions are added up to formulate the force balance for the body. Regarding the exact sample selection—since the limbs' are assumed massless and do not have dynamics—only the contact forces values at the body's knots are required. A piecewise linear interpolation is done again for the contact forces, as analysed before. It is worth noting that the second-order integration accuracy of the body's dynamics is still valid.

5.2.3 Results

We perform two simulation studies of the proposed method with unspecified contact sequences. The first one is concerned with motion generation for a bipedal robot model, while the second one for a quadrupedal robot model. After the computation of the optimal motion, the inverse kinematics approach is used to compute joint commands, as similarly done for the hopper model in the previous

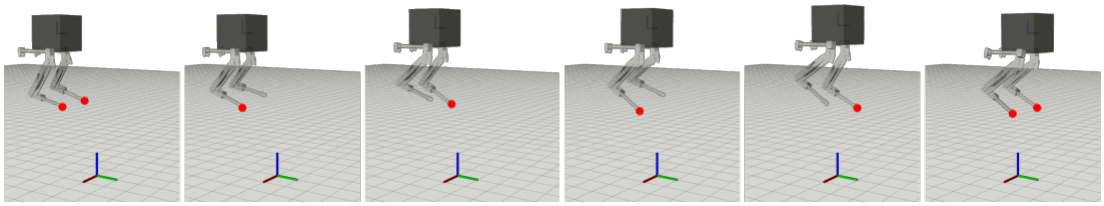


Figure 5.10: Snapshots of a walking motion for a bipedal model. The contact sequence is not pre-specified and is an outcome of optimization's solution.

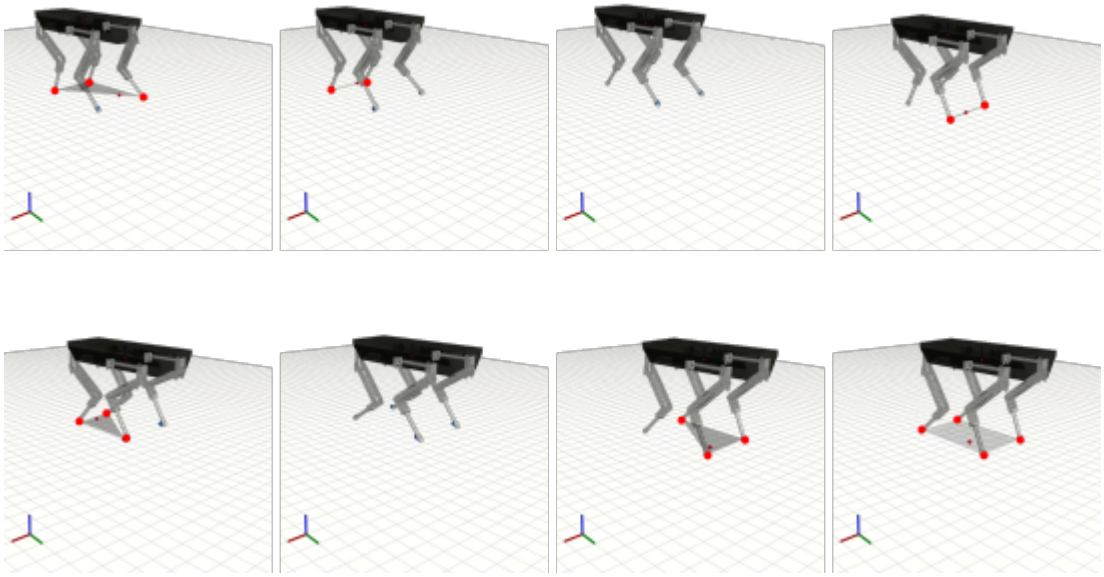


Figure 5.11: Snapshots of a galloping motion for a quadruped model. The support polygon and the ZMP are denoted with a grey area on the ground and a red point, respectively.

section. The visualization of the motions is done using the `Xpp` package (Winkler, 2017). Given the initial state, the task is to reach a specific Cartesian position with a specific joint configuration (nominal) with zero velocity. Snapshots of the resulting motions are shown in Figures 5.10 and 5.11.

The major drawback of the proposed approach is the lack of sparsity of the formulation. Both the kinematics and dynamics constraints depend on the variables that define the position and forces of the body and end-effectors throughout the trajectory. As a result, the sparsity property of the problem—a common assumption in large-scale nonlinear optimization solvers—is lost. Thus, this approach is not pursued any further. The next chapter presents a different approach that

is more general, computes directly joint-space quantities, and generalizes to an arbitrary number of leg configurations.

Chapter 6

Contact-implicit trajectory optimization in joint space

THIS chapter presents a novel contact-implicit TO method using an analytically solvable contact model to enable planning of interactions with hard, soft, and slippery environments. Specifically, we propose a novel contact model that can be computed in closed-form, satisfies friction cone constraints and can be embedded into direct TO frameworks without complementarity constraints. The closed-form solution decouples the computation of the contact forces from other actuation forces and this property is used to formulate a minimal direct optimization problem expressed with configuration variables only. Our simulation study demonstrates the advantages over the rigid contact model and a TO approach based on complementarity constraints. The proposed model enables physics-based optimization for a wide range of interactions with hard, slippery, and soft grounds in a unified manner, described by two parameters only. By computing trotting and jumping motions for a quadruped robot, the framework demonstrates the versatility for multi-contact motion planning on surfaces with different physical properties.

This chapter is published as: I. Chatzinikolaidis, Y. You, and Z. Li (2020). "Contact-implicit trajectory optimization using an analytically solvable contact model for locomotion on variable ground". In: *IEEE Robotics and Automation Letters* 5.4, pp. 6357–6364. DOI: 10.1109/LRA.2020.3010754

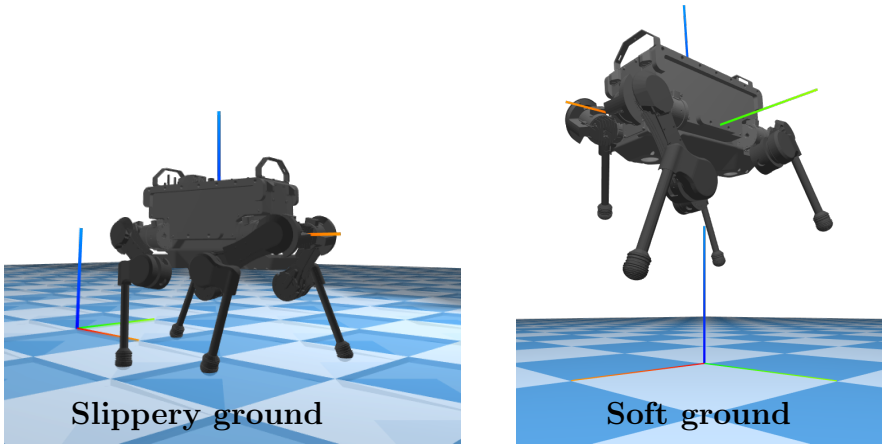


Figure 6.1: Dynamic motions computed by the proposed framework: trotting on slippery ground (left); jumping on soft ground (right).

6.1 Introduction

Physical interactions in humans, animals, and robots require models of the contact properties of the environment to plan movements. The necessary contact forces are generated due to intricate interactions between the contact media and are in practice difficult to model. Therefore, proper modelling of these interactions is important for a motion planning framework that aims to produce contact-rich behaviours on variable grounds, such as locomotion on soft floors.

Environments in our daily life exhibit many properties: they can be hard, soft, slippery, or combinations thereof, as shown in Figure 6.1. In terms of modelling, some of these aspects are usually missing in typical motion planning because most contact interactions are assumed rigid. For example, two usual approaches are using either spring-damper (Fahmi et al., 2020) or ad hoc penalization schemes (Carius et al., 2019). In this work, we present a novel contact model in a principled formalism that can capture such properties without the drawbacks of spring-damper models.

A common solution in complex multi-contact planning is to split the original problem into a series of stages, obtaining more tractable sub-problems that are still able to solve the original one. Examples are the work by Tonneau, Del Prete, et al. (2018) for general contact plans or the work by Carpentier and Mansard (2018) with a pre-specified contact sequence. Their main benefit is fast computation since each stage is usually designed to be efficiently solvable. However, it is challenging

to properly design these stages to compute general plans, without restricting the solution space or leading to infeasibilities for the subsequent stages. The focus here is on approaches that avoid such decompositions and can reason about the generated motion plans holistically.

TO has emerged as a powerful framework to design locally optimal trajectories for highly dynamical and underactuated systems (Betts, 2010; Kelly, 2017). One of its main benefits is that it allows the setting of high-level goals, expressed as cost functions while outputting a variety of motions as solutions. This is especially important for legged locomotion and multi-contact motion. Traditional approaches struggle to generalize across different scenarios or non-periodic motions, while TO methods are significantly more versatile (Neunert et al., 2017; Radulescu et al., 2017). Expressing complex multi-contact planning through a TO lens enables the solution of a wide range of problems with minimum modifications. Though learning approaches demonstrate similar generalization (Yang, Yuan, Heng, et al., 2020; Dallali et al., 2012), here we focus on the model-based optimization paradigm.

A significant problem in TO with contacts is proper modelling, as planning requires discontinuous and combinatorial reasoning. Thus, some approaches focus on embedding part of the problem in the TO description. This is possible for simple legged systems; for instance, one-leg hoppers (Chatzinikolaidis, Stouraitis, et al., 2018) and bipeds with a pre-defined periodic gait pattern like running (Mombaur, 2009). While this leads to problems with a very specific structure that are usually easier to solve, adapting to different legged configurations is difficult. This work focuses on approaches that work for arbitrary legged systems. This is possible for contact-implicit formulations that do not require a priori specification of the contact sequence.

An alternative approach is to describe the problem in a bilevel fashion: The outer level updates the state of the model, while the inner level computes the contact information (Yunt and Glocker, 2006; Erez and Todorov, 2012; Carius et al., 2018). Bilevel methods are usually solved by formulating the KKT conditions of the inner level, leading to a MPCC (Dempe and Dutta, 2012). Contrary to bilevel approaches, our work focuses on direct methods that formulate and solve the problem in a single level, such that the optimizer can reason about contacts by optimizing forces, with benefits for long-term physical reasoning (Toussaint,

Ha, and Driess, 2020).

Enabling the optimizer to directly reason about contact forces was proven very powerful for generating complicated contact-implicit motion plans. One way is to allow contact forces to act from a distance (Mordatch, Todorov, and Popović, 2012); while this is important for discovering contacts, penalizing these forces for physically realistic motion can be challenging. A more principled formalism is introduced in (Posa, Cantu, and Tedrake, 2014), where the problem is elegantly posed as an MPCC. This allows leveraging relaxations for this class of problems already studied in the optimization literature. Albeit these relaxations, a fundamental problem lies in the complementarity constraints, which usually violate constraint qualification tests (Betts, 2010). Some of these constraints are due to the contact model used.

Therefore, our principal motivation is to introduce a contact model that does not require the specification of complementarity constraints. Such an idea is discussed in (Drumwright and Shell, 2010), where they propose a pair of convex optimisation problems that compute the contact forces for simulation purposes. For direct TO, the previous work either focused on MPCC formulations or used a spring-damper model (Neunert et al., 2017; Önol, Long, and Padır, 2019).

In this work, we present a TO formulation with a contact model expressed as a pair of quadratic problems that can be computed in closed form. Thus, complementarity constraints are not required while problems associated with spring-damper models such as energy injections, stiff differential equations, and difficulties imposing the friction cone constraint are mitigated. Furthermore, our framework allows deriving the equations of motions for physical interaction with environments characterized by different stiffness, viscosity, and friction. By using the proposed framework, a variety of motion plans can be computed for hard, soft, and slippery surfaces by setting a small number of parameters.

6.1.1 Contributions

The major contributions of this chapter are summarized as follows:

- An analytically solvable contact model suitable for direct contact-implicit TO, which can be utilized in formulations without complementarity constraints, while satisfying unilaterality and friction cone constraints. The proposed contact

model is generic and can be used to compute motion plans on hard, soft, and slippery surfaces in a unified manner.

- A TO framework that integrates the new contact model for generating contact-implicit motion plans for a high DoF robot, demonstrating the advantages of the proposed method, with extensive comparisons performed against the rigid contact model and a TO formulation with complementarity constraints.

The remaining sections are organized as follows. In Section 6.2, the proposed contact model is derived and the overall direct TO formulation is elaborated. Section 6.3 presents the comparisons with (i) a different contact model, (ii) an alternative direct TO formulation based on complementarity constraints, (iii) followed by a variety of computed quadrupedal motions on terrains with different properties. Finally, we summarize and discuss future outlooks in Section 6.4.

6.2 Trajectory optimization formulation

6.2.1 Optimal control problem

The continuous OCP can be expressed as

$$\min_{q(t),\tau(t)} \quad l_N(q) + \int_0^N l(q,\tau)dt \tag{6.1a}$$

$$\text{s.t.} \quad M(q)\dot{v} + H(q,v) = S\tau + J^T(q)\mathrm{f} \tag{6.1b}$$

$$\mathrm{f} = \begin{cases} \arg\min_{\mathrm{f}} & k(q,v,\mathrm{f}) \\ \text{s.t.} & \mathrm{f} \in \mathbb{F}_\mu \end{cases} \tag{6.1c}$$

$$g\,(q,\tau) \in \mathbb{Z} \tag{6.1d}$$

$$q(0) = q^0 \qquad v(0) = v^0 \tag{6.1e}$$

$$t \in [0, N], \tag{6.1f}$$

where $l(q,\tau)$ is an additive cost, and $l_N(q)$ is the final state cost. These can be general sufficiently smooth functions but we focus on positive definite quadratic forms. Equation (6.1b) specifies the dynamics of the system, where $J(q)$ and f are the concatenated Jacobians and contact forces, respectively. The constraints (6.1d) specify general path constraints imposed on the optimal trajectory, *e.g.* joint and torque limits. Finally, (6.1e) and (6.1f) specify the initial state and time.

The optimization problem in (6.1c) specifies the contact forces via a mathematical program which makes (6.1) a bilevel optimization problem. The first approach is to provide gradients to the upper level via sensitivity analysis, but this can make long-term physical reasoning hard (Toussaint, Ha, and Driess, 2020). The second approach is to introduce the contact forces as variables, and then describe (6.1c) via its KKT conditions; that is, by imposing complementarity constraints. For example, the NCP contact model requires constraints for avoiding penetrations and the KKT conditions of the MDP (Manchester, Doshi, et al., 2019). A third approach is to solve (6.1c) for the contact forces, which are then not introduced as variables, and the associated complementarity constraints become unnecessary. We present such an approach next.

6.2.2 Contact model with analytical solution

First, the solution for the normal components is specified. We refer to it as the *frictionless case*. To obtain a unique solution, the strict non-penetration constraint is replaced with a quadratic program that has a unique solution and penalizes penetrations and the magnitude of the normal forces, while satisfying unilateral contact impulse constraints.

Given this solution, the tangential components are computed for what we call the *friction case*. Instead of the MDP, the velocities in contact space are minimized as in (Todorov, 2014). Since we focus on TO, the advantage of this approach is that an invertible contact model for the tangential components analogous to (3.19) can be formulated, and its unique minimum can be analytically derived.

6.2.2.1 Frictionless case

The following quadratic problem specifies each normal component as

$$\begin{aligned} \min_{\lambda_{n(i)}} \quad & \frac{1}{2r_{n(i)}}\lambda_{n(i)}^2 + \lambda_{n(i)}\phi_i^+(q). \\ \text{s.t.} \quad & \lambda_{n(i)} \geq 0. \end{aligned} \tag{6.2}$$

Since the next step gap distance is available, the velocity complementarity constraint becomes unnecessary. The solution is

$$\lambda_{n(i)} = r_{n(i)} \max\left\{-\phi_i^+(q), 0\right\}. \tag{6.3}$$

Notice that the model depends on information from the next time instant, in contrast to traditional spring models. By examining the solution, penetration

occurs when the contact impulses are activated and become positive; otherwise, they are zero. Furthermore, $r_{n(i)}$ expresses the trade-off between the magnitude of the normal impulse and the penetration depth. The larger $r_{n(i)}$, the smaller the penetration.

6.2.2.2 Friction case

The optimisation problem that minimises the velocities in the contact frame for the inverse dynamics case has the form

$$\begin{aligned} \min_{\lambda_{t(i)}} \quad & \frac{1}{2}\lambda_{t(i)}^T R_{t(i)}^{-1}\lambda_{t(i)} + \lambda_{t(i)}^T {\mathrm{v}^+}_{t(i)} \\ \text{s.t.} \quad & \|\lambda_{t(i)}\|^2 \leq \mu_i^2 \lambda_{n(i)}^2, \end{aligned} \tag{6.4}$$

where $R_{t(i)} = \text{diag}\left\{r_{t(i)}, r_{t(i)}\right\}$. Essentially, r_t trades off tangential velocity and force; the smaller its value, the more tangential forces are penalized.

This is a projection to circle problem and two cases can be identified. If the solution lies inside the cone, the problem is an unconstrained quadratic one, and the solution is

$$\lambda_{t(i)} = -R_{t(i)} {\mathrm{v}^+}_{t(i)}. \tag{6.5}$$

Otherwise, the solution lies on the boundary

$$\lambda_{t(i)} = -\mu_i \lambda_{n(i)} \hat{\mathrm{v}}^+_{t(i)}, \tag{6.6}$$

where $\hat{\mathrm{v}}^+_{t(i)} = \frac{{\mathrm{v}^+}_{t(i)}}{\|{\mathrm{v}^+}_{t(i)}\|}$.

For $r_{t(i)} \to 0$ the solution approaches the frictionless case, while for $r_{t(i)} \gg 0$ energy dissipation is increased. The tangential components of the impulse are given by (6.5) if $r_{t(i)}\|{\mathrm{v}^+}_{t(i)}\| \leq \mu_i \lambda_{n(i)}$; otherwise, they are given by (6.6). Finally, the tangential force is opposite to the tangential velocity; thus, the reaction force is dissipative.

6.2.2.3 Smoothing

In continuous optimization, smoothness of the objective and the constraints for at least the second derivative is required (Nocedal and Wright, 2006). From the previous analysis, it is clear that the computed impulses contain switches that can cause problems for the optimizer. Therefore, a procedure to remove

the discontinuities is discussed below. First, the solution for the *friction case* is expressed using a `max` function,

$$\lambda_{t(i)} = \hat{\mathrm{v}}^{+}_{t(i)} \max\left\{-\mu_i \lambda_{n(i)}, -r_{t(i)} \|\mathrm{v^+}_{t(i)}\|\right\}. \tag{6.7}$$

As a result, a smooth approximation to the `max` function is required, which is now present in the solutions for both cases. Multiple definitions for a smooth max function exist, and the selected `softmax` function is

$$\mathrm{smax}(\alpha, \beta; \epsilon) = \frac{\alpha + \beta + \sqrt{(\alpha - \beta)^2 + \epsilon^2}}{2}, \tag{6.8}$$

where for $\epsilon > 0 \to 0$ the approximation becomes stricter.

6.2.3 Direct transcription

According to the solutions (6.3) and (6.7), the contact impulses are described as a function of joint configurations and velocities, and the Jacobian can be used to map these quantities to the contact velocity in (6.7). Afterwards, these terms can be substituted in (6.1b), which becomes a function of the joint positions, velocities, and accelerations only.

Note that (6.1) is an infinite-dimensional continuous problem that can be transcribed to a finite discrete one (Betts, 2010). An implicit Euler discretization is selected due to its numerical properties, *e.g.* it is an A-stable method. Thus, problem (6.1) can be expressed as

$$\min_{q_i, v_i, \tau_i} \quad l_N(q_N) + dt \sum_{i=1}^{N-1} l_i(q_i, v_i, \tau_i) \tag{6.9a}$$

$$\text{s.t.} \quad M_{i+1}(v_{i+1} - v_i) = dt\left(S\tau_{i+1} - H_{i+1} + J_{i+1}^T \lambda(q_{i+1}, v_{i+1})\right) \tag{6.9b}$$

$$v_{i+1} dt = q_{i+1} - q_i \tag{6.9c}$$

$$g\left(q_i, \tau_i\right) \in \mathbb{Z} \tag{6.9d}$$

$$q_0 = q^0 \tag{6.9e}$$

$$v_0 = v^0 \tag{6.9e}$$

$$i \in [0, N-1]. \tag{6.9f}$$

Since (6.9c) are linear to the joint velocities, they can be removed from the optimization problem by substituting the right-hand side. Similarly, joint torques

are split into actuated τ and underactuated τ^u parts using (6.9b) (Erez and Todorov, 2012). Then τ is given as a function of the joint configurations and can be substituted directly on the objective (6.9a). Finally, the underactuated dynamics should be zero, yielding the overall trajectory optimization problem

$$\begin{aligned}
\min_{q_i} \quad & l_N(q_N) + dt \sum_{i=1}^{N-1} l_i\big(q_i, \tau(q_i, q_{i+1})\big) \\
\text{s.t.} \quad & M^u_{i+1}(q_{k+1} - 2q_i + q_{i-1}) = dt^2 \left(J^{uT}_{i+1}\lambda_{i+1} - H^u_{i+1}\right) \\
& g\left(q_i, \tau_i\right) \in \mathbb{Z} \\
& q_0 \;\; = q^0 \\
& q_{-1} = q^0 - dtu^0 \\
& i \in [0, N-1],
\end{aligned} \tag{6.10}$$

where $(\cdot)^u$ denotes the unactuated part of the quantity. It is worth noting that the Hessian and Jacobian remain sparse.

Finally, the MRP is selected (Terzakis, Lourakis, and Ait-Boudaoud, 2018) to represent the floating base—instead of *e.g.* Euler angles, rotation matrices or quaternions—for the following reasons. As with any three-parameter orientation parameterization, it possesses a singularity. For the MRP representation, this singularity is located after a full revolution. That places the singularity as far as possible from the origin. The polynomial expression, minimal number of parameters, and lack of unit norm constraints make this representation a suitable candidate for nonlinear optimization.

6.3 Results

Next, simulations are conducted to quantitatively validate the proposed formulation. The rigid body dynamics of the models are computed using the `RigidBodyDynamics.jl` library (Koolen and Deits, 2019). For all the cases, the optimization problems are formulated with CasADi (Andersson et al., 2018) for automatic differentiation, and solved by IPOPT, a large-scale IP solver (Wächter and Biegler, 2006). All results are obtained on an `Intel(R) Xeon(R) CPU E3-1505M V6, 3.00 GHz` with `32GB RAM`.

IPOPT allows the selection of a linear solver for computing the Newton steps; we used the `MA57` solver when performing comparisons and `MA86` otherwise (HSL,

Table 6.1: Parameters for the unactuated rigid body models.

Model	Position $[m]$	Orientation (MRP)	Body angular vel. $[rad/s]$	Body linear vel. $[m/s]$
Ball	$\begin{bmatrix} 0.1 \\ -0.75 \\ 0.3 \end{bmatrix}$	$\begin{bmatrix} -0.1617 \\ 0.566 \\ -0.0809 \end{bmatrix}$	$\begin{bmatrix} -0.372 \\ 1.208 \\ -0.834 \end{bmatrix}$	$\begin{bmatrix} -1.379 \\ -1.386 \\ -0.743 \end{bmatrix}$
Brick	$\begin{bmatrix} 0.1 \\ -0.75 \\ 1.7 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -1 \\ -0.2 \\ 0.126 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

2020). The difference is that `MA57` is deterministic but suitable for small to medium sized problems, while `MA86` is a parallel, non-deterministic solver suitable for large problems. In our experience, `MA86` performs better and is preferred when its non-deterministic property does not affect the comparisons.

As in all nonlinear optimization problems, proper scaling is important. Since configurations are the only variables, scaling here is straightforward and only the position of the floating joint requires special treatment. For the constraints and objective function, we use the default gradient-based scaling available in IPOPT.

First, the results for two basic models are studied before proceeding to a complex robot model: A rigid ball model that constitutes the simplest 3D floating model with one contact point and a rigid brick with eight contact points. These models are suitable for benchmarking and require minimum parameter tuning; the small number of states and the unique state solution provide a framework for direct comparisons. For benchmarking, we avoided more complicated models which can make the results less comparable due to the high-dimensional representations and non-trivial choices of parameters.

The initial state for all ball and brick simulations is summarized in Table 6.1, where the root body spatial velocity is defined in the body frame, while we use $\epsilon = 0.001$ for smoothing. Regarding the simulated motions, the ball and the brick were dropped on a flat ground. Parameter r_n is selected so that no bouncing occurs. Thus, for the frictionless case, the ball slides on the surface, while it rolls when friction is present. The dropped brick touches the plane with line contact. It slowly rotates until settling on a large side—where four vertices are active contact points[1].

[1] A video of all simulated cases is available at youtu.be/eLx1DebDHmY.

6.3.1 Comparison with physics simulation

To evaluate the contact impulses computed by the proposed approach, we performed a comparison with the rigid contact model which is typically used in simulation engines. The aim is to demonstrate that the model can represent different environmental interactions by an intuitive selection of its parameters. Assuming that the frictionless case corresponds to an extremely slippery interaction, our model can simulate very slippery conditions (numerically identical to the frictionless one), up until minimally slippery (numerically identical to the friction case).

For the primitive models, the optimization is equivalent to a root-finding problem. We perform comparisons against the nonlinear complementarity model with the PGS solver (Horak and Trinkle, 2019). This implementation uses a semi-implicit Euler integration scheme for the dynamics (so that the problem remains linear) while we use an implicit one. Furthermore, that trajectory is computed step by step since the computation is done in a simulation setting, whereas our TO formulation computes the whole trajectory simultaneously.

Figure 6.2 illustrates the position and orientation of the ball. First, the PGS solver was executed with and without friction and the resulting two solutions are plotted. Afterwards, $r_n = 100^{\mathrm{N}}/_{\mathrm{m}}$ is fixed for the normal component, and a parameter sweep is performed for r_t, with $\mu = 0.5$. It is verified that by changing the parameter r_t, we obtain the friction and frictionless solutions, as well as additional in-between solutions; these correspond to slipping motions if r_t is small, while they become more dissipative as r_t increases.

6.3.2 Comparison with a MPCC formulation

An approach most related to our proposed is the one presented in (Manchester, Doshi, et al., 2019), which is an extension of (Posa, Cantu, and Tedrake, 2014). Therefore, we perform numerical comparisons to understand their differences. The results are analyzed in terms of the number of iterations and solution time. Two comparisons are performed for each model, one in a frictionless setting and one with friction. The mean, standard deviation, and iterations are shown in Table 6.2, where each simulation is repeated n times.

For the comparison, the relaxation suitable for IP methods is utilized (Manchester,

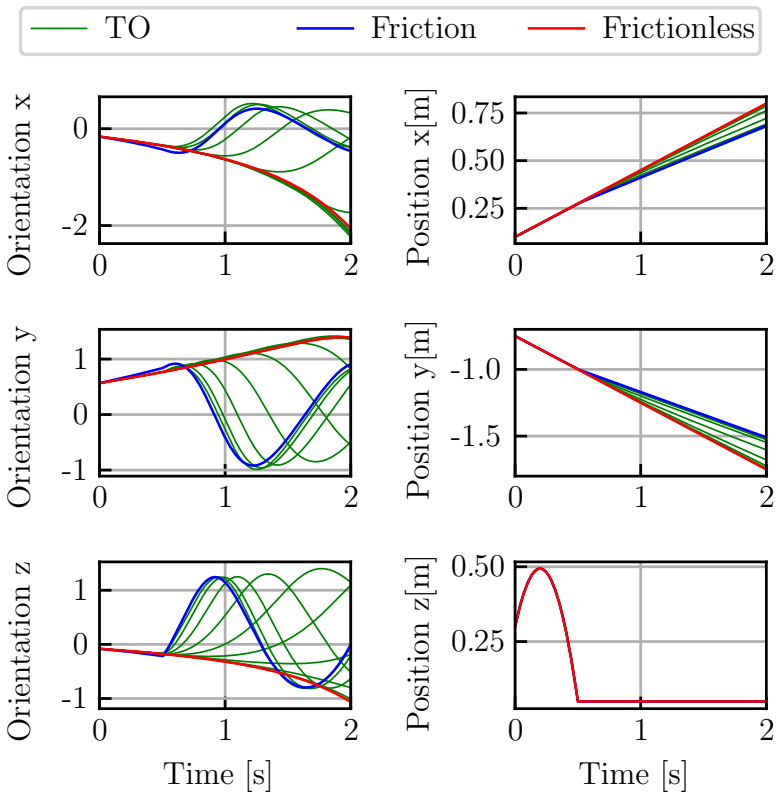


Figure 6.2: By changing the value of the parameter r_t, we can obtain a family of solutions that range from the frictionless to the dissipative friction case.

Doshi, et al., 2019). The dynamics for both methods are enforced using the inverse dynamics approach in Section 6.2.3. Regarding the slack variables weighing in (Manchester, Doshi, et al., 2019), $\alpha = 1$ is selected. The other aspects of the mathematical program with complementarity constraints remain intact.

6.3.2.1 Ball model

For both methods, the friction coefficient is $\mu = 0.5$, and our model parameters are chosen as $r_n = 100^{\mathrm{N}}/_{\mathrm{m}}$ and $r_t = 1^{\mathrm{N}}/_{\mathrm{m/s}}$ in the friction case, to match the response with the contact model used by the MPCC. We initialize both methods with zero variables, while we select a time step $dt = 0.1$s and final time $t_N = 2$s.

The RMSE for the frictionless case is 0.0218N (only normal force component), while for the friction case is $(0.0037, 0.007, 0.0218)$N; the ball's mass is 0.2kg. Note that tangential forces are generated only at the knot after the contact event and are zero otherwise. Also, the MPCC method uses a linearized friction cone, whereas we use the full cone model. The linearized version does not properly

Table 6.2: Running time and iterations of the MPCC versus our proposed formulation.

Method	MPCC		Proposed		
	Wall time $[s]$	Iter.	Wall time $[s]$	Iter.	n
Ball frictionless	0.773 ± 0.011	24	0.507 ± 0.016	15	5
Ball friction	3.78 ± 0.047	104	2.374 ± 0.023	64	5
Brick frictionless	1340 ± 15.27	6034	508.83 ± 0.56	1893	3
Brick friction	1151 ± 20.18	3015	317.68 ± 4.69	819	3

capture solutions that lie on the boundary (Section 3.4.2.2).

6.3.2.2 Brick model

We select $\mu = 0.6$, $r_n = 1000^{\text{N}}/_{\text{m}}$ and $r_t = 10^{\text{N}}/_{\text{m/s}}$ for our model's parameters. We initialize both methods with zero variables, with a time step $dt = 0.05$s for a horizon of $t_N = 3.5$s.

In the frictionless case, the number of variables for the MPCC formulation is 1540, while our formulation has 420. Both methods have the same number of equalities (420), while MPCC additionally includes 2240 inequalities. It is also very common to experience plateaus during the iterations with the MPCC approach, while we avoid such a problem by adapting the parameters. These plateaus are the main reason for the increased number of iterations in the brick frictionless case.

Similarly, the number of variables of the MPCC problem is 6580 for the case with friction, while for the proposed method is again 420. The MPCC problem has 2660 equality and 8960 inequality constraints, while ours has 420 equalities and zero inequalities. The computed normal contact impulses are shown in Figure 6.3, where the results show only the four activated contact points for clarity.

6.3.2.3 Overall remarks

Based on the comparison, there are two characteristics that are advantageous:

- The size of the optimization problem is kept minimum as we do not introduce extra variables (*e.g.* slack variables, Lagrange multipliers). This is important because general-purpose nonlinear solvers usually demonstrate higher than

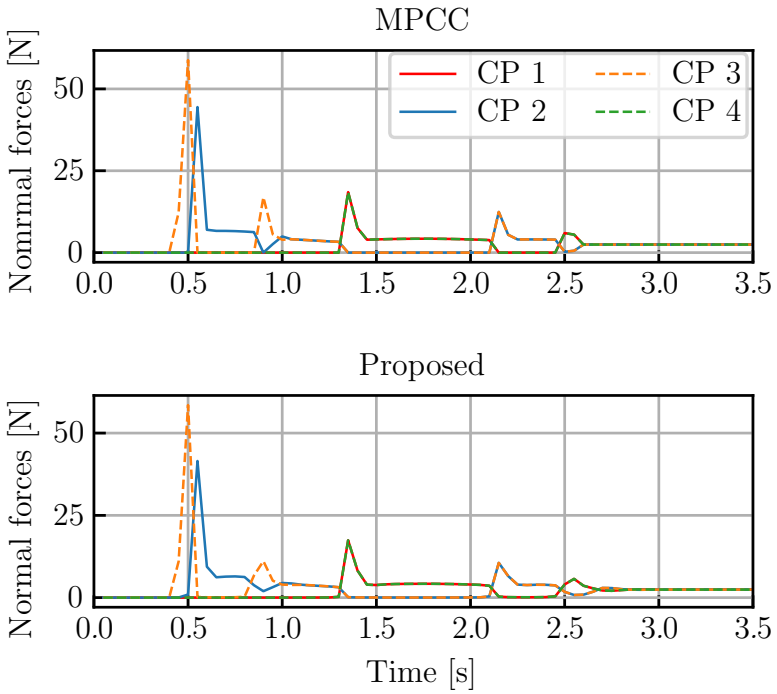


Figure 6.3: Comparison of the normal impulses at the four contact points (CP) of the brick with friction. The contact impulses at the other four inactive vertices are zero and not displayed. As expected, the differences between the two solutions are negligible confirming the validity of the comparison.

linear in time complexity since they do not utilize block factorization for solving the problem. Also, the optimizer's search space is smaller.

- The parameters in our proposed method are few and have direct physical meaning, which makes selection intuitive.

For the same problem instantiation, our method generally converges faster and requires a smaller number of steps. In our experience, choosing parameters that produce similar behaviours is not hard. For the MPCC method, there are no parameters to select and the performance is fixed for a specific problem instantiation.

Finally, our comparison did not include a cost function for reasons explained before. This is favourable for the MPCC formulation because selecting α in general can be challenging. For actuated systems with multiple solutions, α needs to be correctly tuned to drive the slack variables to values that sufficiently minimize complementarity violations, without affecting the optimized task. In our approach, there are no such parameters. Once appropriate values for r_n and r_t are selected

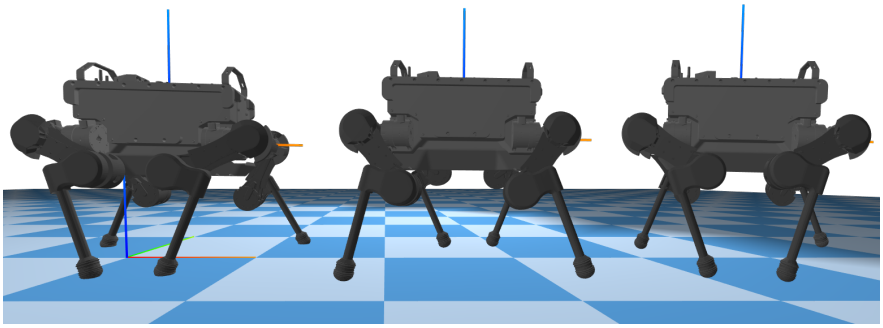


Figure 6.4: Trotting on hard ground snapshots (left to right).

to model the environment, they do not change between different tasks.

6.3.3 ANYmal trotting on hard and slippery surfaces

The proposed method is applied to the quadrupedal robot ANYmal (Hutter et al., 2016). We use similar gains as in (Neunert et al., 2017) to generate trotting gaits. The torque limits of the system (40Nm) are set as inequality constraints. Since the limits are provided, the torque penalization term in the cost function is decreased while the joint velocity penalization term is increased.

Furthermore, a step size of $dt = 0.08$s and a horizon of $t_N = 4$s is selected, resulting in a problem with 900 variables. By selecting this step size, we aim to demonstrate a positive aspect of our contact model: it is able to handle such large step sizes while not suffering from numerical stiffness. The step size is $\times 40$ bigger than the one used in (Neunert et al., 2017) and might not capture the maximum impulse. Here, we aim to compute a feasible motion plan which can be afterwards interpolated to generate reference trajectories for commanding such a motion on a robot. Since most robots have rubbers on the feet, a certain level of shock absorption not captured by the model is expected. Moreover, problems of short impulses are mitigated in case of locomotion on soft ground.

We initialize the optimizer with a nominal standing configuration for the whole duration; the same configuration is set as initial and desired final. This initialization is used only for the simulation on hard ground. Using this solution, we initialize the same optimization problem on a slippery surface. The purpose of this is to demonstrate the motion adaptations due to different environmental properties. Snapshots of the computed motions are shown in Figure 6.4.

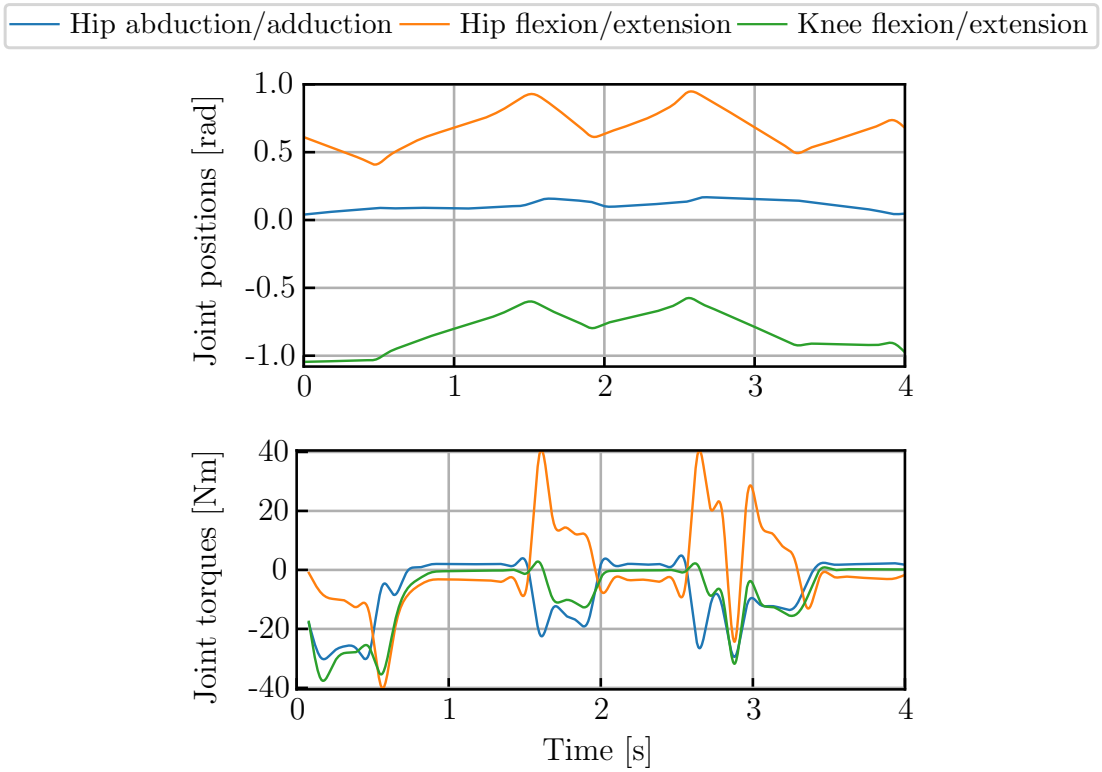


Figure 6.5: The joint position and torque (40Nm limit) trajectories of the left front leg (top) and left hind leg (bottom) for the trotting motion on hard ground.

6.3.3.1 Hard ground

For this case, we select parameters $r_n = 20^{\mathrm{N}}/_{\mathrm{m}}$ and $r_t = 20^{\mathrm{N}}/_{\mathrm{m/s}}$. The diagonal legs step together as expected in a trotting gait, while four steps are taken in total. The duration of each step is different, as shown in Figure 6.5; a possible reason is due to the requirement of stopping at the end of the motion. In this case, to start and finish the periodic gait on time, a transient state of fast stepping is necessary.

6.3.3.2 Slippery ground

To simulate a slippery ground, we select $r_t = 4^{\mathrm{N}}/_{\mathrm{m/s}}$. Compared to the previous solution, there are two notable differences. First, the ground clearance from the moving legs is significantly smaller. Since the slipping motions are abrupt, the optimizer tries to keep the contact points closer to the surface for fast activation. Second, the solution now relies more heavily on the hind legs to push the body forward, while the front legs are mostly used for stabilization. Similar results are reported in (Carius et al., 2019). Finally, a comparison between the computed solutions in the two cases for the motion along the x-axis is shown in Figure 6.6.

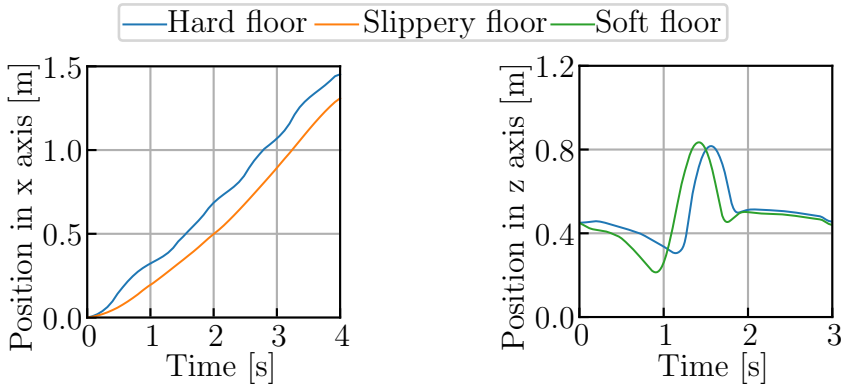


Figure 6.6: Body position in the x-axis during trotting on hard and slippery ground, and body position in the z-axis during jumping on hard and soft ground.

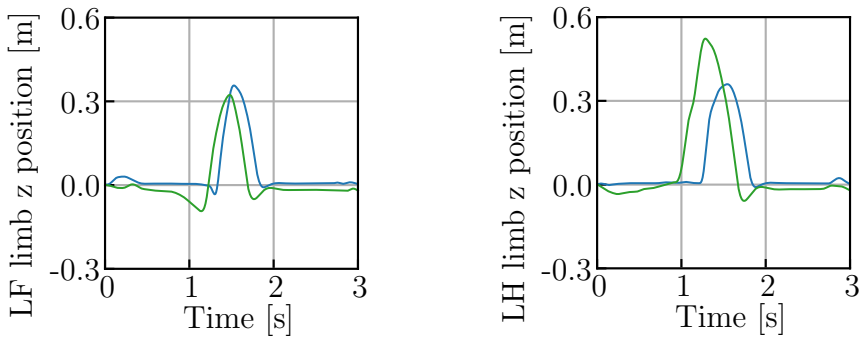


Figure 6.7: The foot height during jumping on hard and soft ground for the left front (LF) and left hind (LH) legs.

6.3.4 ANYmal jumping on hard and soft surfaces

Next, we compute a jumping motion using the ANYmal model on a hard and a soft ground. We use a waypoint at the middle of the trajectory for reaching a 0.8m height. As in the previous case, we specify only the initial and final state, and adapt the gains regarding torque and joint velocity penalization, without a maximum torque inequality constraint. This is because the specified motion is fast and with the current torque limitations the optimizer would struggle to find a solution that reaches the desired height.

We select a time horizon of $t_N = 3$s and a step size of $dt = 0.06$s, resulting in 900 variables. Again, we use the motion computed on the hard ground to initialize the jumping on the soft ground, aiming to demonstrate the motion adaptation. Unless specified, we use the same contact model parameters as in the trotting case. The position of the body in the z-axis for both cases is shown in Figure 6.6.

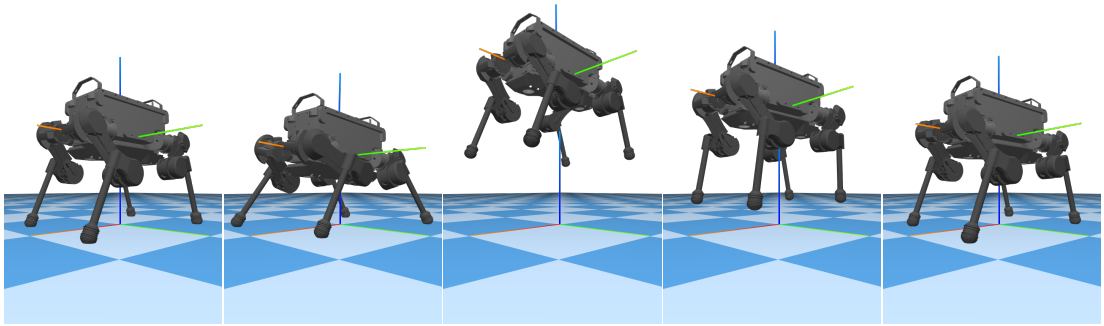


Figure 6.8: Optimized jumping motion on hard ground, at the same time instances as in the soft ground case.

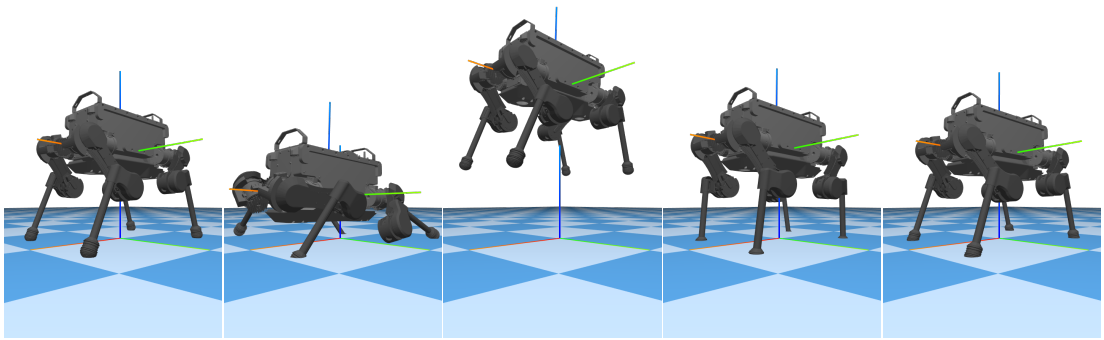


Figure 6.9: Optimized jumping motion on soft ground, where the feet penetrate the ground surface.

6.3.4.1 Hard ground

The snapshots from the computed solution are shown in Figure 6.8. We identify 6 phases: Lowering the body to prepare for lift-off, the lift-off, reaching the desired height and configuration, touch-down, and transitioning to the desired final state. Changing the torque penalization weight affects the apex height and time instant that this is reached.

6.3.4.2 Soft ground

The penetration of the limb is shown in Figure 6.7, while the snapshots of the motion are shown in Figure 6.9. The salient aspects of the motion are the same; differences are identified during the lift-off and touch-down phases. Specifically, during the lift-off preparation, the body is lowered in a similar manner but the feet penetrate deeper into the soft ground. Finally, a small oscillation occurs after touch-down due to the ground's softness and is quickly damped.

6.4 Conclusion

This chapter proposes a contact model suitable for direct TO formulations, followed by simulation validations with a wide range of models and settings: from simple objects (ball, brick) to a complex multi-body robot model in various locomotion modes (trotting, jumping) and ground conditions (hard, soft, and slippery). It is demonstrated that the proposed contact-implicit TO method can compute complicated motion plans for multi-contact interactions. An important feature in the new formulation is an improved, principled contact model which is solved in closed-form and expressed as a function of the state. Furthermore, this contact model avoids complementarity constraints for its description and automatically satisfies friction cone constraints, while not suffering from problems of energy injections and small step sizes. Moreover, it is described by two parameters that have intuitive physical interpretation and can be straightforwardly selected.

Nevertheless, there are several aspects worth of discussion and of future improvement. First, the parameters r_n and r_t need to be tuned for different robot models or new conditions in the environment's characteristics. This is a common property among soft contact models. Second, the presented method is suitable for solving contact-implicit planning problems in an offline setting, since the computation can not be performed in real-time yet. The computational time can be improved by taking into account the stage-wise nature of the decision problem. Lastly, a common feature of TO formulations is that the cost function needs to be specified for each task, *i.e.* the cost function is task-dependent.

The motion adaptations for various ground conditions demonstrate the importance of integrating environmental properties into motion planning. Future work will focus on systematic identification of parameters r_n and r_t and hardware validations. Another extension of the framework will focus on improving the numerical accuracy by using higher-order integration methods as in (Patel et al., 2019).

Chapter 7

Differential dynamic programming with contacts

THIS chapter presents a DDP approach for systems characterized by implicit dynamics using sensitivity analysis, such as those modelled via inverse dynamics, variational, and implicit integrators. It leads to a more general formulation of DDP, enabling the use of the faster RNE inverse dynamics that avoids the computation of the mass matrix. We leverage the implicit formulation for precise and exact contact modelling in DDP, where we focus on two contributions: (1) contact dynamics at the acceleration level; (2) formulation using an invertible contact model in the forward pass and a closed-form solution in the backward pass to improve the numerical resolution of contacts. The performance of the proposed framework is validated by comparing implicit versus explicit DDP for the swing-up of a double pendulum, and by planning motions for two tasks using a single leg model making multi-body contacts with the environment: standing up from ground, where a priori contact enumeration is challenging, and maintaining balance under an external perturbation.

This chapter is published as: I. Chatzinikolaidis and Z. Li (2021). "Trajectory optimization of contact-rich motions using implicit differential dynamic programming". In: *IEEE Robotics and Automation Letters*. Vol. 6. 2, pp. 2626–2633. DOI: 10.1109/LRA.2021.3061341

7.1 Introduction

The long-standing research goal of creating robots capable of physically interacting with our environment remains elusive. Typical tasks, such as moving around the environment (locomotion) and modifying the surroundings (manipulation), ultimately require a complex sequence of physical contacts between the robot and the external world. Achieving such capabilities requires effective solutions for producing contact-rich motions.

To date, we still have limited technologies to replicate animal- or human-level interaction skills on robots. This observation forces us to rethink the root of these limitations, which is more at an algorithmic and theoretical level rather than in hardware; it is nowadays possible yet difficult to validate a large range of physical capabilities in high-fidelity physics simulation. The scope here is on producing contact-rich motions for robot locomotion, leaving the applicability and adaptation on robot manipulation for future work.

TO has attracted increasing research interest for motion planning and control of highly dynamical, underactuated robots (Kelly, 2017). This is due to its potential of generating complex motions in a high-level manner: A user can design and specify a desired task using physical terms with associated weights via a cost function (Todorov, 2018), which can also be automatically tuned (Yuan, Chatzinikolaidis, and Z. Li, 2019). This approach is quite flexible, encompassing a wide-range of cases; for example, additional contact points can be included in the optimization to increase the robustness against perturbations for loco-manipulation tasks (Wolfslag et al., 2020).

This is particularly interesting for robotic systems that require through-contact motion plans, *i.e.* plans that involve multiple unspecified contact interactions. Physical contacts are traditionally difficult to model and incorporate in motion planning frameworks. Most approaches are multiphase, in the sense that contact schedule patterns (Chatzinikolaidis, Stouraitis, et al., 2018; J. Wang et al., 2020) or corresponding timings are provided a priori, while contacts are desirable with the end-effectors only. This leads to difficulties in practical implementations because the selection of locations and timings is in general non-trivial, while restricting contacts to end-effectors only limits the motion repertoire.

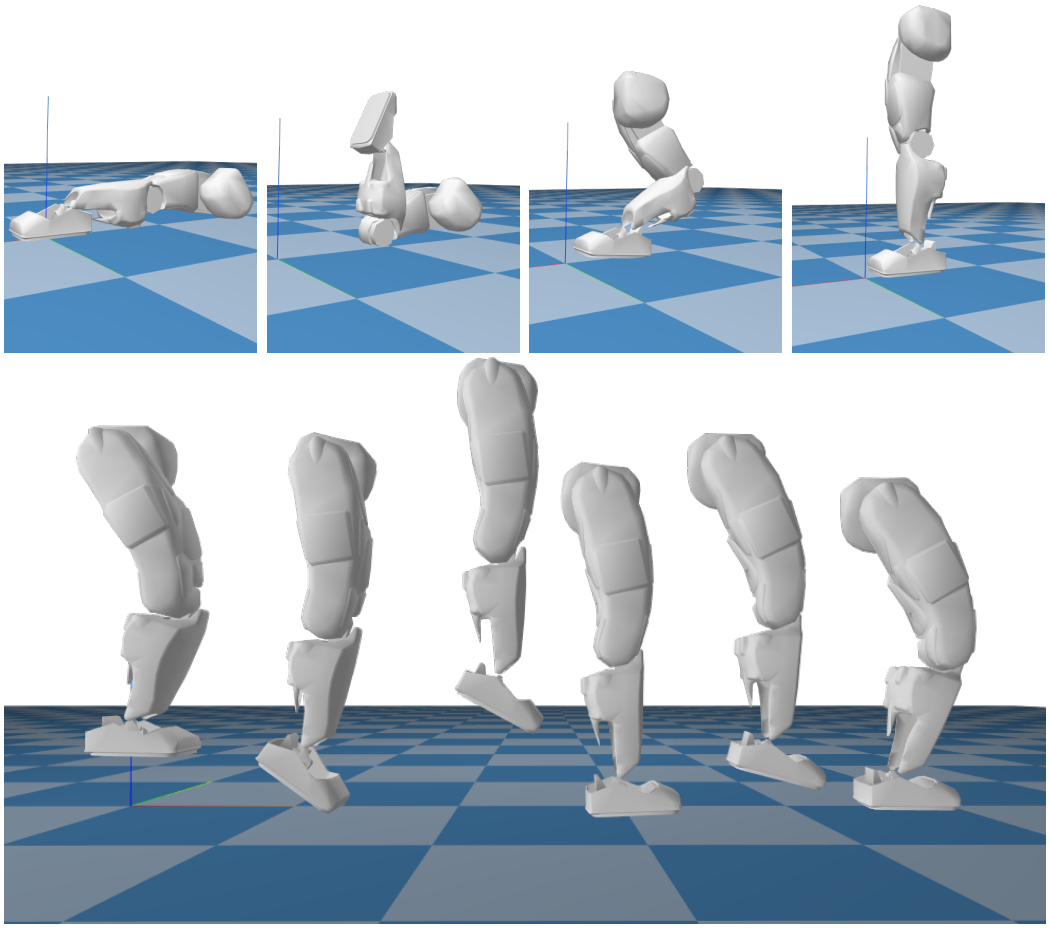


Figure 7.1: Complex multi-contact motions of a single-leg robot model computed by the proposed framework in time-lapsed snapshots: dynamic standing up from the ground (top), and balancing against an external perturbation (bottom).

DDP—a prominent shooting TO methodology—is among the most promising approaches for its efficiency in through-contact motion planning. This is demonstrated by a multitude of previous works that used DDP as backbone: From the simulated results (Tassa, Erez, and Todorov, 2012), to real-time applications for high-dimensional legged robots (Neunert et al., 2017; Carius et al., 2019).

However, properly modelling contacts is a considerable challenge; most DDP implementations resort to approximations and simplifications requiring well-tuned contact parameters. A fundamental reason is that contact phenomena are canonically described implicitly. This happens because decisions about whether a potential contact should be activated or not depends on information from the next time instant in a discrete-time setting. For example, a suitable contact force

should be selected at the current time step so that the contact at the next time step does not lead to an undesirable penetration.

The original DDP algorithm and its subsequent studies often assume that the discrete-time systems considered are explicitly defined. Thus, it relies only on the forward sensitivity of the state's evolution. Yet DDP can be readily applied to implicitly defined dynamical systems (Hairer and Wanner, 1996) when combined with appropriate sensitivity analysis. These are typically more challenging because they require the solution of nonlinear equations. However, they offer computational advantages, *e.g.* providing stability even for stiff differential equations. Further, handling implicitly defined systems allows more principled contact modelling in DDP.

7.1.1 Contributions

In this chapter, we provide *theoretical* and *algorithmic* contributions as:

- A sensitivity analysis approach for applying DDP to explicitly and implicitly defined systems in a unified manner.
- Based on this, we propose an approach leveraging an invertible model (Todorov, 2014) for exact contact resolution in DDP.
- Results demonstrating the possibility of exploiting properties of implicit integrators in DDP settings.

We benchmark our approach by applying it on implicitly and explicitly defined models, and on two cases of multi-contact whole-body motion planning for a planar single-leg robot that makes multi-body contacts: standing-up from ground and balancing from an initial perturbation in a receding horizon fashion (Figure 7.1). Our approach is equally applicable to models with large DoF and arbitrary contact configurations, such as using multiple legs.

The remaining sections are organized as follows. Section 7.2 discusses prior work on the DDP algorithm, and applications of DDP for through-contact motion planning. In Section 7.3, we present our application of DDP and, in Section 7.4, how to utilize it for through-contact planning. Section 7.5 provides comparisons between explicit and implicit systems in the context of DDP, and two motion planning studies for a single leg standing up and balancing in multi-contact settings. We

summarize and conclude in Section 7.6.

7.2 Prior work

7.2.1 Differential dynamic programming

DDP was originally introduced in (Mayne, 1966). Its main advantage with respect to the Dynamic Programming algorithm (Bellman and Dreyfus, 1962) is that it does not suffer from the curse of dimensionality by sacrificing global optimality. Subsequently, a number of improvements of DDP have been introduced. Recently, there was a resurgence of interest due to its potential for efficient planning for high-dimensional systems.

DDP is a second-order algorithm that exhibits quadratic convergence similar to Newton's methods (Liao and Shoemaker, 1992). Thus, it requires second-order information, which can be computationally challenging for high-dimensional models. To resolve this, the iLQR variant performs a Gauss-Newton approximation of the Hessian based on first-order information only, albeit with superlinear convergence (Todorov and W. Li, 2005).

The original DDP algorithm is concerned with unconstrained discrete dynamical systems only. Control bounds can be considered via a projected Newton QP solver (Tassa, Mansard, and Todorov, 2014). More general nonlinear inequality constraints via an active-set method (Xie, Liu, and Hauser, 2017). In robotics, it is common to consider multiple tasks in a hierarchical fashion, which is possible to do for DDP too (Geisert et al., 2017). In legged locomotion, the discontinuous nature of contact phenomena has led to the development of tailored approaches. For example, a pre-defined gait pattern and centroidal dynamics model was considered by Budhiraja et al. (2018), and more general hybrid systems by H. Li and Wensing (2020). We underline that the DDP framework presented next can incorporate the previous approaches straightforwardly.

Finally, a brief discussion about the application of DDP for implicitly defined systems from a Lie theoretic viewpoint is given by Boutselis and Theodorou (2020). Here, we present a more complete and deep treatment, with extensive comparisons. Furthermore, our vector-based formulation is much more familiar and common for robotic systems applications.

7.2.2 Through-contact motion planning

Applications of DDP for motion planning and control of legged robots have been very impressive. From simple, approximate models up to whole-body models, DDP provides a means for fast and even real-time solutions.

In (Tassa, Erez, and Todorov, 2012), DDP is used to control a humanoid model. A diverse set of behaviours is generated by simply changing weights in the cost function through a graphical user interface. An approximate solution for the contact dynamics is used, with a contact model similar to the one that is used here. The implicit formulation that we present next allows the consideration of contacts in DDP without requiring approximations to the contact model itself.

For quadruped robots, a diverse set of motions both in simulation and in hardware is shown by Neunert et al. (2017). To take into account contacts, a nonlinear spring-damper model was used. Even though tuning for each contact is done independently, spring-damper models can be difficult to tune in practice and require very small time steps. It is common for the optimizer to explore states where the current model parameters are not valid, while the small time steps translate into a large problem. Here, in the forward pass, the model takes into account all possible contacts in a centralized manner (through the coupling with the contact-space inertia matrix), while independently solve for each contact in the backward pass (by leveraging our implicit DDP formulation and the model's invertibility). Thus, performance is similar to complementarity formulations with large time steps, while we are capable to compute straightforwardly gradients in the backward pass.

To eliminate the unrealistic effects of spring-damper models, a hard contact model is used in (Carius et al., 2018). Unfortunately, contact impulses require the numerical solution of a QCQP, typical in time-stepping approaches with unilateral and friction cone constraints, and formulates the problem in a bilevel fashion. This complicates the derivative computation due to the numerical nature of the solution. We resolve this issue by leveraging the invertibility of the contact model: in the forward pass, the QCQP is solved with the associated constraints; in the backward pass, a closed-form computation is used that avoids the bilevel formulation. As a result, this does not pose issues with differentiation and leads to a faster and simpler implementation, without the need for backpropagation.

A multiple shooting variant is presented by Mastalli, Budhiraja, et al. (2020), extending the work by Giftthaler et al. (2018). It allows easier initialization since both state and control sequences can be used. Unfortunately, the intermediate iterates of the algorithm are infeasible, meaning that early stopping with a feasible trajectory, as in DDP, is not possible. This is a necessary property in our case since the through-contact motion planning approach that we present is running in a receding horizon fashion. Furthermore, the contact schedule is pre-defined by Mastalli, Budhiraja, et al. (2020), while here contacts are activated according to the natural dynamics of the system. Finally, friction cone constraints are neglected or can be taken into account through penalization in the cost function, which can be in practice difficult to tune and can lead to unrealistic solutions. Due to the imposition of contacts as equality constraints, attractive forces can arise at the solution, violating the unilateral constraint. Our framework here utilizes full unilateral and friction cone contact constraints without any approximation or penalization.

7.3 Implicit differential dynamic programming

Our point of departure from the original DDP algorithm is the dynamics in (3.4). Instead of the explicit dynamics, we assume dynamics of the form

$$f(x', x, u) = 0. \tag{7.1}$$

This will allow us to apply DDP for systems expressed via inverse dynamics, implicit or variational integrators, *etc.* Our focus will be contact dynamics, but we return to this later.

The goal is to compute the derivatives for the quadratic approximation of the Q-function (3.8). Terms related to the running cost l_i are trivial and will be omitted. Thus, we focus on the first and second-order sensitivity of the next step value function. A treatment of sensitivity analysis in the context of Newton methods can be found in (S. Zimmermann, Poranne, et al., 2019).

7.3.1 First-order sensitivity analysis

The first-order sensitivity of the value function in (3.6) is

$$V'_x = \frac{\partial V'}{\partial x} = \frac{\partial V'}{\partial x'}\frac{\partial x'}{\partial x} = V'_{x'}\frac{\partial x'}{\partial x}.$$

Here, V'_x is the sensitivity of the next step value function with respect to the current state, while $V'_{x'}$ is the sensitivity of the next step value function with respect to the next state; connected by the previous equation. Based on (7.1) we have

$$\frac{df}{dx} = f_{x'}\frac{\partial x'}{\partial x} + f_x = 0 \Rightarrow \frac{\partial x'}{\partial x} = -f_{x'}^{-1} f_x, \tag{7.2}$$

where it is assumed that for any x and u, x' can be computed so that (7.1) is satisfied. Combining the previous two equations gives

$$V'_x = -V'_{x'} f_{x'}^{-1} f_x.$$

In practice, a faster computation can be achieved using the adjoint method (Strang, 2007) by computing first the quantity

$$s^T = V'_{x'} f_{x'}^{-1} \Rightarrow V'^T_{x'} = f^T_{x'} s$$

and then

$$V'_x = -s^T f_x. \tag{7.3a}$$

If we confine ourselves in a first-order analysis only this is computationally advantageous (Strang, 2007), but the computation of $\frac{\partial x'}{\partial x}$ in (7.2) is required for the second-order expansion. By a similar reasoning, V'_u is computed as

$$V'_u = -s^T f_u, \tag{7.3b}$$

which concludes our first-order analysis.

We now have all the ingredients for the first-order approximation of the Q-function. For example, the Q_x term in (3.8) is given by

$$Q_x = l_x - s^T f_x.$$

7.3.2 Second-order sensitivity analysis

The second-order approximation of the value function is

$$V'_{xx} = \frac{\partial x'}{\partial x}^T V'_{x'x'} \frac{\partial x'}{\partial x} + V'_{x'} \frac{\partial^2 x'}{\partial x^2}.$$

The term $\frac{\partial^2 x'}{\partial x^2}$ constitutes a third-order tensor. We use matrix notation for the contractions but assume that their computation is clear from the context. It is computed as

$$\frac{d^2 f}{dx^2} = 0 \Rightarrow \frac{\partial^2 x'}{\partial x^2} = f_{x'}^{-1} \left(\frac{\partial x'}{\partial x}^T f_{x'x'} \frac{\partial x'}{\partial x} + \frac{\partial x'}{\partial x}^T f_{x'x} + f_{xx'} \frac{\partial x'}{\partial x} + f_{xx} \right).$$

By combining the last two equations we have that

$$V'_{xx} = \frac{\partial x'}{\partial x}^T V'_{x'x'} \frac{\partial x'}{\partial x} - s^T \left(\frac{\partial x'}{\partial x}^T f_{x'x'} \frac{\partial x'}{\partial x} + \frac{\partial x'}{\partial x}^T f_{x'x} + f_{xx'} \frac{\partial x'}{\partial x} + f_{xx} \right). \tag{7.4a}$$

For the remaining two terms in (3.8), a similar reasoning can be used to compute them as

$$V'_{xu} = \frac{\partial x'}{\partial x}^T V'_{x'x'} \frac{\partial x'}{\partial u} - s^T \left(\frac{\partial x'}{\partial x}^T f_{x'x'} \frac{\partial x'}{\partial u} + \frac{\partial x'}{\partial x}^T f_{x'u} + f_{xx'} \frac{\partial x'}{\partial u} + f_{xu} \right), \tag{7.4b}$$

$$V'_{uu} = \frac{\partial x'}{\partial u}^T V'_{x'x'} \frac{\partial x'}{\partial u} - s^T \left(\frac{\partial x'}{\partial u}^T f_{x'x'} \frac{\partial x'}{\partial u} + \frac{\partial x'}{\partial u}^T f_{x'u} + f_{ux'} \frac{\partial x'}{\partial u} + f_{uu} \right). \tag{7.4c}$$

This concludes the second-order sensitivity analysis. We can now compute all terms in (3.8). The rest of the DDP algorithm is implemented without changes.

7.3.3 Gauss-Newton approximation

Especially for robot models with many degrees of freedom, computing the tensor terms (7.4) can be prohibitively expensive. Fortunately, it is possible to do a Gauss-Newton approximation of the Hessian—equivalent to iLQR—by ignoring them. Thus, the second-order sensitivity terms of the value function in an iLQR setting become

$$V'_{xx} = \frac{\partial x'}{\partial x}^T V'_{x'x'} \frac{\partial x'}{\partial x} \tag{7.5a}$$

$$V'_{xu} = \frac{\partial x'}{\partial x}^T V'_{x'x'} \frac{\partial x'}{\partial u} \tag{7.5b}$$

$$V'_{uu} = \frac{\partial x'}{\partial u}^T V'_{x'x'} \frac{\partial x'}{\partial u}. \tag{7.5c}$$

7.4 Acceleration-level contact dynamics

We describe here a contact resolution framework at the acceleration level rather than the commonly used velocity level. This way, we avoid the necessary first-order discretization of the dynamics. Other assumptions are not required about the robot's model (such as the assumption about a constant Jacobian in (3.13) that is inherent in the velocity-impulse formulations), without increasing the computation complexity. As such, we consider it a superior choice. It is also the default choice in MuJoCo (Roboti LLC, 2020), which is a state-of-the-art robotics simulator.

Starting from the continuous time dynamics (3.12), we multiply both sides by JM^{-1} and add $\dot{J}v$, which gives

$$\underbrace{J\dot{v} + \dot{J}v}_{\alpha^+} = \underbrace{JM^{-1}J^T}_{A}\mathrm{f} + \underbrace{JM^{-1}(S\tau - H) + \dot{J}v}_{\alpha^-}. \tag{7.6}$$

We can interpret this equation as follows: α^- is the unconstrained acceleration in contact space in the absence of any contacts, which is corrected by the term $A\mathrm{f}$ to result in the actual acceleration α^+ that satisfies the contact constraints.

As already explained, the contact model that we utilize was proposed by Todorov (2014). It computes the necessary contact forces by solving the following convex optimization problem that tries to minimize accelerations in contact space

$$\begin{aligned} \min_{\mathrm{f}} \quad & \frac{1}{2}\mathrm{f}^T(A+R)\mathrm{f} + \mathrm{f}^T(\alpha^- - \alpha^*) \\ \text{s.t.} \quad & \mathrm{f} \in \left\{\mathrm{f} \mid \mathrm{f}^{n(k)} \geq 0, \|\mathrm{f}^{t(k)}\| \leq \mu_k \mathrm{f}^{n(k)}, \forall k\right\}, \end{aligned} \tag{7.7}$$

which is the equivalent to (3.17) for accelerations.

While the bias accelerations α^* can be in a general Baumgarte stabilization form (3.18), a choice that works reasonably good across models is

$$\alpha^* = \dot{J}v - \frac{1}{dt^2}\phi(q) - \frac{1}{dt}Jv, \tag{7.8}$$

with $\phi(q)$ the gap distance, positive when bodies are separate. The first term is used to cancel the same term in α^+ and α^- and simplify computations. The second and third terms are obtained by a Taylor expansion of the gap distance function and ignoring third and higher-order terms.

In the forward pass, the above optimization problem is solved for the contact forces using a standard PGS solver (Horak and Trinkle, 2019). Though in principle this can be implemented in the backward pass, the computation of the gradients becomes more complicated since we have to differentiate a numeric solution. Even with automatic differentiation, the quality of the gradients can suffer. Instead, a diagonal approximation of the system is assumed and an approximate solution to the contact forces is computed (Tassa, Erez, and Todorov, 2012). The implicit formulation avoids this issue and the exact solution for the contact forces is given in a closed form.

By utilizing the implicit DDP framework and the invertibility of the model,

Algorithm 2: Forward pass with contacts.

Input: x, κ_i, K_i, R, and $\mu_k, \forall k$.

Output: x' and f.

1 Compute $A + R$ and $\alpha^- - \alpha^*$ based on (7.6) and (7.8).

2 Solve (7.7) for the contact forces f.

3 Solve (7.1) together with (3.11b) for the next state x'.

problem (3.19) is expressed in acceleration space

$$\begin{aligned} \min_{\mathrm{f}} \quad & \frac{1}{2}\mathrm{f}^T R\mathrm{f} + \mathrm{f}^T(\alpha^+ - \alpha^*) \\ \text{s.t.} \quad & \mathrm{f} \in \left\{\mathrm{f} \mid \mathrm{f}^{n(k)} \geq 0, \|\mathrm{f}^{t(k)}\| \leq \mu_k \mathrm{f}^{n(k)}, \forall k\right\}. \end{aligned} \tag{7.9}$$

For the computation of α^+ as given by (7.6), the joint acceleration $\dot{v}$ is required. In the classical DDP algorithm this is not available, since we only have access to the current state q and v, and the acceleration is computed after the contact forces. In the implicit form, since we have additionally available the next state x', the computation of the acceleration is possible. Thus, we can compute each contact force in closed form as

$$\mathrm{f}_k = P_\mu\{-R^{-1}(\alpha^+ - \alpha^*)\}. \tag{7.10}$$

P_μ projects contact forces to the cone with coefficient μ, described in Algorithm 1.

After the computation of the contact forces, we can enforce the implicit dynamics (7.1) either using a forward or inverse dynamics formulation. Given the available information, the computation of inverse dynamics is cheaper and numerically superior (Hollerbach, 1980; Ferrolho et al., 2020). Furthermore, this decoupling between the forward and backward pass allows us to avoid the rootfinding problem during the forward, that would be necessary for a fully implicit implementation. Having to solve the rootfinding problem in the forward pass increases the computation time of the implicit formulation. We summarize the DDP computations subject to contacts in Algorithms 2 and 3.

Algorithm 3: Backward pass with contacts.

Input: x', x, u, R, and $\mu_k, \forall k$.

Output: κ_i and K_i.

1 Compute α^+, α^*, and f from (7.6), (7.8), and (7.10).

2 Differentiate (7.1) using the computed contact forces f to obtain the required expansion terms of f.

3 Compute the value function terms in (7.3) and (7.4).

4 Compute the Q-function terms in (3.8).

5 Compute the gains κ_i and K_i in (3.9), and the current value function terms in (3.10) for the next step $i-1$.

7.5 Results

7.5.1 Implementation details

The Julia library `RigidBodyDynamics.jl` is used for the computation of the rigid-body dynamics (Koolen and Deits, 2019). Computation of first-, second- and third-order tensors is done using forward-mode automatic differentiation (Revels, Lubin, and Papamarkou, 2016). All results are obtained on an `Intel(R) Xeon(R) CPU E3-1505M V6, 3.00 GHz` with `32GB RAM`.

We begin by performing multiple comparisons between implicit and explicit DDP formulations for a double pendulum swing-up task. Next, we present two problems that require multi-contact motion planning: A single leg that is required a) to stand up from the ground, and b) to balance from an initial random state.

7.5.2 Aggregate double pendulum swing-up

For the double pendulum swing-up task, we generate 100 random trials (that is, with a random initial state) and we specify an objective that includes a desired upright posture at the end of a $T = 5$s horizon, with a time step of 10ms, while penalizing joint torques at intermediate states. Additionally, joint limits are modelled using unilateral forces at the joints. Only the unilateral constraint is

The accompanying code is available at github.com/ichatzinikolaidis/iDDP and the video at youtu.be/w8oOPqo6oC0.

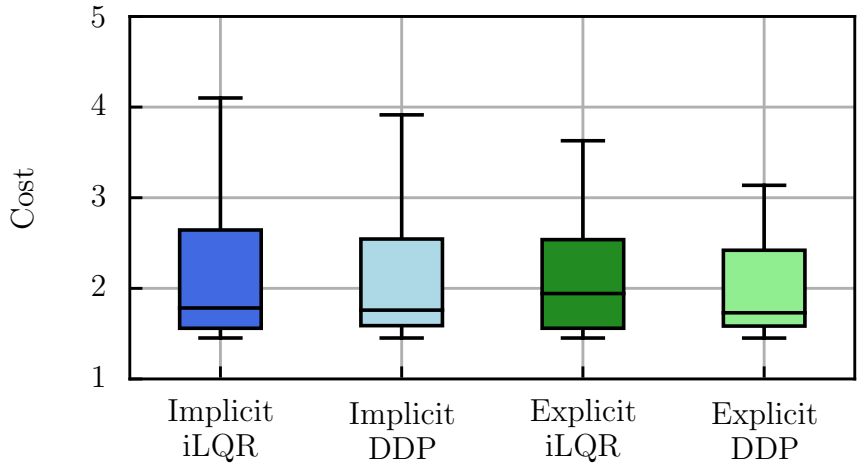


Figure 7.2: Aggregate results for the total trajectory cost of each variant.

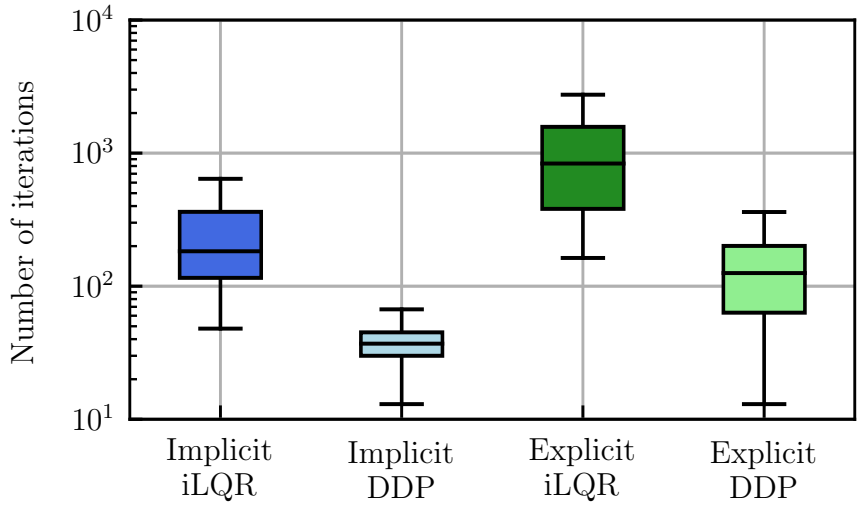


Figure 7.3: Aggregate results for the total number of iterations of each variant.

imposed (forces push the joint away from the limit), while friction is not required.

We compare four variants of the methods presented in this work:

- Implicit DDP with backward Euler dynamics.
- Implicit iLQR with backward Euler dynamics.
- Explicit DDP with forward Euler dynamics.
- Explicit iLQR with forward Euler dynamics.

For every random initialization, the four variants are executed until convergence (or until an upper iteration limit is reached) and the number of iterations and total cost of the trajectory is logged. Aggregate box plot results for the cost and

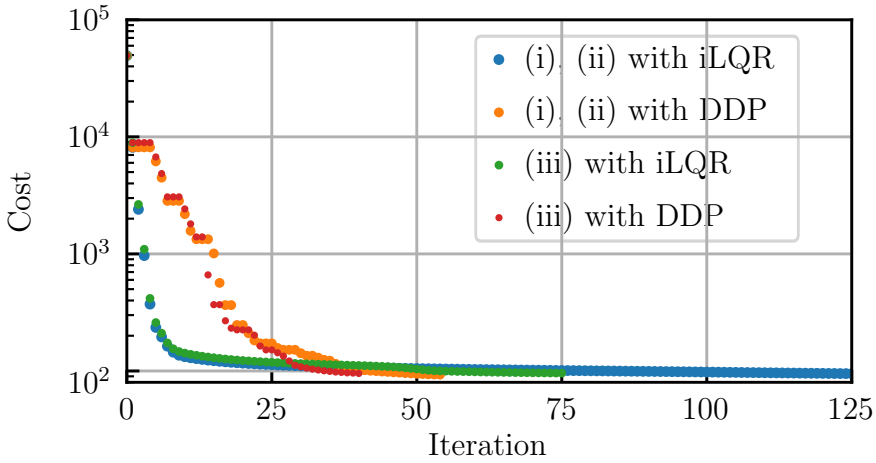


Figure 7.4: Cost per iteration for the different formulations.

the number of iterations are shown in Figures 7.2 and 7.3, respectively.

From the comparison, the implicit formulations result in considerably fewer iterations than the explicit counterparts. Both median, minimum and maximum values, and the rest of the statistical properties in Figure 7.3 are improved with an implicit formulation regarding the number of iterations. As expected, the trade-off for this is the larger in general cost of the resulting trajectory in Figure 7.2. This can be partially explained by the fact that since the explicit formulations perform on average more iterations, they are capable to fine-tune the resulting trajectory more. But given the considerable fewer iterations for the implicit formulations, this aspect is more important in terms of the overall performance.

A possible reason behind this is the integrator's properties. Implicit Euler is an A-stable method suitable even for stiff systems. As such, it usually exhibits energy decrease—instead of the common increase in explicit methods—that makes the whole formulation more stable.

7.5.3 Single double pendulum swing-up

7.5.3.1 Cost per iteration and timings

We evaluate the cost per iteration for one double pendulum swing-up and compare 6 different formulations (each with a DDP and iLQR variant):

(i) Forward Euler dynamics in the forward and backward passes.

(ii) Forward Euler dynamics in the forward pass, and forward Euler inverse

Table 7.1: Effect of time step on number of iterations until convergence.

Time step	10^{-4}	10^{-3}	10^{-2}
(i) / (ii)	56	68	126
(iii)	56	66	75

dynamics in the backward pass.

(iii) Backward Euler dynamics in the forward and backward passes.

We use the same duration and time step as in the previous case but with a different cost function, without joint limits, and initialize at the stable equilibrium. The results are shown in Figure 7.4. Formulation (i) corresponds to a classical iLQR/DDP with explicit dynamics. Formulation (ii) is enabled by the presented framework. The computation of the Jacobian and tensor terms is based on the automatic differentiation of the inverse dynamics. Since (ii) is equivalent to (i), the solutions by the two approaches are exactly the same and are plotted together in Figure 7.4. Differences are found in the computation time, as reported next. Formulation (iii) is implicit in both passes, enabled by the presented framework.

In terms of computation, formulations (i) and (ii) with iLQR require 126 iterations, while with DDP require 55 iterations. In terms of timings, the mean time of each iteration for (i) with iLQR is 5.87ms and with DDP 29.91ms. For (ii) with iLQR is 5.03ms and with DDP 28.49ms. While the differences are not significant for such a low-dimensional model, these can become starker for robot models with larger degrees of freedom. DDP typically scales cubically $O(N^3)$ with the model's degrees of freedom. For (iii), 75 iterations for iLQR and 40 iterations with DDP. The mean computation time of each iterations with iLQR is 7.19ms and with DDP 72.22ms. The increased computation is due to the solution of a nonlinear system of equations in the forward pass.

7.5.3.2 Effect of time step size

We focus now on the effect of the time step size to the solution of the problem. We solve the same problem as before for multiple time step selections and report the number of iterations required until convergence. Since formulations (i) and (ii) are equivalent, we focus the comparison on (i) and (iii). We solve them using

iLQR but similar conclusions could be drawn if DDP was used.

The results are shown on Table 7.1. For small time steps, the two formulations are essentially equivalent and, thus, require the same number of iterations. As the time step increases, the influence of the integrator's damping in (iii) becomes more apparent. This results in a desirable decrease in the number of iterations for convergence. The motions are included in the accompanying video. For larger time steps, the accuracy of both first-order integrators worsens significantly.

7.5.4 Multi-contact stand-up

Next, we consider a planar 3 DoF single leg of a humanoid robot and the task now is to stand-up upright from the ground. The model can make multiple contacts with the terrain using all the bodies of its structure, but self-collisions are inactive. We pre-define a number of possible contact points but we do not prescribe the contact activation pattern. Adding a contact detection mechanism and avoiding the pre-specification of contacts is another possibility, as typically done in simulation engines.

The cost function of the problem is defined as

$$l = w_{q_f}\|q_f - q_{\mathrm{g}}\|^2 + w_{v_f}||v_f||^2 + \sum_i (w_\tau\|\tau_i\|^2 + w_v\|v_i\|^2).$$

Penalization of the velocity and joint torque is applied throughout the trajectory, while a goal state is defined in the final cost term. The motion duration is 4s with a time step of 10ms; this is a relatively large time step for contacts, but our aim here is to output an approximate contact-rich motion plan. Given this plan as input, it is possible to post-process it to increase the quality.

The friction coefficient is selected as $\mu = 0.7$. Parameter R is initialized with a value of 1 for all components. While in principle it can take arbitrary values, we can test the validity from a numerical viewpoint as follows (Roboti LLC, 2020): We run the forward and backward pass separately and compare the computed forces. The two solutions should match according to the desired numerical precision.

Snapshots of the computed motion are shown at the top row of Figure 7.5. The main difficulty is that the problem exhibits a number of contact possibilities. Thus, mode enumeration can be very challenging. Notice also how delicate heel balance emerges while reaching the upright configuration. Our TO framework is

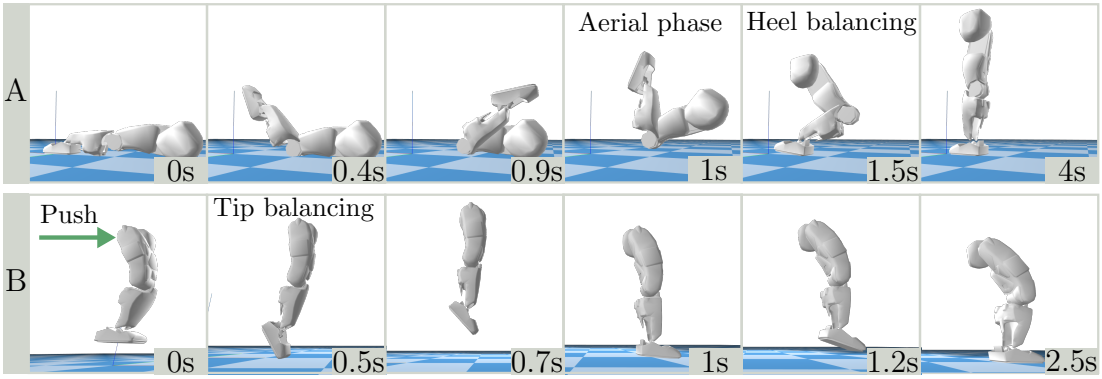


Figure 7.5: Time-lapse snapshots of contact-rich motions: (A) Standing-up from ground by dynamic maneuvers using large momentum with aerial phases. (B) Robust balancing by toe tipping and jumping that withstands an external push.

capable of outputting a locally optimal motion plan. Even though a zero torque initial solution is used here, its quality greatly affects the quality of the computed motion. Finally, by changing the terms in the cost function, it is possible to obtain different solutions, *e.g.* more conservative but with higher torque cost.

Figure 7.6 shows the base and joint positions during the task execution. There is an initial explosive and dynamic motion at about 1s. Such a motion would be in practice difficult to track. Yet being able to compute such a complex motion from high-level input only demonstrates the power of DDP-based approaches. There are a couple of ways to mitigate that: an obvious approach is to increase the torque, position, and velocity penalizations accordingly. Another option is to include terms that penalize the rate of the commanded torques. Finally, a more principled approach is to penalize high-frequency components of the signals involved (Grandia et al., 2019).

7.5.5 Multi-contact balancing

Using the same model as before, the state now is randomly initialized in the air. The task is to keep the initial posture with zero velocity, *i.e.* to balance. In contrast to the previous case, this problem is formulated in a MPC fashion. A fixed number of 15 iterations for DDP is pre-specified; this makes real-time iterations of the algorithm possible. The horizon length is $T = 0.5$s, with the simulation running at 200Hz, while our framework runs at 20Hz. The structure of the cost function remains the same as before, albeit the weight regarding the final

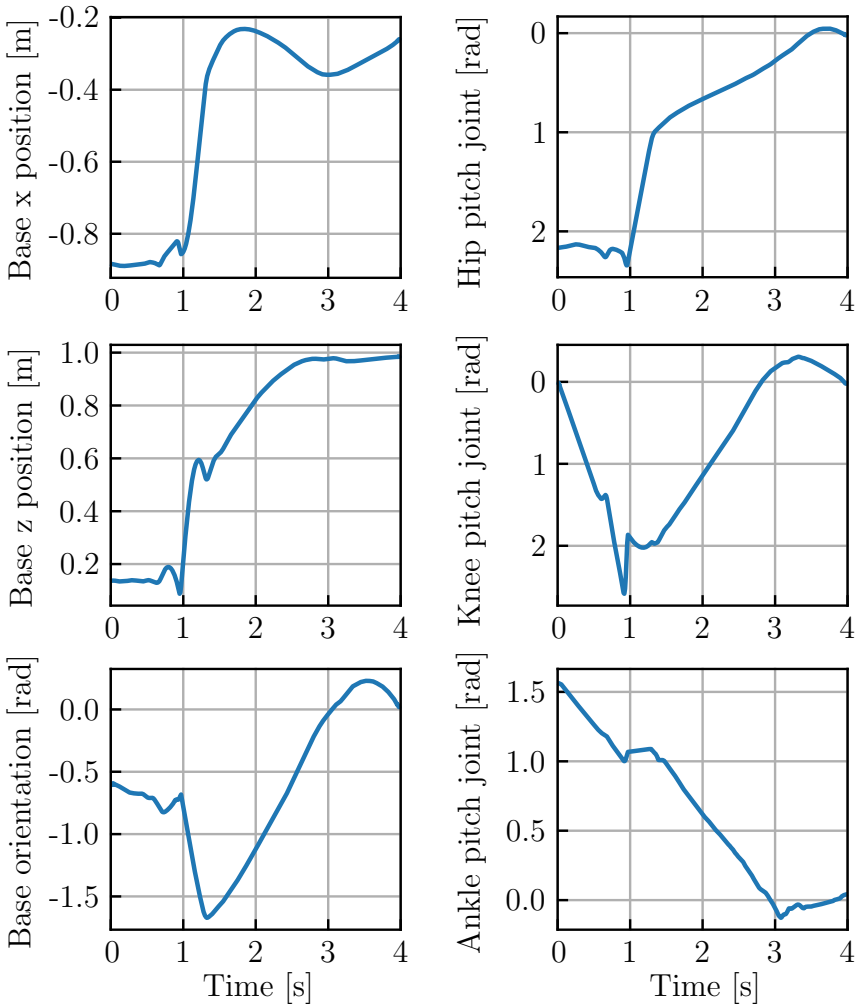


Figure 7.6: Base and joint positions during the standing up task.

velocity is increased to bias more towards a static final configuration.

The balancing motion is shown at the bottom of Figure 7.5. The computed motion naturally performs a series of jumps to dissipate kinetic energy and come to a complete stop. The underactuated foot tilting emerged as the outcome of optimization without the need for programming explicit controllers as in (Z. Li, Zhou, Zhu, et al., 2017). Compared to the case in the previous section, the receding horizon formulation is capable of producing better motions in general. This is because the constant updates allow it to escape iterations with a very small cost decrease, which can be common in the fixed horizon optimization of the previous case. If a bad initialization is specified or the horizon and frequency are not chosen properly, the receding horizon formulation can be trapped too. The selection of these parameters depends on the desired task and the initial state.

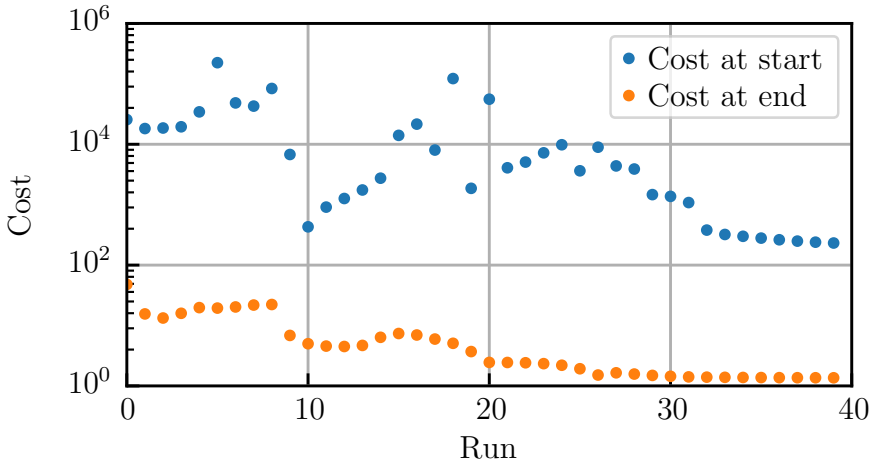


Figure 7.7: Semi-log plot of the total trajectory cost at the start (blue) and end (red) of each run for the receding horizon formulation.

Finally, a semi-log plot of the total trajectory cost at the beginning and at the end of each DDP step is shown in Figure 7.7. We notice that in about 20 runs a successful balancing motion is computed. Afterwards, each run rapidly converges to this motion. The reason why the cost is increased at the beginning of each run is because the horizon moves; the predicted trajectory for the new segment at the end of the previous horizon is that the robot will essentially fall, which incurs a large cost. Additionally, during the initial runs, the motion is highly unstable and a suitable balancing motion is not discovered yet. Thus, the total trajectory cost varies greatly between consecutive runs.

7.6 Conclusion

This chapter presented an application of DDP suited for systems with implicitly defined dynamics that can handle dynamical interactions, with a particular focus on through-contact motion planning. This allowed extending the original DDP to a larger class of dynamics models, *e.g.* models based on inverse dynamics. We described how to use the implicit formulation for accurate contact resolution in the DDP framework without requiring approximations of contact dynamics. The proposed method is exact and straightforward to implement, utilizing a closed-form solution for quality gradient computations. We demonstrated its properties in a number of cases: comparisons of implicit and explicit dynamics for a double pendulum, and two case studies for a single leg model that required challenging

multi-contact motion plans.

While the original DDP provides both feedforward and feedback gains that guarantee a level of robustness against small perturbations, we noticed that the computed motion plans can sometimes fail if the conditions of the problem change slightly. Though one can introduce robustness as part of the trajectory optimization modelling, we believe that running the framework in a receding horizon fashion is more appropriate and promising. Thus, the motion plans should be updated online to withstand unexpected perturbations.

It is worth noting that DDP simulates the dynamics of the system and activates a contact point if appropriate. Thus, contacts are taken into account according to the system's natural dynamics (Posa, Cantu, and Tedrake, 2014), which may lead to abrupt motions (Tassa, Erez, and Todorov, 2012). Being a shooting method for unconstrained systems, DDP is limited in terms of active search for potential contacts. Further improvements can be made by combinatorial planning and exploration, where transcription-based methods demonstrated better capabilities and flexibility (Chatzinikolaidis, You, and Z. Li, 2020; Patel et al., 2019), although requiring additional and non-negligible computation cost in practice.

Chapter 8

Conclusion

Robots that are able to seamlessly interact and work right next to humans are a major expectation from robotics as a scientific and applied discipline. While the field has progressed rapidly the past couple of years, there is quite a long distance to cover to achieve this goal. For legged robots in particular, the progress has been tremendous. From being strictly confined to laboratories only a few years back, legged robots have started to become widely commercialized now. Their deployment is currently focused on niche applications, but in the future these will inevitably become much wider.

In this context, robots that are able to perform motions in contact-rich settings are necessary. For commercial legged robots, tasks like walking—when it is acceptable to perform contacts with the end-effectors only on rigid and flat environments—tend to converge to acceptable solutions. Recovering from unexpected events, performing tasks by exploiting the whole structure, and realizing whole-body multi-contact motions in a compliant manner cannot be reliably performed yet. The former goals require heavier emphasis on the motion planning aspect, while the latter goal on the control aspect. This thesis focuses primarily on the motion planning side.

8.1 Summary

Specifically in this thesis, the motion planning problem in the presence of unspecified contacts is analyzed. The target is to be able to perform a variety of

tasks using a holistic framework (that does not require designing or changing the solution structure for every task individually), while requiring high-level tasks' description only (in the form of cost functions). This thesis constitutes a step towards this direction. It provides novel approaches to achieve this, investigating the problem from different perspectives, each with its own merits and limitations.

In Chapter 5, a task space optimization-based formulation is proposed. Modelling in task space allows to take into account constraints imposed by the environment, such as gaps and obstacles, in a straightforward manner. An integrator with second-order accuracy for the dynamics is proposed, which is able to exactly represent the translational and more importantly the rotational dynamics of the model. Accuracy and rotational dynamics satisfaction are important for successfully translating the computed trajectories to joint space using a whole-body inverse dynamics controller.

Chapter 6 presents a method formulated directly in joint space that aims to address some of the limitations of the previous work. Specifically, it takes into account the full dynamics and kinematics of the system and does not require any prior information about the contacts; only pre-specifying the available contact points. Furthermore, it is able to compute motions in a more general settings than traditional motion planning approaches, including properties of the environment such as being slippery and soft. Its advantages are demonstrated by comparing against previous work and its feasibility by computing motion plans for a high-dimensional quadrupedal robot model. For the latter motion computations, adaptations for a trotting gait on hard and slippery terrains are shown, and for a jumping motion on hard and soft terrains.

Chapter 7 presents an extension of DDP for dynamical systems defined implicitly, enabling its application to a wider set of models. Furthermore, an application for through-contact motion planning is presented. It is shown how to resolve contacts in the acceleration level, with similar performance to time-stepping approaches, and how an implicit definition of contacts in DDP enables exact contact modelling, fully respecting unilateral and friction cone constraints. Full kinematics and dynamics are facilitated, as in the previous case. But additionally, pre-specification of the contact points is not required, enabling motion planning with contacts everywhere around the model's structure. Comparison between implicitly and explicitly defined dynamics for a double pendulum model are shown.

Finally, two cases studies for a single-leg model capable to contact with all of its body are presented. The first is a standing up task, while the second a balancing task formulated in a MPC fashion.

All previous methods output nominal motion plans which would be in practice difficult to track even in simulation. Chapter 4 discusses the necessary background on WBC, which is the state-of-the-art approach for tracking motion plans for legged robots. Apart from presenting an overview of the methodology, a specific implementation is elaborated. Lastly, two applications of the controller for WBC of two high-dimensional humanoid robots are presented.

Along with the main material of this thesis, Chapter 1 presents an introduction to the problem and the main contributions of this work, Chapter 2 discusses prior work in motion planning by holistic optimization in detail and a more general context of prior works in legged locomotion, and Chapter 3 the necessary theoretical background that forms the basis of the topics discussed.

8.2 Limitations

At this point, it is important to underline limitations of the discussed topics. These essentially stimulate the discussion about possible future directions and improvements. The focus is on the limitations presented on the three main chapters of the thesis (Chapters 5 to 7). Limitations of previous work are discussed in Chapter 2 and in the context of WBC at the end of Chapter 4.

For Chapter 5, a limitation is the need to pre-specify the gait pattern, since this is not independent from the rest of the motion planning problem. Additionally, limitations arise due to the simplified model used. While translational and rotational dynamics are satisfied—at the discretization points at least—its applicability for commanding WBC motions requires further investigation. This is partially mitigated by the fact the multiple previous works successfully use the single rigid body for commanding dynamic motions, typically with lower accuracy. For the formulation that does not require the gate pattern as input, the biggest and important hurdle is the lack of sparsity.

For Chapter 6, the biggest limitation is the computation speed, which will be discussed in more detail at the end of this section. Another issue is the selection

of the parameters that define the properties of the environment. As already discussed, there exists an important trade-off between solution speed and accurate environmental modelling; pushing the contact forces model's parameters towards a non-smooth representation makes the overall solution more challenging. A mechanism for automatically adapting these parameters similar to the mechanism of IP methods for controlling the barrier parameters is a potential solution, but a more thorough study is required.

For Chapter 7, the limitations are the same to the ones that in general DDP approaches share. The most important limitation of DDP approaches, as already discussed, is the limited contact discovery capabilities, requiring more careful cost function selection to properly guide the solution. For the specific method proposed, care needs to be taken regarding the impact of the increased computations due to the additional derivative computation requirements. Yet for a careful implementation, there is only a minor overhead with respect to the original DDP formulation.

Finally, there are three main issues that affect all presented approaches: Initialization, cost function selection, and computation speed. Proper initialization is an issue that in general plagues non-convex optimization-based approaches. For the proposed methods, the implicit DDP is the one that suffers the most due to its shooting basis. The initialization depends on the control input only, making it harder to identify good initial seeds in the control space. Both the task and joint space TO approaches require initializations with state variables too. These variables are more intuitive; there are easy to implement heuristics that can provide a good starting point.

Regarding the cost function selection for non-convex optimization approaches, hierarchical structures are less favourable than weighted approaches, both for theoretical reasons as well as due to the non-negligible computation cost. Yet weighting selection is challenging too. The task space method is the most insensitive due to the hard constraints encoding of the task's properties, and as a result the less automatic. The joint space approach offers a good balance, with a harder to select cost function than the task space approach. Yet the ability to automatically activate and deactivate contacts makes the selection less sensitive than the task-space approach. The DDP method is the hardest to tune because of the lack of constraints support in the formulation and the inability to actively

explore potential contacts.

The last common issue is the computation speed, which is a critical component for the real-time applications with unspecified contacts that this work envisions. For the task and joint space methods, the current computation rates are unacceptable for real-time implementations. While exact computation times vary greatly with the task and our inputs (e.g. initialization, environment model), for the task-space method the real-time factor for the single leg model can typically be 2× to 10×. For the joint-space method and the `Anymal` model, timings can be even worse, like 20× to 500×. Thus, the sparsity in the constraint Jacobian that these methods exploit is not sufficient and greater effort in improving this aspect is required, which could potentially lead to custom optimization solver implementations. The DDP method is more favourable in this aspect and can be run in real-time with a proper implementation, as relevant prior work has already demonstrated. This points to another property that time discretized formulations should exploit: sparsity in the time horizon. Instead of doing a global quadratic approximation of the problem, a local approximation around each time step seems more appropriate for faster computations.

8.3 Future directions

Based on these limitations, there are a couple of possible future directions to improve aspects of the proposed approaches. There is much exciting research work to be done before fully autonomous robots—able to fluently interact with the environment—are realized.

8.3.1 Informed initialization

Providing suitable initializations to non-convex optimization problems is a particularly effective approach to improve their computation rates. Even simple strategies, like linear interpolation between the initial state and end goal, can provide improvements. While in general guarantees from initialization would be difficult to extract (providing proper initialization to general non-convex optimization should be NP-hard, as solving the optimization problem itself is), exploring different initialization approaches might uncover effective strategies in domain-specific robotic applications. As a result, improved strategies can positively affect the

computation times of all presented strategies.

One possible approach is to bootstrap models of varying complexity, while providing as input the solutions by the simpler model. In the context of animation, this is demonstrated by D. Zimmermann et al. (2015). The aim is that the more complicated models will converge in a small number of iterations. It will be also attractive if the whole scheme converges faster than a single optimization with a simple initialization strategy. Instead of a strict hierarchical structure, the more complicated models should be able to adapt the solution by the simpler models according to their needs, without fixing parts of the solution (*e.g.* the gait as computed by the simpler models).

A somewhat orthogonal approach is to setup a single optimization problem but vary the model's complexity with the time horizon (H. Li, Frei, and Wensing, 2021). While this does not provide initialization per se, the different models can provide information about the future cost—similar to value function approximation in aggregate dynamic programming approaches (Powell, 2009). This can make initialization for the latter parts of the trajectory easier, and overall the incorporation of simple models can provide computational improvements.

Another recently explored area is to somehow store solutions and effectively retrieve them subsequently, as a memory of motion approach (Mansard, Del Prete, et al., 2018). In contrast to the previous methods, this approach retains the same model throughout the horizon and does not solve multiple problems, making it the most direct method to tackle the problem. Yet building the database can be challenging, for example due to discontinuities (Merkt et al., 2020).

A final approach is to use logic-geometric programming which provides a hierarchical solution to the problem (Toussaint, 2015). This approach enables an alternating search between symbolic action spaces and continuous optimization of the configurations dictated by the symbolic decisions. The symbolic search is formulated efficiently and provides a skeleton based on the computed action sequence. This skeleton achieves two purposes: 1) It constraints the mode sequence, simplifying the TO formulation. 2) Simplifies the initialization of the variables within the modes.

8.3.2 Objective function definition

A different research strand should focus on the cost function definition, for example proper selection of the constituting weights. Machine learning approaches, *e.g.* mixture of experts, can provide a structured way to combine a set of pre-defined primitive cost terms. Inspiration for the selection of the primitive cost terms could be the terms typically used in WBC approaches. The aim should be generalization across tasks. Currently, state-of-the-art TO methods are formulated with separate and distinct cost function selection for each task. For each task, the cost function is manually tuned—a tedious and error-prone process.

8.3.3 Sparsity in the time horizon

As already mentioned, exploitation of the problems' sparsity in the time horizon for transcription-based optimal control approaches can significantly improve the solution speeds. This is inherent in dynamic programming approaches and variants thereof, due to the principle of optimality. For unconstrained and equality constrained problems, Ricatti-based approaches provide a mature way to do that. For inequality constrained problems, *e.g.* friction cone, the exact solution strategy is unclear. Inspiration can be drawn from the optimization literature, for example by combining tools for active set, barrier, and augmented Lagrangian methods to transform it to an equality constrained problem. Alternatively, QP could potentially be leveraged to handle the inequalities, similar to what was done for WBC methods.

8.3.4 Higher-order methods

Finally, the work on implicit DDP opens up the avenue of introducing higher accuracy integration schemes in DDP settings. Such studies have already been conducted for contact implicit transcription-based TO (Manchester, Doshi, et al., 2019; Patel et al., 2019), but not on DDP methods to the best of our knowledge. Additionally, an issue that was identified to the current implicit DDP status is the sensitivitive to small perturbations that can fundamentally alter the contact pattern. This makes it difficult to track the computed plans even in simulations, which aligns with the concern expressed by Posa, Kuindersma, and Tedrake (2016). High accuracy integration schemes might be possible to mitigate this issue.

Bibliography

B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. Caldwell, J. Cappelletto, J. Grieco, G. Fernández-López, and C. Semini (2018). "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization". In: *IEEE Robotics and Automation Letters* 3.3, pp. 2531–2538. DOI: 10.1109/LRA.2017.2779821 (cit. on pp. 22, 62).

M. Al Borno, M. de Lasa, and A. Hertzmann (2013). "Trajectory optimization for full-body movements with complex contacts". In: *IEEE Transactions on Visualization and Computer Graphics* 19.8, pp. 1405–1414. DOI: 10.1109/TVCG.2012.325 (cit. on p. 22).

A. Ames and I. Poulakakis (2017). In: *Bioinspired Legged Locomotion*. Ed. by M. Sharbafi and A. Seyfarth. Butterworth-Heinemann. Chap. 4.7 Hybrid zero dynamics control of legged robots. DOI: 10.1016/B978-0-12-803766-9.00006-3 (cit. on p. 20).

J. Andersson, J. Gillis, G. Horn, J. Rawlings, and M. Diehl (2018). "CasADi – A software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* 11.1. DOI: 10.1007/s12532-018-0139-4 (cit. on pp. 71, 90).

J. Baumgarte (1972). "Stabilization of constraints and integrals of motion in dynamical systems". In: *Computer Methods in Applied Mechanics and Engineering* 1.1, pp. 1–16. DOI: 10.1016/0045-7825(72)90018-7 (cit. on p. 45).

R. Bellman and S. Dreyfus (1962). *Applied dynamic programming*. Princeton University Press. ISBN: 9781400874651 (cit. on pp. 28, 105).

J. Betts (2010). *Practical methods for optimal control and estimation using nonlinear programming*. SIAM. DOI: 10.1137/1.9780898718577 (cit. on pp. 33, 34, 65, 84, 85, 89).

G. Boutselis and E. Theodorou (2020). "Discrete-time differential dynamic programming on Lie groups: Derivation, convergence analysis and numerical results". In: *IEEE Transactions on Automatic Control*, pp. 1–1. DOI: 10.1109/TAC.2020.3034206 (cit. on p. 105).

K. Bouyarmane, S. Caron, A. Escande, and A. Kheddar (2019). "Multi-contact motion planning and control". In: *Humanoid Robotics: A Reference*. Ed. by A. Goswami and P. Vadakkepat. Springer Netherlands, pp. 1763–1804. DOI: 10.1007/978-94-007-6046-2_32 (cit. on p. 17).

K. Bouyarmane and A. Kheddar (2018). "On weight-prioritized multitask control of humanoid robots". In: *IEEE Transactions on Automatic Control* 63.6, pp. 1632–1647. DOI: 10.1109/TAC.2017.2752085 (cit. on p. 50).

T. Bretl (2006). "Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem". In: *International Journal of Robotics Research* 25.4, pp. 317–342. DOI: 10.1177/0278364906063979 (cit. on p. 17).

R. Budhiraja, J. Carpentier, C. Mastalli, and N. Mansard (2018). "Differential dynamic programming for multi-phase rigid contact dynamics". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 1–9. DOI: 10.1109/HUMANOIDS.2018.8624925 (cit. on p. 105).

A. Cangelosi, J. Bongard, M. Fischer, and S. Nolfi (2015). "Embodied intelligence". In: *Springer Handbook of Computational Intelligence*. Ed. by J. Kacprzyk and W. Pedrycz. Springer, pp. 697–714. DOI: 10.1007/978-3-662-43505-2_37 (cit. on p. 2).

J. Carius, R. Ranftl, V. Koltun, and M. Hutter (2018). "Trajectory optimization with implicit hard contacts". In: *IEEE Robotics and Automation Letters* 3.4, pp. 3316–3323. DOI: 10.1109/LRA.2018.2852785 (cit. on pp. 84, 106).

J. Carius, R. Ranftl, V. Koltun, and M. Hutter (2019). "Trajectory optimization for legged robots with slipping motions". In: *IEEE Robotics and Automation*

Letters 4.3, pp. 3013–3020. DOI: 10.1109/LRA.2019.2923967 (cit. on pp. 83, 97, 103).

S. Caron, Q.-C. Pham, and Y. Nakamura (2015). "Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench for rectangular support areas". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 5107–5112. DOI: 10.1109/ICRA.2015.7139910 (cit. on p. 55).

S. Caron, Q.-C. Pham, and Y. Nakamura (2017). "ZMP support areas for multicontact mobility under frictional constraints". In: *IEEE Transactions on Robotics* 33.1, pp. 67–80. DOI: 10.1109/TRO.2016.2623338 (cit. on p. 11).

J. Carpentier and N. Mansard (2018). "Multicontact locomotion of legged robots". In: *IEEE Transactions on Robotics* 34.6, pp. 1441–1460. DOI: 10.1109/TRO.2018.2862902 (cit. on pp. 17, 83).

G. Cavagna, H. Thys, and A. Zamboni (1976). "The sources of external work in level walking and running". In: *The Journal of Physiology* 262.3, pp. 639–657. DOI: 10.1113/jphysiol.1976.sp011613 (cit. on p. 11).

I. Chatzinikolaidis and Z. Li (2021). "Trajectory optimization of contact-rich motions using implicit differential dynamic programming". In: *IEEE Robotics and Automation Letters*. Vol. 6. 2, pp. 2626–2633. DOI: 10.1109/LRA.2021.3061341 (cit. on pp. 8, 101).

I. Chatzinikolaidis, T. Stouraitis, S. Vijayakumar, and Z. Li (2018). "Nonlinear optimization using discrete variational mechanics for dynamic maneuvers of a 3D one-leg hopper". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 932–937. DOI: 10.1109/HUMANOIDS.2018.8624981 (cit. on pp. 7, 61, 84, 102).

I. Chatzinikolaidis, Y. You, and Z. Li (2020). "Contact-implicit trajectory optimization using an analytically solvable contact model for locomotion on variable ground". In: *IEEE Robotics and Automation Letters* 5.4, pp. 6357–6364. DOI: 10.1109/LRA.2020.3010754 (cit. on pp. 7, 82, 120).

J. Chestnutt, J. Kuffner, K. Nishiwaki, and S. Kagami (2003). "Planning biped navigation strategies in complex environments". In: *Proc. IEEE International Conference on Humanoid Robots* (cit. on p. 16).

H. Dai, A. Valenzuela, and R. Tedrake (2014). "Whole-body motion planning with centroidal dynamics and full kinematics". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 295–302. DOI: 10.1109/HUMANOIDS.2014.7041375 (cit. on p. 21).

H. Dallali, P. Kormushev, Z. Li, and D. Caldwell (2012). "On global optimization of walking gaits for the compliant humanoid robot, COMAN using reinforcement learning". In: *Cybernetics and Information Technologies* 12.3, pp. 39–52. URL: www.cit.iit.bas.bg/CIT_2012/v12-3/Dallali-Kormushev-Li-Caldwell-39-52.pdf (cit. on p. 84).

C. de Boor (1978). *A practical guide to splines*. Springer-Verlag (cit. on p. 71).

V. De Sapio, O. Khatib, and S. Delp (2006). "Task-level approaches for the control of constrained multibody systems". In: *Multibody System Dynamics* 16.1, pp. 73–102. DOI: 10.1007/s11044-006-9017-3 (cit. on p. 51).

R. Deits and R. Tedrake (2014). "Footstep planning on uneven terrain with mixed-integer convex optimization". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 279–286. DOI: 10.1109/HUMANOIDS.2014.7041373 (cit. on p. 16).

S. Dempe and J. Dutta (2012). "Is bilevel programming a special case of a mathematical program with complementarity constraints?" In: *Mathematical Programming* 131.1, pp. 37–48. DOI: 10.1007/s10107-010-0342-1 (cit. on p. 84).

J. Di Carlo, P. Wensing, B. Katz, G. Bledt, and S. Kim (2018). "Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 1–9. DOI: 10.1109/IROS.2018.8594448 (cit. on p. 17).

A. Dietrich (2016). *Whole-body impedance control of wheeled humanoid robots*. Springer Tracts in Advanced Robotics. Springer International Publishing. DOI: 10.1007/978-3-319-40557-5 (cit. on p. 50).

E. Drumwright and D. Shell (2010). "Modeling contact friction and joint friction in dynamic robotic simulation using the principle of maximum dissipation". In: *Algorithmic Foundations of Robotics IX*. Springer Tracts in Advanced Robotics.

Springer, Berlin, Heidelberg, pp. 249–266. DOI: 10.1007/978-3-642-17452-0_15 (cit. on p. 85).

J. Englsberger, C. Ott, and A. Albu-Schäffer (2015). "Three-dimensional bipedal walking control based on divergent component of motion". In: *IEEE Transactions on Robotics* 31.2, pp. 355–368. DOI: 10.1109/TRO.2015.2405592 (cit. on pp. 12, 61).

J. Englsberger, C. Ott, M. Roa, A. Albu-Schäffer, and G. Hirzinger (2011). "Bipedal walking control based on capture point dynamics". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 4420–4427. DOI: 10.1109/IROS.2011.6094435 (cit. on p. 12).

T. Erez and E. Todorov (2012). "Trajectory optimization for domains with contacts using inverse dynamics". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 4914–4919. DOI: 10.1109/IROS.2012.6386181 (cit. on pp. 84, 90).

A. Escande and A. Kheddar (2009). "Contact planning for acyclic motion with tasks constraints". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 435–440. DOI: 10.1109/IROS.2009.5354371 (cit. on p. 16).

A. Escande, N. Mansard, and P.-B. Wieber (2014). "Hierarchical quadratic programming: Fast online humanoid-robot motion generation". In: *International Journal of Robotics Research* 33.7, pp. 1006–1028. DOI: 10.1177/0278364914521306 (cit. on p. 52).

S. Fahmi, M. Focchi, A. Radulescu, G. Fink, V. Barasuol, and C. Semini (2020). "STANCE: Locomotion adaptation over soft terrain". In: *IEEE Transactions on Robotics* 36.2, pp. 443–457. DOI: 10.1109/TRO.2019.2954670 (cit. on p. 83).

M. Fallon, S. Kuindersma, S. Karumanchi, M. Antone, T. Schneider, H. Dai, C. D'Arpino, R. Deits, M. DiCicco, D. Fourie, T. Koolen, P. Marion, M. Posa, A. Valenzuela, K.-T. Yu, J. Shah, K. Iagnemma, R. Tedrake, and S. Teller (2015). "An architecture for online affordance-based perception and whole-body planning". In: *Journal of Field Robotics* 32.2, pp. 229–254. DOI: 10.1002/rob.21546 (cit. on p. 16).

R. Featherstone (2008). *Rigid body dynamics algorithms*. Springer US. DOI: 10.1007/978-1-4899-7560-7 (cit. on p. 67).

S. Feng, E. Whitman, X. Xinjilefu, and C. Atkeson (2015). "Optimization-based full body control for the DARPA Robotics Challenge". In: *Journal of Field Robotics* 32.2, pp. 293–312. DOI: 10.1002/rob.21559 (cit. on pp. 54, 55).

S. Feng, X. Xinjilefu, C. Atkeson, and J. Kim (2016). "Robust dynamic walking using online foot step optimization". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 5373–5378. DOI: 10.1109/IROS.2016.7759790 (cit. on p. 16).

P. Fernbach, S. Tonneau, O. Stasse, J. Carpentier, and M. Taïx (2020). "C-CROC: Continuous and convex resolution of centroidal dynamic trajectories for legged robots in multicontact scenarios". In: *IEEE Transactions on Robotics* 36.3, pp. 676–691. DOI: 10.1109/TRO.2020.2964787 (cit. on p. 17).

H. Ferrolho, V. Ivan, W. Merkt, I. Havoutis, and S. Vijayakumar (2020). "Inverse dynamics vs. forward dynamics in direct transcription formulations for trajectory optimization". In: arXiv: 2010.05359 [cs.RO] (cit. on p. 111).

M. Focchi, A. del Prete, I. Havoutis, R. Featherstone, D. Caldwell, and C. Semini (2017). "High-slope terrain locomotion for torque-controlled quadruped robots". In: *Autonomous Robots* 41, pp. 259–272. DOI: 10.1007/s10514-016-9573-1 (cit. on p. 53).

G. Frison (2015). "Algorithms and Methods for High-Performance Model Predictive Control". PhD thesis. Technical University of Denmark (DTU). URL: https://orbit.dtu.dk/en/publications/algorithms-and-methods-for-high-performance-model-predictive-cont (cit. on p. 24).

R. Full and D. Koditschek (1999). "Templates and anchors: Neuromechanical hypotheses of legged locomotion on land". In: *Journal of Experimental Biology* 202.23, pp. 3325–3332. URL: https://jeb.biologists.org/content/202/23/3325 (cit. on pp. 10, 13).

S. Gangapurwala, A. Mitchell, and I. Havoutis (2020). "Guided constrained policy optimization for dynamic quadrupedal robot locomotion". In: *IEEE Robotics and Automation Letters* 5.2, pp. 3642–3649. DOI: 10.1109/LRA.2020.2979656 (cit. on p. 26).

M. Geisert, A. Del Prete, N. Mansard, F. Romano, and F. Nori (2017). "Regularized hierarchical differential dynamic programming". In: *IEEE Transactions on Robotics* 33.4, pp. 819–833. DOI: 10.1109/TRO.2017.2671355 (cit. on p. 105).

H. Geyer, A. Seyfarth, and R. Blickhan (2006). "Compliant leg behaviour explains basic dynamics of walking and running". In: *Proceedings of the Royal Society B: Biological Sciences* 273.1603, pp. 2861–2867. DOI: 10.1098/rspb.2006.3637 (cit. on p. 13).

M. Gienger, M. Toussaint, and C. Goerick (2010). "Whole-body motion planning – Building blocks for intelligent systems". In: *Motion planning for humanoid robots*. Ed. by K. Harada, E. Yoshida, and K. Yokoi. Springer London, pp. 67–98. DOI: 10.1007/978-1-84996-220-9_3 (cit. on p. 51).

M. Giftthaler, M. Neunert, M. Stäuble, J. Buchli, and M. Diehl (2018). "A family of iterative Gauss-Newton shooting methods for nonlinear optimal control". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 1–9. DOI: 10.1109/IROS.2018.8593840 (cit. on p. 107).

R. Goebel, R. Sanfelice, and A. Teel (2009). "Hybrid dynamical systems". In: *IEEE Control Systems Magazine* 29.2, pp. 28–93. DOI: 10.1109/MCS.2008.931718 (cit. on p. 19).

R. Grandia, F. Farshidian, A. Dosovitskiy, R. Ranftl, and M. Hutter (2019). "Frequency-aware model predictive control". In: *IEEE Robotics and Automation Letters* 4.2, pp. 1517–1524. DOI: 10.1109/LRA.2019.2895882 (cit. on p. 117).

A. Haddadi and K. Hashtrudi-Zaad (2012). "Real-time identification of Hunt–Crossley dynamic models of contact environments". In: *IEEE Transactions on Robotics* 28.3, pp. 555–566. DOI: 10.1109/TRO.2012.2183054 (cit. on p. 46).

E. Hairer and G. Wanner (1996). *Solving Ordinary Differential Equations II*. Springer. DOI: 10.1007/978-3-642-05221-7 (cit. on p. 104).

C. Hargraves and S. Paris (1987). "Direct trajectory optimization using nonlinear programming and collocation". In: *Journal of Guidance, Control, and Dynamics* 10.4, pp. 338–342. DOI: 10.2514/3.20223 (cit. on p. 34).

K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox (2008). "Motion planning for legged robots on varied terrain". In: *International Journal of*

Robotics Research 27.11–12, pp. 1325–1349. DOI: 10.1177/0278364908098447 (cit. on p. 18).

B. Henze, M. Roa, and C. Ott (2016). "Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios". In: *International Journal of Robotics Research* 35.12, pp. 1522–1543. DOI: 10.1177/0278364916653815 (cit. on p. 53).

A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl (2010). "Online walking motion generation with automatic footstep placement". In: *Advanced Robotics* 24.5-6, pp. 719–737. DOI: 10.1163/016918610X493552 (cit. on p. 16).

A. Herdt, N. Perrin, and P.-B. Wieber (2010). "Walking without thinking about it". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 190–195. DOI: 10.1109/IROS.2010.5654429 (cit. on p. 16).

A. Hereid, E. Cousineau, C. Hubicki, and A. Ames (2016). "3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1447–1454. DOI: 10.1109/ICRA.2016.7487279 (cit. on p. 20).

H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa (2006). "A universal stability criterion of the foot contact of legged robots - adios ZMP". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1976–1983. DOI: 10.1109/ROBOT.2006.1641995 (cit. on p. 11).

N. Hogan (1985). "Impedance control: An approach to manipulation: Part I—Theory". In: *Journal of Dynamic Systems, Measurement, and Control* 107.1, pp. 1–7. DOI: 10.1115/1.3140702 (cit. on p. 49).

J. Hollerbach (1980). "A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity". In: *IEEE Transactions on Systems, Man, and Cybernetics* 10.11, pp. 730–736. DOI: 10.1109/TSMC.1980.4308393 (cit. on p. 111).

P. Horak and J. Trinkle (2019). "On the similarities and differences among contact models in robot simulation". In: *IEEE Robotics and Automation Letters* 4.2, pp. 493–499. DOI: 10.1109/LRA.2019.2891085 (cit. on pp. 42, 44, 45, 92, 110).

HSL (2020). *HSL. A collection of Fortran codes for large scale scientific computation.* URL: http://www.hsl.rl.ac.uk/ (cit. on p. 90).

W. Hu, I. Chatzinikolaidis, K. Yuan, and Z. Li (2018). "Comparison study of nonlinear optimization of step durations and foot placement for dynamic walking". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 433–439. DOI: 10.1109/ICRA.2018.8461101 (cit. on pp. 12, 61).

M. Hutter, C. Gehring, D. Jud, A. Lauber, C. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger (2016). "ANYmal - a highly mobile and dynamic quadrupedal robot". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 38–44. DOI: 10.1109/IROS.2016.7758092 (cit. on pp. 3, 40, 96).

D. Huynh (2009). "Metrics for 3D rotations: Comparison and analysis". In: *Journal of Mathematical Imaging and Vision* 35.2, pp. 155–164. DOI: 10.1007/s10851-009-0161-2 (cit. on p. 72).

J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter (2019). "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26. DOI: 10.1126/scirobotics.aau5872 (cit. on p. 25).

S.-H. Hyon (2009). "Compliant terrain adaptation for biped humanoids without measuring ground surface and contact forces". In: *IEEE Transactions on Robotics* 25.1, pp. 171–178. DOI: 10.1109/TRO.2008.2006870 (cit. on p. 52).

S.-H. Hyon, J. Hale, and G. Cheng (2007). "Full-body compliant human–humanoid interaction: Balancing in the presence of unknown external forces". In: *IEEE Transactions on Robotics* 23.5, pp. 884–898. DOI: 10.1109/TRO.2007.904896 (cit. on p. 52).

O. Junge, J. Marsden, and S. Ober-Blöbaum (2005). "Discrete mechanics and optimal control". In: *IFAC Proceedings Volumes* 38.1, pp. 538–543. DOI: 10.3182/20050703-6-CZ-1902.00745 (cit. on p. 68).

S. Kajita, H. Hirukawa, K. Harada, and K. Yokoi (2014). *Introduction to humanoid robotics*. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-54536-8 (cit. on pp. 10, 12).

S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa (2003). "Resolved momentum control: Humanoid motion planning based on the linear and angular momentum". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 1644–1650. DOI: 10.1109/IROS.2003.1248880 (cit. on p. 49).

M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal (2011). "Learning, planning, and control for quadruped locomotion over challenging terrain". In: *International Journal of Robotics Research* 30.2, pp. 236–258. DOI: 10.1177/0278364910388677 (cit. on p. 16).

O. Kanoun, F. Lamiraux, and P.-B. Wieber (2011). "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task". In: *IEEE Transactions on Robotics* 27.4, pp. 785–792. DOI: 10.1109/TRO.2011.2142450 (cit. on p. 51).

N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, Z. Ren, E. M. Hoffman, M. Kamedula, G. Rigano, J. Malzahn, S. Cordasco, P. Guria, A. Margan, and N. Tsagarakis (2019). "CENTAURO: A hybrid locomotion and high power resilient manipulation platform". In: *IEEE Robotics and Automation Letters* 4.2, pp. 1595–1602. DOI: 10.1109/LRA.2019.2896758 (cit. on p. 3).

M. Kelly (2017). "An introduction to trajectory optimization: How to do your own direct collocation". In: *SIAM Review* 59.4, pp. 849–904. DOI: 10.1137/16M1062569 (cit. on pp. 33, 84, 102).

O. Khatib (1987). "A unified approach for motion and force control of robot manipulators: The operational space formulation". In: *IEEE Journal of Robotics and Automation* 3.1, pp. 43–53. DOI: 10.1109/JRA.1987.1087068 (cit. on p. 49).

J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard (2015). "Whole-body model-predictive control applied to the HRP-2 humanoid". In: *Proc. IEEE International Conference on Intelligent*

Robots and Systems, pp. 3346–3351. DOI: 10.1109/IROS.2015.7353843 (cit. on p. 22).

N. Koenig and A. Howard (2004). "Design and use paradigms for Gazebo, an open-source multi-robot simulator". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 2149–2154. DOI: 10.1109/IROS.2004.1389727 (cit. on p. 56).

J. Kolter, M. Rodgers, and A. Ng (2008). "A control architecture for quadruped locomotion over rough terrain". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 811–818. DOI: 10.1109/ROBOT.2008.4543305 (cit. on p. 16).

T. Koolen (2020). "Balance control and locomotion planning for humanoid robots using nonlinear centroidal models". PhD thesis. Massachusetts Institute of Technology. DOI: 1721.1/128291 (cit. on p. 22).

T. Koolen, S. Bertrand, G. Thomas, T. de Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt (2016). "Design of a momentum-based control framework and application to the humanoid robot Atlas". In: *International Journal of Humanoid Robotics* 13.1, p. 1650007. DOI: 10.1142/S0219843616500079 (cit. on p. 54).

T. Koolen, T. de Boer, J. Rebula, A. Goswami, and J. Pratt (2012). "Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models". In: *International Journal of Robotics Research* 31.9, pp. 1094–1113. DOI: 10.1177/0278364912452673 (cit. on pp. 12, 61).

T. Koolen and R. Deits (2019). "Julia for robotics: Simulation and real-time control in a high-level programming language". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 604–611. DOI: 10.1109/ICRA.2019.8793875 (cit. on pp. 90, 112).

E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski (2017). "The DARPA Robotics Challenge finals: Results and perspectives". In: *Journal of Field Robotics* 34.2, pp. 229–240. DOI: 10.1002/rob.21683 (cit. on p. 53).

J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue (2002). "Dynamically-stable motion planning for humanoid robots". In: *Autonomous Robots* 12.1, pp. 105–118. DOI: 10.1023/A:1013219111657 (cit. on p. 17).

S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake (2016). "Optimization-based locomotion planning, estimation, and control design for the Atlas humanoid robot". In: *Autonomous Robots* 40.3, pp. 429–455. DOI: 10.1007/s10514-015-9479-3 (cit. on pp. 17, 54).

S. Kuindersma, F. Permenter, and R. Tedrake (2014). "An efficiently solvable quadratic program for stabilizing dynamic locomotion". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 2589–2594. DOI: 10.1109/ICRA.2014.6907230 (cit. on pp. 53, 66).

S. LaValle (2006). *Planning algorithms*. Cambridge University Press. DOI: 10.1017/CBO9780511546877 (cit. on p. 17).

J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter (2020). "Learning quadrupedal locomotion over challenging terrain". In: *Science Robotics* 5.47. DOI: 10.1126/scirobotics.abc5986 (cit. on p. 26).

S. Lengagne, J. Vaillant, E. Yoshida, and A. Kheddar (2013). "Generation of whole-body optimal dynamic multi-contact motions". In: *International Journal of Robotics Research* 32.9–10, pp. 1104–1119. DOI: 10.1177/0278364913478990 (cit. on p. 17).

H. Li, R. Frei, and P. Wensing (2021). "Model hierarchy predictive control of robotic systems". In: *IEEE Robotics and Automation Letters*. DOI: 10.1109/LRA.2021.3061322 (cit. on p. 126).

H. Li and P. Wensing (2020). "Hybrid systems differential dynamic programming for whole-body motion planning of legged robots". In: *IEEE Robotics and Automation Letters* 5.4, pp. 5448–5455. DOI: 10.1109/LRA.2020.3007475 (cit. on pp. 22, 105).

Q. Li, I. Chatzinikolaidis, Y. Yang, S. Vijayakumar, and Z. Li (2017). "Robust foot placement control for dynamic walking using online parameter estimation". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 165–170. DOI: 10.1109/HUMANOIDS.2017.8239552 (cit. on p. 12).

Z. Li, C. Zhou, H. Dallali, N. Tsagarakis, and D. Caldwell (2014). "Comparison study of two inverted pendulum models for balance recovery". In: *Proc. IEEE*

International Conference on Humanoid Robots, pp. 67–72. DOI: 10.1109/HUMANOIDS.2014.7041339 (cit. on p. 11).

Z. Li, C. Zhou, Q. Zhu, and R. Xiong (2017). "Humanoid balancing behavior featured by underactuated foot motion". In: *IEEE Transactions on Robotics* 33.2, pp. 298–312. DOI: 10.1109/TRO.2016.2629489 (cit. on p. 118).

L. Liao and C. Shoemaker (1992). *Advantages of differential dynamic programming over Newton's method for discrete-time optimal control problems*. Tech. rep. URL: https://hdl.handle.net/1813/5474 (cit. on pp. 35, 105).

D. Liberzon (2012). *Calculus of variations and optimal control theory*. Princeton University Press. DOI: 10.2307/J.CTVCM4G0S (cit. on p. 28).

Y. Lin, L. Righetti, and D. Berenson (2020). "Robust humanoid contact planning with learned zero- and one-step capturability prediction". In: *IEEE Robotics and Automation Letters* 5.2, pp. 2451–2458. DOI: 10.1109/LRA.2020.2972825 (cit. on p. 16).

Z. Manchester, N. Doshi, R. Wood, and S. Kuindersma (2019). "Contact-implicit trajectory optimization using variational integrators". In: *International Journal of Robotics Research* 38.12–13, pp. 1463–1476. DOI: 10.1177/0278364919849235 (cit. on pp. 22, 63, 87, 92, 93, 127).

Z. Manchester and M. Peck (2016). "Quaternion variational integrators for spacecraft dynamics". In: *Journal of Guidance, Control, and Dynamics* 39.1, pp. 69–76. DOI: 10.2514/1.G001176 (cit. on p. 69).

N. Mansard, A. Del Prete, M. Geisert, S. Tonneau, and O. Stasse (2018). "Using a memory of motion to efficiently warm-start a nonlinear predictive controller". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 2986–2993. DOI: 10.1109/ICRA.2018.8463154 (cit. on p. 126).

N. Mansard, O. Khatib, and A. Kheddar (2009). "A unified approach to integrate unilateral constraints in the stack of tasks". In: *IEEE Transactions on Robotics* 25.3, pp. 670–685. DOI: 10.1109/TRO.2009.2020345 (cit. on p. 51).

J. Marsden and M. West (2001). "Discrete mechanics and variational integrators". In: *Acta Numerica* 10, pp. 357–514. DOI: 10.1017/S096249290100006X (cit. on p. 68).

C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard (2020). "Crocoddyl: An efficient and versatile framework for multi-contact optimal control". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 2536–2542. DOI: 10.1109/ICRA40945.2020.9196673 (cit. on pp. 22, 107).

C. Mastalli, I. Havoutis, M. Focchi, D. Caldwell, and C. Semini (2020). "Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control". In: *IEEE Transactions on Robotics* 36.6, pp. 1635–1648. DOI: 10.1109/TRO.2020.3003464 (cit. on p. 16).

MathWorks Inc. (2018). *MATLAB and Simulink*. Version 2018a. Natick, MA, US (cit. on p. 70).

D. Mayne (1966). "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems". In: *International Journal of Control* 3.1, pp. 85–95. DOI: 10.1080/00207176608921369 (cit. on pp. 35, 105).

W. Merkt, V. Ivan, T. Dinev, I. Havoutis, and S. Vijayakumar (2020). *Memory clustering using persistent homology for multimodality- and discontinuity-sensitive learning of optimal control warm-starts*. arXiv: 2010.01024 [cs.RO] (cit. on p. 126).

M. Mistry and L. Righetti (2011). "Operational space control of constrained and underactuated systems". In: *Proc. Robotics: Science and Systems*. DOI: 10.15607/RSS.2011.VII.031 (cit. on p. 52).

K. Mombaur (2009). "Using optimization to create self-stable human-like running". In: *Robotica* 27.3, pp. 321–330. DOI: 10.1017/S0263574708004724 (cit. on pp. 20, 84).

I. Mordatch, M. de Lasa, and A. Hertzmann (2010). "Robust physics-based locomotion using low-dimensional planning". In: *ACM Transactions on Graphics* 29.4. DOI: 10.1145/1778765.1778808 (cit. on p. 13).

I. Mordatch, K. Lowrey, and E. Todorov (2015). "Ensemble-CIO: Full-body dynamic motion planning that transfers to physical humanoids". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 5307–5314. DOI: 10.1109/IROS.2015.7354126 (cit. on p. 21).

I. Mordatch, Z. Popović, and E. Todorov (2012). "Contact-invariant optimization for hand manipulation". In: *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation*. Ed. by J. Lee and P. Kry. DOI: 10.2312/SCA/SCA12/137-144 (cit. on p. 21).

I. Mordatch, E. Todorov, and Z. Popović (2012). "Discovery of complex behaviors through contact-invariant optimization". In: *ACM Transactions on Graphics* 31.4, 43:1–43:8. DOI: 10.1145/2185520.2185539 (cit. on pp. 21, 63, 85).

J. J. Moreau (2011). "On unilateral constraints, friction and plasticity". In: *New Variational Techniques in Mathematical Physics*. Ed. by G. Capriz and G. Stampacchia. C.I.M.E. Summer Schools. Springer Berlin Heidelberg, pp. 171–322. DOI: 10.1007/978-3-642-10960-7_7 (cit. on p. 43).

F. Moro and L. Sentis (2019). "Whole-body control of humanoid robots". In: *Humanoid Robotics: A Reference*. Ed. by A. Goswami and P. Vadakkepat. Springer Netherlands, pp. 1161–1183. DOI: 10.1007/978-94-007-6046-2_51 (cit. on p. 49).

D. Murray and S. Yakowitz (1984). "Differential dynamic programming and Newton's method for discrete optimal control problems". In: *Journal of Optimization Theory and Applications* 43.3, pp. 395–414. DOI: 10.1007/BF00934463 (cit. on p. 35).

K. Murty (1997). *Linear complementarity, linear and nonlinear programming*. Internet Edition. Heldermann Verlag. URL: http://www-personal.umich.edu/~murty/books/linear_complementarity_webbook/ (cit. on p. 44).

Y. Nakamura, H. Hanafusa, and T. Yoshikawa (1987). "Task-priority based redundancy control of robot manipulators". In: *International Journal of Robotics Research* 6.2, pp. 3–15. DOI: 10.1177/027836498700600201 (cit. on p. 50).

M. Neunert, F. Farshidian, A. Winkler, and J. Buchli (2017). "Trajectory optimization through contacts and automatic gait discovery for quadrupeds". In: *IEEE Robotics and Automation Letters* 2.3, pp. 1502–1509. DOI: 10.1109/LRA.2017.2665685 (cit. on pp. 22, 84, 85, 96, 103, 106).

J. Nocedal and S. Wright (2006). *Numerical optimization*. Springer-Verlag. DOI: 10.1007/978-0-387-40065-5 (cit. on pp. 32, 88).

A. Önol, R. Corcodel, P. Long, and T. Padır (2020). "Tuning-free contact-implicit trajectory optimization". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1183–1189. DOI: 10.1109/ICRA40945.2020.9196805 (cit. on p. 21).

A. Önol, P. Long, and T. Padır (2019). "Contact-implicit trajectory optimization based on a variable smooth contact model and successive convexification". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 2447–2453. DOI: 10.1109/ICRA.2019.8794250 (cit. on p. 85).

D. Orin, A. Goswami, and S.-H. Lee (2013). "Centroidal dynamics of a humanoid robot". In: *Autonomous Robots* 35.2, pp. 161–176. DOI: 10.1007/s10514-013-9341-4 (cit. on pp. 17, 54, 64).

A. Pandey (2019). "3D surface approximation from point clouds". Undergraduate Dissertation. University of Edinburgh. URL: https://groups.inf.ed.ac.uk/advr/papers/3D_Surface_Approximation_from_Point_Clouds.pdf (cit. on p. 71).

M. Pandy (2003). "Simple and complex models for studying muscle function in walking". In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 358.1437, pp. 1501–1509. DOI: 10.1098/rstb.2003.1338 (cit. on p. 11).

A. Patel, S. Shield, S. Kazi, A. Johnson, and L. Biegler (2019). "Contact-implicit trajectory optimization using orthogonal collocation". In: *IEEE Robotics and Automation Letters* 4.2, pp. 2242–2249. DOI: 10.1109/LRA.2019.2900840 (cit. on pp. 22, 100, 120, 127).

N. Perrin, O. Stasse, L. Baudouin, F. Lamiraux, and E. Yoshida (2012). "Fast humanoid robot collision-free footstep planning using swept volume approximations". In: *IEEE Transactions on Robotics* 28.2, pp. 427–439. DOI: 10.1109/TRO.2011.2172152 (cit. on p. 17).

R. Pfeifer and J. Bongard (2006). *How the body shapes the way we think: A new view of intelligence*. The MIT Press (cit. on p. 2).

B. Ponton, A. Herzog, S. Schaal, and L. Righetti (2016). "A convex model of humanoid momentum dynamics for multi-contact motion generation". In:

Proc. IEEE International Conference on Humanoid Robots, pp. 842–849. DOI: 10.1109/HUMANOIDS.2016.7803371 (cit. on p. 16).

L. Pontryagin (1987). *Mathematical theory of optimal processes*. Routledge. DOI: 10.1201/9780203749319 (cit. on p. 28).

M. Popović, A. Goswami, and H. Herr (2005). "Ground reference points in legged locomotion: Definitions, biological trajectories and control implications". In: *International Journal of Robotics Research* 24.12, pp. 1013–1032. DOI: 10.1177/0278364905058363 (cit. on p. 61).

M. Posa, C. Cantu, and R. Tedrake (2014). "A direct method for trajectory optimization of rigid bodies through contact". In: *International Journal of Robotics Research* 33.1, pp. 69–81. DOI: 10.1177/0278364913506757 (cit. on pp. 20, 21, 63, 76, 85, 92, 120).

M. Posa, S. Kuindersma, and R. Tedrake (2016). "Optimization and stabilization of trajectories for constrained dynamical systems". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1366–1373. DOI: 10.1109/ICRA.2016.7487270 (cit. on pp. 6, 61, 127).

W. Powell (2009). "What you should know about approximate dynamic programming". In: *Naval Research Logistics* 56.3, pp. 239–249. DOI: 10.1002/nav.20347 (cit. on p. 126).

J. Pratt, J. Carff, S. Drakunov, and A. Goswami (2006). "Capture point: A step toward humanoid push recovery". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 200–207. DOI: 10.1109/ICHR.2006.321385 (cit. on p. 12).

N. Radford, P. Strawser, K. Hambuchen, J. Mehling, W. Verdeyen, S. Donnan, J. Holley, J. Sanchez, V. Nguyen, L. Bridgwater, R. Berka, R. Ambrose, M. Myles Markee, N. Fraser-Chanpong, C. McQuin, J. Yamokoski, S. Hart, R. Guo, A. Parsons, B. Wightman, P. Dinh, B. Ames, C. Blakely, C. Edmondson, B. Sommers, R. Rea, C. Tobler, H. Bibby, B. Howard, L. Niu, A. Lee, M. Conover, L. Truong, R. Reed, D. Chesney, R. Platt Jr, G. Johnson, C.-L. Fok, N. Paine, L. Sentis, E. Cousineau, R. Sinnet, J. Lack, M. Powell, B. Morris, A. Ames, and J. Akinyode (2015). "Valkyrie: NASA's first bipedal humanoid robot". In:

Journal of Field Robotics 32.3, pp. 397–419. DOI: 10.1002/rob.21560 (cit. on pp. 3, 56).

A. Radulescu, I. Havoutis, D. Caldwell, and C. Semini (2017). "Whole-body trajectory optimization for non-periodic dynamic motions on quadrupedal systems". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 5302–5307. DOI: 10.1109/ICRA.2017.7989623 (cit. on p. 84).

M. Raibert (1986). *Legged robots that balance*. MIT Press (cit. on p. 14).

S. Redfield (2019). "A definition for robotics as an academic discipline". In: *Nature Machine Intelligence* 1.6, pp. 263–264. DOI: 10.1038/s42256-019-0064-x (cit. on p. 2).

J. Revels, M. Lubin, and T. Papamarkou (2016). "Forward-mode automatic differentiation in Julia". In: arXiv: 1607.07892 [cs.MS] (cit. on p. 112).

Roboti LLC (2020). *MuJoCo computation*. URL: http://www.mujoco.org/book/computation.html (cit. on pp. 109, 116).

R. T. Rockafellar (1993). "Lagrange multipliers and optimality". In: *SIAM Review* 35.2, pp. 183–238. DOI: 10.1137/1035044 (cit. on p. 30).

L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Souères, and J.-Y. Fourquet (2013). "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints". In: *IEEE Transactions on Robotics* 29.2, pp. 346–362. DOI: 10.1109/TRO.2012.2234351 (cit. on p. 53).

J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel (2013). "Finding locally optimal, collision-free trajectories with sequential convex optimization". In: *Proc. Robotics: Science and Systems*. DOI: 10.15607/RSS.2013.IX.031 (cit. on p. 19).

G. Schultz and K. Mombaur (2010). "Modeling and optimal control of human-like running". In: *IEEE/ASME Transactions on Mechatronics* 15.5, pp. 783–792. DOI: 10.1109/TMECH.2009.2035112 (cit. on p. 20).

X. Shen and M. Leok (2017). *Lie group variational integrators for rigid body problems using quaternions*. arXiv: 1705.04404 [math.NA] (cit. on pp. 68, 69).

A. Shkolnik, M. Levashov, I. Manchester, and R. Tedrake (2011). "Bounding on rough terrain with the LittleDog robot". In: *International Journal of Robotics Research* 30.2, pp. 192–215. DOI: 10.1177/0278364910388315 (cit. on p. 17).

M. Srinivasan and A. Ruina (2006). "Computer optimization of a minimal biped model discovers walking and running". In: *Nature* 439.7072, pp. 72–75. DOI: 10.1038/nature04113 (cit. on p. 13).

O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, J. Mirabel, A. Del Prete, P. Souères, N. Mansard, F. Lamiraux, J. Laumond, L. Marchionni, H. Tome, and F. Ferro (2017). "TALOS: A new humanoid research platform targeted for industrial applications". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 689–695. DOI: 10.1109/HUMANOIDS.2017.8246947 (cit. on p. 3).

D. Stewart (2000). "Rigid-body dynamics with friction and impact". In: *SIAM Review* 42.1, pp. 3–39. DOI: 10.1137/S0036144599360110 (cit. on p. 40).

D. Stewart and J. Trinkle (2000). "An implicit time-stepping scheme for rigid body dynamics with Coulomb friction". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 162–169. DOI: 10.1109/ROBOT.2000.844054 (cit. on p. 40).

G. Strang (2007). *Computational science and engineering*. Wellesley-Cambridge Press (cit. on p. 108).

A. Tasora and M. Anitescu (2011). "A matrix-free cone complementarity approach for solving large-scale, nonsmooth, rigid body dynamics". In: *Computer Methods in Applied Mechanics and Engineering* 200.5, pp. 439–453. DOI: 10.1016/j.cma.2010.06.030 (cit. on p. 46).

Y. Tassa (2011). "Theory and implementation of biomimetic motor controllers". PhD thesis. Hebrew University of Jerusalem (cit. on pp. 38, 39).

Y. Tassa, T. Erez, and E. Todorov (2012). "Synthesis and stabilization of complex behaviors through online trajectory optimization". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 4906–4913. DOI: 10.1109/IROS.2012.6386025 (cit. on pp. 22, 39, 103, 106, 110, 120).

Y. Tassa, N. Mansard, and E. Todorov (2014). "Control-limited differential dynamic programming". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1168–1175. DOI: 10.1109/ICRA.2014.6907001 (cit. on p. 105).

G. Terzakis, M. Lourakis, and D. Ait-Boudaoud (2018). "Modified Rodrigues parameters: An efficient representation of orientation in 3D vision and graphics". In: *Journal of Mathematical Imaging and Vision* 60.3, pp. 422–442. DOI: 10.1007/s10851-017-0765-x (cit. on p. 90).

E. Todorov (2011). "A convex, smooth and invertible contact model for trajectory optimization". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 1071–1076. DOI: 10.1109/ICRA.2011.5979814 (cit. on p. 45).

E. Todorov (2014). "Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in MuJoCo". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 6054–6061. DOI: 10.1109/ICRA.2014.6907751 (cit. on pp. 41, 45, 46, 87, 104, 110).

E. Todorov (2018). "Goal directed dynamics". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 2994–3000. DOI: 10.1109/ICRA.2018.8462904 (cit. on p. 102).

E. Todorov and W. Li (2005). "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems". In: *Proc. American Control Conference*, pp. 300–306. DOI: 10.1109/ACC.2005.1469949 (cit. on p. 105).

S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha, and N. Mansard (2018). "An efficient acyclic contact planner for multiped robots". In: *IEEE Transactions on Robotics* 34.3, pp. 586–601. DOI: 10.1109/TRO.2018.2819658 (cit. on pp. 18, 62, 83).

S. Tonneau, D. Song, P. Fernbach, N. Mansard, M. Taïx, and A. Del Prete (2020). "SL1M: Sparse L1-norm minimization for contact planning on uneven terrain". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 6604–6610. DOI: 10.1109/ICRA40945.2020.9197371 (cit. on p. 16).

M. Toussaint (2015). "Logic-geometric programming: An optimization-based approach to combined task and motion planning". In: *Proc. International Joint*

Conference on Artificial Intelligence, pp. 1930–1936. URL: https://www.ijcai.org/Abstract/15/274 (cit. on p. 126).

M. Toussaint, K. Allen, K. Smith, and J. Tenenbaum (2018). "Differentiable physics and stable modes for tool-use and manipulation planning". In: *Proc. Robotics: Science and Systems*. DOI: 10.15607/RSS.2018.XIV.044 (cit. on p. 24).

M. Toussaint, J.-S. Ha, and D. Driess (2020). "Describing physics for physical reasoning: Force-based sequential manipulation planning". In: *IEEE Robotics and Automation Letters* 5.4, pp. 6209–6216. DOI: 10.1109/LRA.2020.3010462 (cit. on pp. 84, 87).

F. Udwadia and R. Kalaba (1996). *Analytical dynamics: A new approach*. Cambridge University Press. DOI: 10.1017/CBO9780511665479 (cit. on p. 50).

A. Valenzuela (2016). "Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain". ScD thesis. Massachusetts Institute of Technology. DOI: 1721.1/103432 (cit. on p. 22).

M. Vukobratović and B. Borovak (2004). "Zero-moment point—thirty five years of its life". In: *International Journal of Humanoid Robotics* 1.1, pp. 157–173. DOI: 10.1142/S0219843604000083 (cit. on pp. 10, 61).

A. Wächter and L. Biegler (2006). "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical Programming* 106.1, pp. 25–57. DOI: 10.1007/s10107-004-0559-y (cit. on p. 90).

K. Wampler and Z. Popović (2009). "Optimal gait and form for animal locomotion". In: *ACM Transactions on Graphics* 28.3. DOI: 10.1145/1531326.1531366 (cit. on p. 19).

J. Wang, I. Chatzinikolaidis, C. Mastalli, W. Wolfslag, G. Xin, S. Tonneau, and S. Vijayakumar (2020). "Automatic gait pattern selection for legged robots". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 3990–3997. DOI: 10.1109/IROS45743.2020.9340789 (cit. on p. 102).

P. Wensing and D. Orin (2013). "High-speed humanoid running through control with a 3D-SLIP model". In: *Proc. IEEE International Conference on Intelligent*

Robots and Systems, pp. 5134–5140. DOI: 10.1109/IROS.2013.6697099 (cit. on p. 13).

P.-B. Wieber (2006). "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations". In: *Proc. IEEE International Conference on Humanoid Robots*, pp. 137–142. DOI: 10.1109/ICHR.2006.321375 (cit. on p. 16).

P.-B. Wieber, R. Tedrake, and S. Kuindersma (2016). "Modeling and control of legged robots". In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer International Publishing, pp. 1203–1234. DOI: 10.1007/978-3-319-32552-1_48 (cit. on p. 15).

D. Wight, E. Kubica, and D. Wang (2007). "Introduction of the foot placement estimator: A dynamic measure of balance for bipedal robotics". In: *Journal of Computational and Nonlinear Dynamics* 3.1. DOI: 10.1115/1.2815334 (cit. on p. 12).

A. Winkler (2017). *Xpp - A collection of ROS packages for the visualization of legged robots*. DOI: 10.5281/zenodo.1037901 (cit. on p. 80).

A. Winkler (2018). "Optimization-based motion planning for legged robots". PhD thesis. ETH Zurich. DOI: 10.3929/ethz-b-000272432 (cit. on p. 17).

A. Winkler, C. Bellicoso, M. Hutter, and J. Buchli (2018). "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization". In: *IEEE Robotics and Automation Letters* 3.3, pp. 1560–1567. DOI: 10.1109/LRA.2018.2798285 (cit. on pp. 63, 67, 76–78).

A. Winkler, F. Farshidian, D. Pardo, M. Neunert, and J. Buchli (2017). "Fast trajectory optimization for legged robots using vertex-based ZMP constraints". In: *IEEE Robotics and Automation Letters* 2.4, pp. 2201–2208. DOI: 10.1109/LRA.2017.2723931 (cit. on p. 11).

A. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. Caldwell, and C. Semini (2015). "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 5148–5154. DOI: 10.1109/ICRA.2015.7139916 (cit. on p. 16).

W. Wolfslag, C. McGreavy, G. Xin, C. Tiseo, S. Vijayakumar, and Z. Li (2020). "Optimisation of body-ground contact for augmenting whole-body loco-manipulation of quadruped robots". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 3694–3701. DOI: 10.1109/IROS45743.2020.9341498 (cit. on p. 102).

W. Xi, Y. Yesilevskiy, and C. D. Remy (2016). "Selecting gaits for economical locomotion of legged robots". In: *International Journal of Robotics Research* 35.9, pp. 1140–1154. DOI: 10.1177/0278364915612572 (cit. on p. 22).

Z. Xie, K. Liu, and K. Hauser (2017). "Differential dynamic programming with nonlinear constraints". In: *Proc. IEEE International Conference on Robotics and Automation*, pp. 695–702. DOI: 10.1109/ICRA.2017.7989086 (cit. on p. 105).

S. Yakowitz (1989). "Algorithms and computational techniques in differential dynamic programming". In: *Control and Dynamic Systems*. Ed. by C. Leondes. Advances in Aerospace Systems Dynamics and Control Systems. Academic Press. DOI: 10.1016/B978-0-12-012731-3.50008-1 (cit. on p. 38).

C. Yang, K. Yuan, S. Heng, T. Komura, and Z. Li (2020). "Learning natural locomotion behaviors for humanoid robots using human bias". In: *IEEE Robotics and Automation Letters* 5.2, pp. 2610–2617. DOI: 10.1109/LRA.2020.2972879 (cit. on pp. 26, 84).

C. Yang, K. Yuan, Q. Zhu, W. Yu, and Z. Li (2020). "Multi-expert learning of adaptive legged locomotion". In: *Science Robotics* 5.49. DOI: 10.1126/scirobotics.abb2174 (cit. on p. 26).

Y. You, Z. Li, D. Caldwell, and N. Tsagarakis (2015). "From one-legged hopping to bipedal running and walking: A unified foot placement control based on regression analysis". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 4492–4497. DOI: 10.1109/IROS.2015.7354015 (cit. on p. 15).

K. Yuan, I. Chatzinikolaidis, and Z. Li (2019). "Bayesian optimization for whole-body control of high-degree-of-freedom robots through reduction of dimensionality". In: *IEEE Robotics and Automation Letters* 4.3, pp. 2268–2275. DOI: 10.1109/LRA.2019.2901308 (cit. on pp. 57, 102).

K. Yuan and Z. Li (2018). "An improved formulation for model predictive control of legged robots for gait planning and feedback control". In: *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 8535–8542. DOI: 10.1109/IROS.2018.8594309 (cit. on p. 61).

K. Yunt and C. Glocker (2006). "Trajectory optimization of mechanical hybrid systems using SUMT". In: *IEEE International Workshop on Advanced Motion Control*, pp. 665–671. DOI: 10.1109/AMC.2006.1631739 (cit. on pp. 21, 84).

A. Zhao, J. Xu, M. Konaković-Luković, J. Hughes, A. Spielberg, D. Rus, and W. Matusik (2020). "RoboGrammar: Graph grammar for terrain-optimized robot design". In: *ACM Transactions on Graphics* 39.6. DOI: 10.1145/3414685.3417831 (cit. on p. 14).

C. Zhou, X. Wang, Z. Li, and N. Tsagarakis (2017). "Overview of gait synthesis for the humanoid COMAN". In: *Journal of Bionic Engineering* 14.1, pp. 15–25. DOI: 10.1016/S1672-6529(16)60373-6 (cit. on p. 61).

D. Zimmermann, S. Coros, Y. Ye, R. Sumner, and M. Gross (2015). "Hierarchical planning and control for complex motor tasks". In: *Proc. ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pp. 73–81. DOI: 10.1145/2786784.2786795 (cit. on pp. 16, 126).

S. Zimmermann, G. Hakimifard, M. Zamora, R. Poranne, and S. Coros (2020). "A multi-level optimization framework for simultaneous grasping and motion planning". In: *IEEE Robotics and Automation Letters* 5.2, pp. 2966–2972. DOI: 10.1109/LRA.2020.2974684 (cit. on p. 19).

S. Zimmermann, R. Poranne, J. Bern, and S. Coros (2019). "PuppetMaster: Robotic animation of marionettes". In: *ACM Transactions on Graphics* 38.4, 103:1–103:11. DOI: 10.1145/3306346.3323003 (cit. on p. 107).

M. Zucker, N. Ratliff, A. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. Bagnell, and S. Srinivasa (2013). "CHOMP: Covariant Hamiltonian optimization for motion planning". In: *International Journal of Robotics Research* 32.9–10, pp. 1164–1193. DOI: 10.1177/0278364913488805 (cit. on p. 17).