



Vivian Suzano Medeiros

Trajectory Optimization for Hybrid Wheeled-Legged Robots in Challenging Terrain

Tese de Doutorado

Thesis presented to the Programa de Pós-graduação em Engenharia Mecânica, do Departamento de Engenharia Mecânica da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Mecânica.

Advisor: Prof. Marco Antonio Meggiolaro

Rio de Janeiro
July 2020



Vivian Suzano Medeiros

Trajectory Optimization for Hybrid Wheeled-Legged Robots in Challenging Terrain

Thesis presented to the Programa de Pós-graduação em Engenharia Mecânica da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Mecânica. Approved by the Examination Committee:

Prof. Marco Antonio Meggiolaro

Advisor

Departamento de Engenharia Mecânica – PUC-Rio

Prof. Mauro Speranza Neto

Departamento de Engenharia Mecânica – PUC-Rio

Prof. Helon Vicente Hultmann Ayala

Departamento de Engenharia Mecânica – PUC-Rio

Dr. Armando Morado Ferreira

Instituto Militar de Engenharia (IME)

Prof. Luciano Luporini Menegaldo

Universidade Federal do Rio de Janeiro (UFRJ)

Rio de Janeiro, July the 29th, 2020

All rights reserved.

Vivian Suzano Medeiros

Holds a master's degree in Mechanical Engineering (2015) and a bachelor's degree in Control and Automation Engineering (2012), both from the Pontifical Catholic University of Rio de Janeiro (PUC-Rio). She is currently a researcher at Center for Research on Inspection Technology (CPTI/PUC-Rio) and an assistant professor of Control Systems in the Mechanical Engineering Department of PUC-Rio. She has experience in the area of Industrial Electronics, Control Systems and Rigid Body Dynamics. She currently works in research in the field of Non-Destructive Testing and Risers Inspection, Embedded Systems and Control, and Mobile Robots for Rough Terrain.

Bibliographic data

Medeiros, Vivian Suzano

Trajectory Optimization for Hybrid Wheeled-Legged Robots in Challenging Terrain / Vivian Suzano Medeiros; advisor: Marco Antonio Meggiolaro. – 2020.

135 f: il. color. ; 30 cm

Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Mecânica, 2020.

Inclui bibliografia

1. Engenharia Mecânica – Teses. 2. Robôs híbridos. 3. Terreno acidentado. 4. Planejamento de movimento. 5. Controle ótimo. 6. Otimização de trajetória. I. Meggiolaro, Marco Antonio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Mecânica. III. Título.

CDD: 621

To my parents, that are no longer with me,
but I keep them alive in my heart.

Acknowledgments

First I would like to thank my family for all the support given through the development of this research. My sister Viviane for always believing in me, my sister Vitória, for keeping me company during thousands of sleepless nights, and my boyfriend Rodrigo, for almost 10 years of companionship, dedication, and encouragement in moments of doubt. A very special thanks to my father, the person that I most admired, who has always encouraged me to improve myself and pursue higher education.

To my advisor, prof. Marco Antonio Meggiolaro, for the guidance and continuous support on the development of this thesis. My sincere thanks also goes to professors Roland Siegwart and Marco Hutter, that accepted me as a visiting Ph.D. student at ETH Zürich and gave me access to their laboratory and research facilities. The possibility of working with the ANYmal robot was essential for my doctorate and it was an incredible opportunity to be in contact with state-of-the-art development in mobile robotics. I wish to thank all the staff and Ph.D. students from the Autonomous Systems Lab and the Robotic Systems Lab, that have welcomed and supported me during my time as a researcher there. In particular, the Ph.D. students Marko Bjelonic and Edo Jelavic for the important contributions to the development of this work.

To my fellow colleagues from the LabRob (Robotics Lab) and the CPTI (Research Center for Pipeline Inspection) at PUC-Rio, for all the support, the conversations, and the pieces of advice during my Ph.D. and, of course, for the unlimited coffee supply. Also, to all my dear friends from college and graduate school, for the happy moments and constant encouragement during these long 4 years of study and dedication.

To the professors and employees of the Mechanical Engineering Department and the Electrical Engineering Department of PUC-Rio, for the quality of teaching and excellent infrastructure, essential for the development of this work.

Finally, to the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and to Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for the scholarship and financial support to this research. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Medeiros, Vivian Suzano; Meggiolaro, Marco Antonio (Advisor). **Trajectory Optimization for Hybrid Wheeled-Legged Robots in Challenging Terrain**. Rio de Janeiro, 2020. 135p. Tese de Doutorado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Wheeled-legged robots are an attractive solution for versatile locomotion in challenging terrain. They combine the speed and efficiency of wheels with the ability of legs to traverse challenging terrain. In general, the challenges with wheeled-legged locomotion involve trajectory generation and motion control for trajectory tracking. This thesis focuses in particular on the trajectory optimization task for wheeled-legged robots navigating in challenging terrain. For this, a motion planning framework is proposed that optimizes over the robot's base position and orientation, and the wheels' positions and contact forces in a single planning problem, taking into account the terrain information and the robot dynamics. The robot is modeled as a single rigid-body, which allows to plan complex motions for long time horizons and still keep a low computational complexity to solve the optimization quickly. The knowledge of the terrain map allows the optimizer to generate feasible motions for obstacle negotiation in a dynamic manner, at higher speeds. Such motions cannot be discovered without taking into account the terrain information. Two different formulations allow for either purely driving motions, where obstacle negotiation is enabled by the legs, or hybrid driving-walking motions, which are able to overcome discontinuities in the terrain profile. The optimization is formulated as a Nonlinear Programming Problem (NLP) and the reference motions are tracked by a hierarchical whole-body controller that computes the torque actuation commands for the robot. The trajectories are verified on the quadrupedal robot ANYmal equipped with non-steerable torque-controlled wheels in simulations and experimental tests. The proposed trajectory optimization framework enables wheeled-legged robots to navigate over challenging terrain, e.g., steps, slopes, stairs, while negotiating these obstacles with dynamic motions.

Keywords

Wheeled-legged robots; Challenging terrain; Motion planning; Optimal control; Trajectory optimization; Hybrid locomotion.

Resumo

Medeiros, Vivian Suzano; Meggiolaro, Marco Antonio. **Otimização de Trajetórias para Robôs Híbridos com Pernas e Rodas em Terrenos Acidentados**. Rio de Janeiro, 2020. 135p. Tese de Doutorado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Robôs híbridos equipados com pernas e rodas são uma solução promissora para uma locomoção versátil em terrenos acidentados. Eles combinam a velocidade e a eficiência das rodas com a capacidade das pernas de atravessar terrenos com obstáculos. Em geral, os desafios em locomoção para robôs híbridos envolvem planejamento de trajetória e sistemas de controle para o rastreamento da trajetória planejada. Esta tese se concentra, em particular, na tarefa de otimização de trajetória para robôs híbridos que navegam em terrenos acidentados. Para isso, propõe-se um algoritmo de planejamento que otimiza a posição e a orientação da base do robô e as posições e forças de contato nas rodas em uma formulação única, levando em consideração as informações do terreno e a dinâmica do robô. O robô é modelado como um único corpo rígido com massa e inércia concentrada no centro de massa, o que permite planejar movimentos complexos por longos horizontes de tempo e ainda manter uma baixa complexidade computacional para resolver a otimização de forma mais eficiente. O conhecimento do mapa do terreno permite que a otimização gere trajetórias para negociação de obstáculos de maneira dinâmica, em velocidades mais altas. Tais movimentos não podem ser gerados sem levar em consideração as informações do terreno. Duas formulações diferentes são apresentadas, uma que permite movimentos somente com as rodas, onde a negociação de obstáculos é permitida pelas pernas, e outra focada em movimentos híbridos dando passos e movendo as rodas, capazes de lidar com discontinuidades no perfil do terreno. A otimização é formulada como um NLP e as trajetórias obtidas são rastreadas por um controlador hierárquico que computa os comandos de atuação de torque para as juntas e as rodas do robô. As trajetórias são verificadas no robô quadrúpede ANYmal equipado com rodas não esterçáveis controladas por torque, em simulações e testes experimentais. O algoritmo proposto de otimização de trajetória permite que robôs com pernas e rodas naveguem por terrenos complexos, contendo, por exemplo, degraus, declives e escadas, enquanto negociam esses obstáculos com movimentos dinâmicos.

Palavras-chave

Robôs híbridos; Terreno acidentado; Planejamento de movimento; Controle ótimo; Otimização de trajetória; Locomoção híbrida.

Table of contents

1	Introduction	20
1.1	Objectives	23
1.2	Related Work	25
1.2.1	ANYmal on Wheels	35
1.3	Contributions	38
1.4	Thesis Organization	40
2	Modeling of Wheeled-Legged Robots	41
2.1	Full Rigid Body Dynamics	41
2.1.1	Equations of Motion	41
2.1.2	Contact Model	43
2.1.3	Nonholonomic Rolling Constraint	44
2.2	Single Rigid Body Dynamics	46
2.3	Trajectory Optimization	47
2.3.1	Transcription Methods	49
3	2D Trajectory Optimization	50
3.1	2D Robot Model	50
3.2	Problem Formulation	52
3.2.1	Constraints	55
3.2.2	Objective Function	57
3.3	Results	58
3.3.1	Implementation	58
3.3.2	Suitable Cost Functions	59
3.3.3	Trajectories	62
3.4	Simulations	63
3.4.1	From 2D to 3D motions	66
3.4.2	Extended Whole-Body Controller	67
3.4.3	Simulation Results	68
3.5	Conclusions	73
4	3D Trajectory Optimization for Driving Motions	75
4.1	TO Formulation for Driving Motions	75
4.1.1	Optimization Variables Parametrization	77
4.1.2	Dynamic and Kinematic Constraints	78
4.1.3	Wheels' Contact Constraints	79
4.1.4	Stability Constraint	83
4.2	Results	85
4.2.1	Implementation	85
4.2.2	Initial Guess	86
4.2.3	Simulations	87
4.2.4	Experimental Results	92
4.3	Conclusions	94

5	3D Trajectory Optimization for Hybrid Motions	97
5.1	TO formulation for Hybrid Motions	98
5.2	Optimization Variables Parametrization	99
5.2.1	Contact Schedule	101
5.3	NLP Constraints	102
5.3.1	Force Constraints	103
5.3.2	Motion Constraints	105
5.4	Results	106
5.4.1	Implementation	108
5.4.2	Simulations	109
5.4.2.1	Flat Terrain	109
5.4.2.2	Gap	113
5.4.2.3	Floating Step	116
5.5	Conclusions	118
6	Conclusions and Future Work	120
6.1	Future Work	122
6.2	Publications	123
	Bibliography	124
A	Hermite Parametrization	135

List of figures

Figure 1.1	Rovers for exploration at the surface of Mars.	21
(a)	Sojourner rover.	21
(b)	Perseverance rover.	21
Figure 1.2	The main robot prototypes that competed in the ARGOS challenge.	21
Figure 1.3	Different robot designs for the current and past DARPA Challenges.	21
Figure 1.4	Different verification methods for the feasibility of the motion plans. (a) Pure visualization with RViz; (b) Physical simulation with Gazebo; (c) Experiments on real robot.	25
Figure 1.5	Articulated wheeled robots: (a) The Sample Return Rover (SRR), developed by the <i>Jet Propulsion Lab</i> for extra-planetary exploration; (b) The PAW (Platform for Ambulating Wheels) robot; and (c) The Octopus robot, developed at EPFL, Lausanne, Switzerland.	27
Figure 1.6	Articulated wheeled robots: (a) The Hylos robot, developed by the <i>Institut des Systèmes Intelligents et de Robotique</i> (ISIR) of the Université Pierre et Marie CURIE, Paris, France; (b) The Hylos robot navigating on asymmetric inclines; and (c) The MHT (Micro-Hydraulic Toolkit) robot, developed by the Defense Research and Development Canada Suffield Research Centre.	27
Figure 1.7	On the left, the Hylos2 robot. On the right, the robot crossing a step with 60° slope employing a quasi-static obstacle crossing approach.	28
Figure 1.8	Rovers with active suspension: (a) The MAMMOTH rover, developed by the Australian Centre for Field Robotics (ACFR), University of Sydney, Australia; (b) The SherpaTT rover, from the University of Bremen, Germany; (c) The Compiros robot, developed by the Université Pierre et Marie CURIE, Paris, France.	29
Figure 1.9	Hybrid robots where driving and stepping phase are treated as separate behaviors: (a) The wheeled humanoid robot DRC-HUBO+ in its two configurations: walking mode (left) and driving mode (right); (b) The Momaro robot, developed by the University of Bonn, Germany; (c) The CENTAURO robot, developed by the <i>Istituto Italiano di Tecnologia</i> (ITT).	30
Figure 1.10	The Menzi Muck excavator, on the left. On the right, simulations with the excavator overcoming a step-like obstacle using the wheeled legs and the shovel simultaneously.	31
Figure 1.11	Two wheeled-legged robots capable of driving and jumping over obstacles.	32
(a)	The Handle robot, from Boston Dynamics.	32
(b)	The Ascento robot, from ETH Zürich, Switzerland.	32

- Figure 1.12 Some of the wheeled-legged robots created by the *Skaterbots* framework, which are capable to perform different types of motions in flat terrain. 32
- Figure 1.13 The legged Robosimian robot in its default position (on the left) and the robot equipped with passive wheels for skating locomotion (on the right). 33
- Figure 1.14 The wheeled-legged robot Pholus, a centaur-like robot capable of hybrid locomotion. 34
- Figure 1.15 On the left, the position of the actuated joints on ANYmal's legs and, on the right, the blue sphere depicts the reachable workspace of one wheel. 35
- Figure 1.16 The ANYmal with wheels negotiating slopes and small steps with a reactive controller. Reprinted from (Bjelonic et al., 2019). 36
- Figure 1.17 The ANYmal with wheels navigating in an underground environment at the DARPA Subterranean Challenge. Reprinted from (Bjelonic et al., 2020). 37
- Figure 1.18 List of prioritized tasks used in the WBC. Tasks in bold are specific for wheeled-legged robots. Reprinted from (Bjelonic et al., 2019). 37
- Figure 1.19 Attempts to overcome a step-like obstacle with a 65° slope and 0.2 m height in simulations with purely reactive controller, performing driving motions (right) and hybrid motions (left). 38
- Figure 1.20 The 2.5D height map of the terrain generated from stereo camera data using the ROS package *elevation_mapping* (Fankhauser et al., 2014). 38
- Figure 2.1 Representation of the actuation joints of a wheeled quadrupedal robot, modeled as a floating base B to which the legs and wheels are attached. The coordinate frames are the fixed inertial frame I and the frame B , attached to un-actuated base. Figure adapted from (Hutter et al., 2017). 42
- Figure 2.2 The coordinate frames of the robot's wheel. 45
- Figure 2.3 Single rigid body representation of a quadrupedal robot. The gray overlaid model of the ANYmal robot with wheels is only for visualization and it is not included in the dynamics, all the system's mass is located at the robot's CoM. 46
- Figure 3.1 Planar model of the ANYmal robot traversing rough terrain. 51
- Figure 3.2 Contact forces on each wheel, considering a frame C_i located at the wheel's contact point with the terrain. The axis \mathbf{c}_z is aligned with the terrain normal and \mathbf{c}_x is aligned with the rolling direction of the wheel. 52
- Figure 3.3 Illustration of the discretization points (nodes) for the robot's trajectory. 53
- Figure 3.4 Decision variables and constraints of the two-dimensional TO formulation. 54

- Figure 3.5 Illustration of the stability measure for the planar case. 56
- Figure 3.6 2D optimization results for the robot driving on flat terrain with different cost functions. The red lines are the contact forces, the green lines are the decomposed normal and traction forces, the back full lines are the gravity force and the black dotted lines indicate the direction of the CoM's acceleration. The pink line is the total force acting on the robot's CoM and the yellow lines represent the tipover axis normals, the limits for stability. 60
- (a) Maximize stability measure. 60
- (b) Minimize difference between normal forces. 60
- (c) No cost function. 60
- Figure 3.7 Plots for the contact forces on the wheels and the stability measure α for each motion. 62
- (a) Plots for the motion that maximizes the stability measure. 62
- (b) Plots for the motion that minimizes the difference between normal forces. 62
- (c) Plots for the feasibility problem solution, i.e., no cost function. 62
- Figure 3.8 2D motions for terrains with ascending and descending trajectories. 65
- (a) 2D motion for the parabola-modeled half-pipe terrain. 65
- (b) 2D motion for the increasing/decreasing slope. 65
- Figure 3.9 2D motion for the robot driving up a step 0.2 m high and a 65° slope. 65
- Figure 3.10 Trajectories for the planar case transcribed into 3D trajectories for simulations with the actual robot. For the wheels, the first letter indicates the left (L) or right (R) and the second, indicates front (F) or hind (H). 66
- Figure 3.11 The estimation of the terrain shape by the reactive controller currently implemented on ANYmal with wheels. The terrain estimate is obtained by fitting a plane through the most recent wheel's positions and moving it by the wheel radius along the plane normal. 68
- Figure 3.12 Our extended whole-body controller (WBC) uses the terrain information to estimate the wheels contact points. (a) The approach described in (Bjelonic et al., 2019) estimates the shape of the terrain by fitting a plane through the most recent wheels' positions. Note that the frictions cones are oriented with the normal plane, which is not accurate in this case. (b) Using the knowledge of the terrain normals, we are able to predict the contact points and the friction cones along the terrain map. 68
- Figure 3.13 The ANYmal robot driving over a 0.5 m half-pipe at an average speed of 1.0 m/s. The light blue arrows on the wheels are the contact forces and the dark blue arrows are the wheels' linear velocity. The yellow sphere is the robot's CoM and the red arrow is base's linear acceleration. 69

- Figure 3.14 The desired positions provided as input to the extended WBC (dashed line) and the simulated measured positions of the robot's base and wheels (full line) while traversing the half-pipe terrain. 70
- Figure 3.15 ANYmal with wheels attempting to traverse the half-pipe terrain with blind locomotion at 1.0 m/s. As soon as the controller perceives the drop in the terrain (1), it drastically lowers its center of mass (CoM) for stabilization (2). Immediately after the descend, there is a rising on the terrain (3), but the robot is not able to adjust the pitch angle of the base in time, causing the hind legs to reach their kinematic limits and bending in the other direction (4). The robot reaches the other side of the gap in that configuration. 70
- Figure 3.16 Simulation of the robot driving over two increasing and then decreasing ramps with a 30° slope. 71
- Figure 3.17 The planned trajectory for the terrain with opposite slopes and the measured data from the simulation. 71
- Figure 3.18 Failed attempts to overcome the slopes with blind locomotion at 1.0 m/s speed (on the left), and at 2.0 m/s speed (on the right). 71
- Figure 3.19 Simulation of the robot driving up a step 0.2 m high with a 65° slope. 72
- Figure 3.20 The desired and measured linear velocity for the robot's base and left wheels while driving up the 65° step. 73
- Figure 3.21 On the left is the robot moving on a wavy slope with a purely reactive controller, while on the right is the robot traversing the same terrain using the Trajectory Optimization (TO) framework. 73
- Figure 4.1 Decision variables and constraints of the TO formulation. The constraints marked as blue are specific for optimizing driving motions over rough terrain. 76
- Figure 4.2 Coordinate frames used for the motion planning. The frame B is attached to the CoM of the robot and each wheel has a frame attached to their contact point with the ground. On the right, there is a detailed view of the wheel's contact frame C_i . The axis \mathbf{c}_z is aligned with the terrain normal \mathbf{n} and the \mathbf{c}_x axis is aligned with the rolling direction of the wheel. In the wheels' frames, the first letter indicates the left (L) or right (R) and the second, indicates front (F) or hind (H). 77
- Figure 4.3 Hermite spline parametrization. Decision variables inside the optimization are the black dots (nodes) and the red lines (state derivatives at each node). The blue curves are the polynomials defined by two consecutive nodes and their derivatives. 78
- Figure 4.4 The robot model for the optimization. The joint limits of the robot are assumed not violated if the wheel's contact point \mathbf{p}_i is in inside the parallelepiped \mathcal{R}_i . The contact forces \mathbf{f}_i on the wheels are constraint to remain inside the friction cone. 80

- Figure 4.5 Friction pyramids (in red) used as an approximation for the friction cones. 81
- Figure 4.6 Computed acceleration and velocity of the robot's LF wheel while driving 1.0 m in 1.0 s in flat terrain with and without the acceleration continuity constraint. The red dots are the times in which the decision variables are optimized over and the polynomial junctions, spaced 0.1 s apart in this case. 82
- (a) Wheel's motion generated **without** the acceleration continuity constraint. 82
- (b) Wheel's motion generated **with** the acceleration continuity constraint. 82
- Figure 4.7 Illustration of the Force-Angle Stability Measure for a quadrupedal robot. \mathbf{f} is the sum of all forces and angular loads acting on the CoM and \mathbf{f}_1 is the component of \mathbf{f} that acts on the 1st tipover axis; ϕ_1 is the stability angle w.r.t. to the first tipover axis. Same procedure is carried out to determine the correspondent \mathbf{f}_i and ϕ_i for all tipover axes. All the vectors are represented in the inertial frame. 83
- Figure 4.8 Overview of the motion planning framework. The motion planner computes the reference trajectories for a specified time horizon that is given as input to the WBC, that computes the reference torques for the robot. 86
- Figure 4.9 ANYmal drives on a steep incline with a 30° inclination at average speed of 1.0 m/s. The dark blue arrows on the wheels are the wheels' speed, the light blue arrows are the contact forces, the red arrow on the base is its linear velocity and yellow line is the CoM position vertically projected on the terrain. Results for: (1) $\beta \geq 0^\circ$; (2) $\beta \geq 10^\circ$; (3) $\beta \geq 20^\circ$ 87
- Figure 4.10 The ANYmal robot driving over a 0.5 m half-pipe at an average speed of 1.2 m/s. 88
- Figure 4.11 The ANYmal robot driving over two consecutive slopes with a height of 0.2 m at an average speed of 1.5 m/s. 88
- Figure 4.12 On the top, the linear position and velocity of the robot's base and, on the left, the right and left front wheel positions. The graphs show the desired motions provided as input to the extended WBC and the measured positions from the simulations of the two slopes maneuver. On the bottom right, the variation of the stability angle along the trajectory. The robot reaches a maximum speed of nearly 2.0 m/s. 89
- Figure 4.13 The ANYmal robot driving over two consecutive obstacles on different sides of the path at an average speed of 0.5 m/s. 90
- Figure 4.14 ANYmal drives over stairs composed by steps with 0.2 m in height and a 0.4 m distance between them. 90
- Figure 4.15 ANYmal drives over the step with a 45° slope in two different ways. Both motions are performed in average speed of 0.5 m/s. 93
- (a) The robot goes up the step by driving both wheels up at the same time. 93

(b) The robot drives up one wheel at a time.	93
Figure 4.16 Snapshots of the robot driving up a step 0.2 m high (40% of the legs length) and a 65° linear transition using the TO framework.	94
(a) The obstacle is positioned on the right side of the path.	94
(b) The step is aligned with the center of the robot's base.	94
Figure 4.17 The desired positions of the robot's base and wheels provided as input to the extended WBC (dashed line) and the measured positions (full line) obtained during the experimental tests.	95
(a) Results for the 45° step.	95
(b) Results for the 65° step.	95
Figure 5.1 Example of a contact schedule for a quadrupedal robot performing a static walking gait. Phases where the wheels are in contact are indicated with colors and lift-off phases are the white spaces. T_i is the duration of each phase.	98
Figure 5.2 Decision variables and constraints of the TO formulation for hybrid motions. The set of constraints on the wheels' variables depends on the contact state of the wheels.	99
Figure 5.3 Gazebo simulation of the ANYmal robot equipped with torque controllable wheels executing different trajectories generated by the motion planner. The purple line is the base trajectory, the green lines and the yellow lines are respectively the front wheels and hind wheels' trajectories.	101
(a) Purely walking trajectory: the wheel has zero velocity while in contact with the ground.	101
(b) Hybrid driving-walking trajectory: the wheel is allowed to move in contact phase.	101
Figure 5.4 Phase-based parametrization for the wheels' contact points and forces. Nodes (red dots) in the gray area are swing nodes while nodes in the white area are contact nodes. Graphs p_x and p_z are the x and z dimension of the wheel contact point trajectory, while graphs f_x and f_z represent the x and z dimension of the wheel contact force.	102
Figure 5.5 Some examples of gaits used for hybrid trajectory generation: (a) driving, (b) hybrid gallop, (c) hybrid walk, (d) hybrid running trot, (e) hybrid trot, (f) hybrid bounding. Contact phases are indicated with colors and swing phases with white spaces. An example of the wheels' trajectories for the hybrid running trot are indicated in the top figure, each with a different color, matching the colors in the gait pattern graphs.	103
Figure 5.6 Variables of the NLP for a quadrupedal robot performing a hybrid running trot. The parallelepiped \mathcal{R}_i indicates the feasible workspace for i^{th} the wheel.	104
Figure 5.7 Coordinate frames used for the hybrid trajectory optimization. The frame B is attached to the CoM of the robot and each wheel in contact phase has a frame C_i attached to its contact point with the ground.	105

Figure 5.8	Influence of the acceleration continuity constraint on the generated motions. The red dots are the polynomial junctions and the gray areas are the contact phases.	107
(a)	Wheel's motion generated without the acceleration continuity constraint.	107
(b)	Wheel's motion generated with the acceleration continuity constraint.	107
Figure 5.9	Position, velocity and acceleration profiles of the robot's LF wheel in the z -direction, with the acceleration constraint.	107
Figure 5.10	Overview of the motion planning framework for hybrid motions. The motion planner computes the reference trajectories for the base and the wheels that are given as input to the WBC, that computes the actuation torques for the robot.	108
Figure 5.11	The motion plans for the base and the wheels generated for different gaits in flat terrain.	110
(a)	Driving gait.	110
(b)	Hybrid gallop gait.	110
(c)	Hybrid running trot.	110
Figure 5.12	The desired positions provided as input to the WBC and the measured positions from the simulation of the robot performing a hybrid running trot motion at an average speed of 1.0 m/s.	111
Figure 5.13	On the top left, a top view of the lateral hybrid trot motion, where the robot has to go from position $(x, y) = (0.0, 0.0)$ to $(x, y) = (2.0, 1.0)$. On the top right, an overview of the motion, in which it is possible to see the swing phases. At the bottom, two snapshots of the motion's simulation showing the lateral displacement of the wheels.	111
Figure 5.14	The robot moving laterally and then turning 90° with a dynamic hybrid trot gait.	112
Figure 5.15	Plot results for the combined lateral motion and base turning. The dashed lines are de desired motions, with the full lines are the data obtained from the whole-body simulation.	112
Figure 5.16	The ANYmal robot performing a dynamic hybrid bounding gait to cross a 0.25 m gap at an average speed of 0.75 m/s.	114
Figure 5.17	Simulation results the hybrid bounding maneuver over the gap.	114
Figure 5.18	The ANYmal robot performing a dynamic hybrid gallop gait to cross the same gap.	115
Figure 5.19	Simulation results for the hybrid gallop maneuver over the gap.	115
Figure 5.20	The ANYmal robot performing a dynamic hybrid gallop gait to cross a step with a 0.2 m height.	117
Figure 5.21	Comparison between the reference trajectories provided to the WBC and the obtained measurements from simulations.	117
Figure 5.22	The ANYmal robot crossing a 0.2 m high step on the right side of its path.	118

Figure 5.23 Simulation results show barely any error on the trajectory tracking for the lateral step. The executed trajectories are almost identical to the desired motions. 118

Figure A.1 Example of a cubic curve using Hermite parametrization. 135

List of tables

Table 3.1	Robot's parameters used for the 2D trajectory optimization.	59
Table 3.2	Different terrains used to test the 2D TO framework.	64
Table 4.1	Computation times for some of the terrains used to test the 3D TO framework.	91
Table 5.1	Computation times for some of the maneuvers used to test the hybrid TO framework in flat terrain.	113

List of Acronyms

BVP – Boundary Value Problem

CoM – Center of Mass

EoM – Equations of Motion

IMU – Inertial Measurement Unit

MPC – Model Predictive Control

NLP – Nonlinear Programming

OCP – Optimal Control Problem

ODE – Ordinary Differential Equation

QP – Quadratic Program

RBD – Rigid Body Dynamics

RMSE – Root-Mean-Square-Error

ROS – Robot Operating System

RSL – Robotic Systems Lab

RTF – Real-Time Factor

SQP – Sequential Quadratic Programming

SRBD – Single Rigid Body Dynamics

TO – Trajectory Optimization

WBC – Whole-Body Controller

ZMP – Zero-Moment Point

1 Introduction

The number of applications for autonomous ground robots has substantially increased in the last 20 years and extensive research has been carried out with the main objective to enable autonomous mobile robots to be safely deployed for different applications, such as planetary exploration, industrial inspection, precision farming, search and rescue tasks and payload delivery. Mobile robots can withstand harsh conditions, such as extreme temperatures or high levels of radiation, and they can be sent to perform tasks in places either inaccessible or too dangerous for humans. In industrial sites, mobile robots can lower the amount of human intervention required in monitoring and maintenance, improving safety and saving considerable operating costs.

Several prototype robots have been developed and tested for these applications in the last few years with promising results. Mobile robots have been investigated for planetary exploration since the 1997's NASA Mars Pathfinder mission, in which the Sojourner rover (Mishkin et al., 1998), depicted in Figure 1.1(a), was deployed to explore the surface of Mars (Bajracharya et al., 2008). Now, the new rover Perseverance (NASA Science, 2020), depicted in Figure 1.1(b), is schedule to be launched to Mars in July 2020.

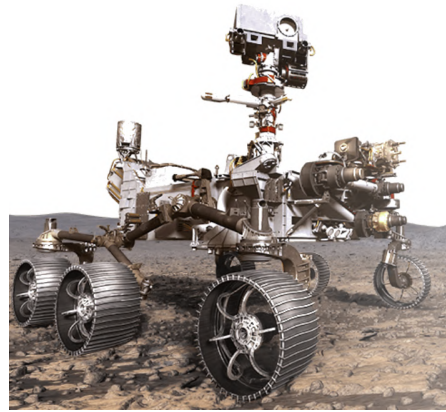
The Autonomous Robot for Gas and Oil Sites (ARGOS) challenge (Total, 2014) was launched in 2014 to foster the development of autonomous inspection robots for oil and gas production sites, encouraging the deployment of mobile robots for monitoring and maintaining of industrial installations. Figure 1.2 shows the robots developed for the challenge. Some of these prototypes are already being commercialized for this application. Similarly, the U.S. Defense Advanced Research Projects Agency (DARPA) has launched several challenges (DARPA, 2015) to advance and accelerate the safe deployment of robots for diverse scenarios, such as disaster response, search and rescue tasks and subterranean exploration. Figure 1.3 shows some of the ground robots developed for the DARPA challenges from 2015 until today.

This large ongoing investment and research in mobile robotics indicates that the development of ground robots will continue to advance until it reaches a point where they have sufficient robustness to be autonomously deployed for most of these scenarios and many other industrial and non-industrial

applications.



(a) Sojourner rover.



(b) Perseverance rover.

Figure 1.1: Rovers for exploration at the surface of Mars.

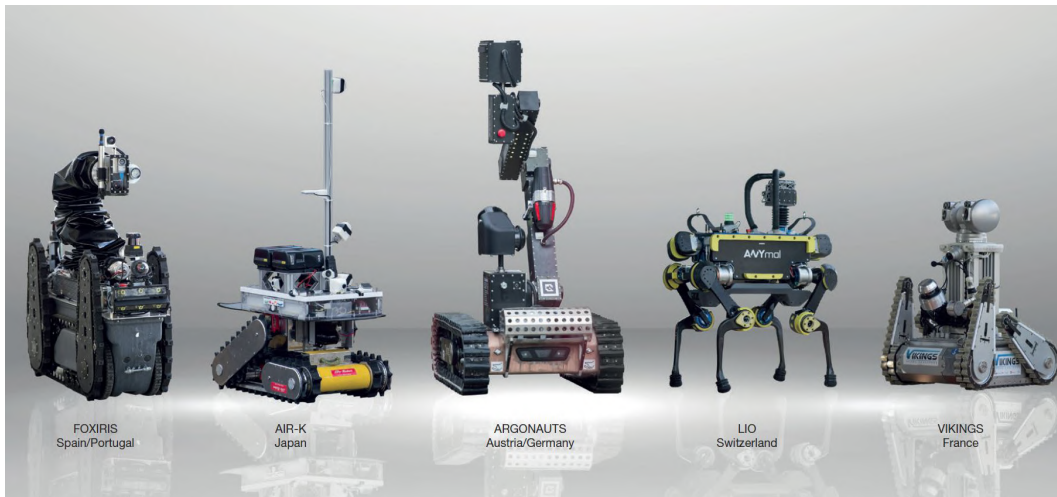


Figure 1.2: The main robot prototypes that competed in the ARGOS challenge.



Figure 1.3: Different robot designs for the current and past DARPA Challenges.

Currently, research in mobile robotics presents a number of challenges in different fields, such as locomotion, perception, localization and navigation. Most of the applications for mobile robots requires safe navigation in rough and unstructured terrain, often with obstacles, which represents a great mobility challenge for the robot. For that reason, this thesis focuses mainly on the challenges of locomotion in unstructured terrain.

Mobile robots for rough terrain are usually divided in two categories: legged robots and wheeled robots. Legged robots have excellent mobility to cope with challenging terrain and are able to overcome large obstacles. Wheeled robots, on the other hand, exhibit high maneuverability on continuous terrain, moving faster and more efficiently than legged systems. One attractive solution is to combine the advantages of both locomotion systems into a wheeled-legged system, that can cope with challenging environments at higher speeds (Bruzzone and Quaglia, 2012). Tasks where the execution time is important would greatly benefit from such systems, such as payload delivery or search and rescue.

In general, the challenges with wheeled-legged locomotion involve trajectory generation and controlling the robot for efficiently tracking the desired trajectories, i.e., motion planning and motion control. The motion planning task involves the formulation and solving of a TO problem, taking into consideration several physical constraints such as stability, dynamics, torque and speed limitations of the actuators, and slippage constraints. On the other hand, the motion control framework is responsible for determining the appropriate torques for the legs' joints and the wheels in order to track the reference motions. This has been the focus of several previous studies and it is commonly done with Inverse Dynamics, that computes the joint torques by solving the Rigid Body Dynamics (RBD) model of the robot for the desired motion (Righetti et al., 2011).

Several approaches can be considered for developing a motion framework for wheeled-legged robots depending on certain locomotion conditions, such as static or dynamic motion, terrain mapping or blind locomotion, proprioceptive or exteroceptive¹ sensing. Considering a static or quasi-static condition, i.e., the robot speed is sufficiently low to neglect inertial effects, a purely reactive controller can be employed for adapting to terrain variations by maintaining a constant desired CoM pose (position and orientation) or by continuously adjusting the CoM to achieve a stable configuration according to some stability

¹Proprioceptive sensors measure values internal to the robot, such joint positions, motor speed, wheel load, battery voltage and base orientation (via inertial sensors). Exteroceptive sensors acquire information from the robot's environment, such as distance measurements, light intensity and sound amplitude (Siegwart et al., 2011).

criteria. In this case, the environment is perceived through proprioceptive sensing and obstacle negotiation is performed on a static step-by-step basis. When performing dynamic motions at higher speeds, reactive approaches can be employed for flat or benign terrain, while obstacle negotiation is enabled through knowledge of the terrain map, taking into account dynamic effects on the robot's center of mass.

In this context, this thesis addresses the trajectory generation problem for wheeled-legged robots with actuated wheels performing dynamic motions in challenging terrain, either through purely driving or hybrid driving-stepping motions. Moreover, we extend the existing control frameworks for such systems by including the terrain map information into the motion controller.

1.1 Objectives

The main objective of this work is to develop a motion optimization framework for wheeled-legged robots with actuated wheels that allows the robot to navigate in challenging and unstructured terrain, e.g., steps, slopes, stairs, gaps, while negotiating these obstacles with dynamic motions. The trajectory optimization framework is responsible for generating the 6 degrees of freedom (DoF) of the robot's base motion (position and orientation) as well as the wheels' positions and contact forces along the trajectory, accounting for the terrain map and the dynamics of the robot. The robot's dynamic model is employed for both motion planning and tracking and no static or quasi-static condition is assumed which, together with the terrain map, allows for planning agile motions. This way, given a target position and a desired average velocity, the motion planner will generate a feasible motion to overcome the mapped terrain. To take full advantage of the hybrid nature of wheeled-legged robots, pure driving and hybrid walking-driving motions are considered in different steps of the research.

While walking locomotion has several advantages in unstructured terrain, the planning task is harder to solve in comparison to pure wheeled locomotion. Driving motions are faster, less energy consuming, more stable and there is no need for a foothold placement strategy or other difficulties associated with legged locomotion. For this reason, a trajectory optimization framework is proposed for generating purely driving motions that allows the robot to overcome challenging obstacles by using the terrain map. This shows that a lot of challenging scenarios can be surmounted with driving motions since the presence of the legs allow for negotiating obstacles that would not be possible with a conventional wheeled robot. The framework is validated

through simulations and experimental tests using customized obstacles in several configurations.

Even though driving motions are suitable for several scenarios, they have some limitations. For discontinuous terrain types, such as small gaps or floating steps, stepping motions are necessary. Thus, the formulation for purely driving motions is extended to allow the generation of hybrid walking-driving motions employing a variety of gaits for discontinuous terrain types. In our work, hybrid locomotion denotes simultaneous walking and driving, not switching between them. The formulation is based on a phase-based parametrization proposed by (Winkler et al., 2018) for legged robots with point feet. The trajectory optimization framework for hybrid motions is demonstrated in simulations with discontinuous obstacles, floating steps and gaps. A discussion on its applicability to the real robot is presented based on the simulation results.

To improve efficiency and integration with real robot, the motion planning frameworks are all coded in C++ and implemented using Robot Operating System (ROS).

The verification of the physical feasibility of the motion plans is carried out in different ways. An initial assessment is made through pure visualization using MATLAB or RViz², which is useful to verify whether the motions are at least visually consistent and there were no major modeling mistakes. However, there is no actual physical simulation and inconsistent accelerations or contact forces would not be identified through visual inspection. A more reliable verification is carried out using a physics-based simulator of the robot model, such as Gazebo (Koenig and Howard, 2004), accounting for the full rigid body dynamics of the robot, the torque limits of the actuators and the friction cone constraints. This ensures that the motion plans are physically correct and indicate their feasibility for application on the real platform. Finally, the ultimate validation comes from experimental tests with the actual robot, which proves that the tracking controller can track the motion plans on a physical system, regardless of eventual modeling errors from physical approximations in the simulator and noise from real-world sensor data.

The robotic platform used for the tests is the robot ANYmal (Hutter et al., 2016), a quadrupedal robot developed by the Robotic Systems Lab, at ETH Zürich, Switzerland, that it was equipped with actuated torque-controllable non-steerable wheels (Bjelonic et al., 2019). Figure 1.4 depicts each of the aforementioned verification tools for the motion plans using ANYmal with wheels. Even though this work focuses on quadrupedal robots, all of the

²RViz is a 3D visualization tool for ROS that allows creating visualizations of the simulated robot model.

approaches presented should be adaptable to robots with more or fewer legs.

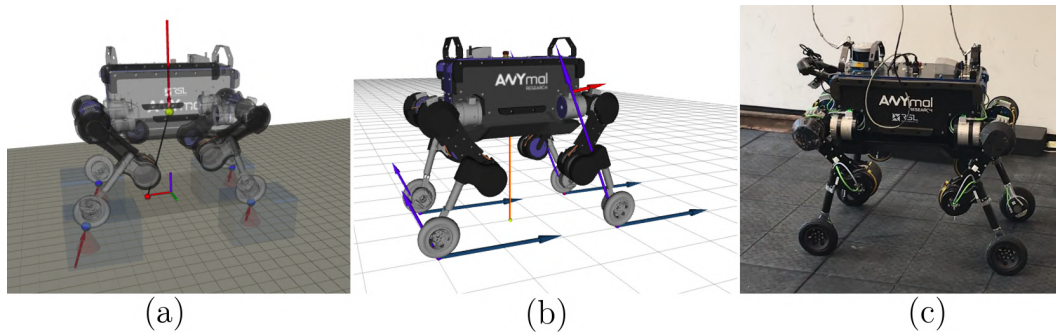


Figure 1.4: Different verification methods for the feasibility of the motion plans. (a) Pure visualization with RViz; (b) Physical simulation with Gazebo; (c) Experiments on real robot.

1.2

Related Work

Motion planning for quadrupedal robots is an established field and previous authors have achieved impressive results (Bellicoso et al., 2018; Winkler et al., 2018; Dai et al., 2014; Kudruss et al., 2015; Wermelinger et al., 2016). These planners are designed to generate dynamic motions for legged robots with point feet. However, a similar set of constraints needs to be enforced for wheeled-legged locomotion, such as friction and kinematic constraints. This is why some of these previous implementations can be used as reference for this thesis. In particular, the trajectory optimization framework proposed by (Winkler et al., 2018) for legged robots, in which the CoM position, orientation, feet position, contact forces and gait timings are concurrently optimized, based on a Single Rigid Body Dynamics (SRBD) model of the robot.

On the other hand, in the field of planning for wheeled-legged robots, most of the previous work focuses on robots performing motions in a purely reactive fashion, rather than generating optimized trajectories for dynamic motion. A number of authors have proposed reactive controller frameworks for wheeled-legged locomotion over uneven terrain. The most common examples are extra-planetary rovers (Lauria et al., 2002; Iagnemma and Dubowsky, 2004; Grand et al., 2010; Cordes et al., 2018; Reid et al., 2016), which employ a purely reactive controller that can adapt to terrain variations by maintaining a desired base pose. These controllers are typically able to execute statically-stable³ driving motions at low speeds, where the legs act as a sophisticated active suspension system.

³Statically stable locomotion requires the moving body to be stable at all times, which means that motion is not needed for maintaining balance. More specifically, the vertical projection of the center of gravity of the moving body will be contained within the convex

In (Iagnemma et al., 2000), a kinematic reconfigurability method for extra-planetary rovers is developed to enhance their mobility in rough terrains. It consists of reconfiguring an articulated robot by minimizing a performance index that takes into account the robot's stability (Papadopoulos and Rey, 2000) and the displacement of the actuated joints. The results are demonstrated on the *Sample Return Rover* (SRR), presented in Figure 1.5(a). A motion control framework for wheeled robots in rough terrain that optimizes the torque in each wheel is presented in (Iagnemma and Dubowsky, 2004). The objective is either to maximize traction or minimize energy consumption, depending on the level of difficulty of the terrain. For more uneven terrain the traction is optimized, and for more benign terrain, the control minimizes the energy consumption. In both cases, a quasi-static condition is assumed and the wheel-terrain contact angles are estimated using simple on-board sensors.

The PAW robot (Smith et al., 2006), shown in Figure 1.5(b), is a quadruped robot equipped with four passive springy legs and four actuated wheels at the end of its legs. The robot is unable to support itself on three legs, which means that it can only perform jumping or bounding maneuvers, while step climbing is possible only in a quasi-static manner. In (Turker et al., 2012), a tailored step-climbing strategy is presented for the PAW robot using wheel traction optimization and posture reconfiguration, limited to a quasi-static condition. The step is detected through measurements obtained from motor encoders and an inertial measurement unit (IMU) on the base. Experimental results show the robot climbing steps with different heights.

The robot Octopus, introduced in (Lauria et al., 2002), is a wheeled mobile robot with 8 motorized wheels and a total of 14 joint actuators (see Figure 1.5(c)). The robot is equipped with tactile wheels that detects obstacles and allows the robot to adapt its behavior to the terrain. A two-dimensional static model and a controller are proposed for the robot. The motion controller optimizes over the wheels and joint's torques by maximizing the ratio between friction and normal contact force on each wheel. A static step-climbing sequence is demonstrated on the real robot.

In (Grand et al., 2004; Grand et al., 2010), a reactive controller for active posture adaptation to terrain variations is proposed for the hybrid wheeled-legged robot Hylos (Figure 1.6(a)). The authors present a general kinetostatic formulation for quasi-static motions of articulated wheeled rovers, and a decoupled posture and trajectory control strategy based on the differential kinematic model of the robot. The desired posture of the robot (composed hull of the body's points of contact with the ground at all times. It is mainly applicable for analyzing low-speed motions because it does not consider inertial forces or external impacts.

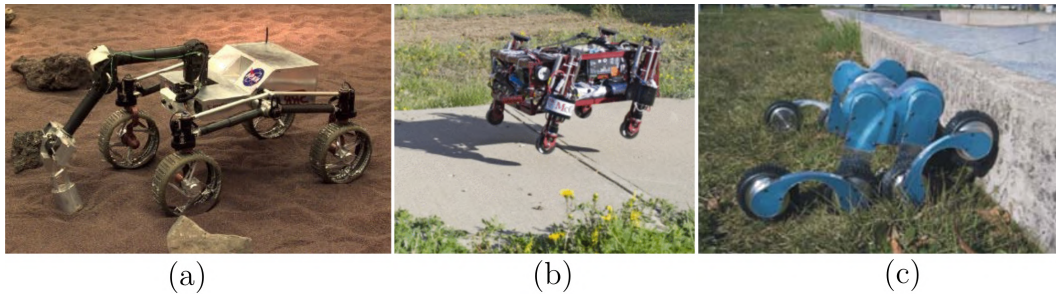


Figure 1.5: Articulated wheeled robots: (a) The Sample Return Rover (SRR), developed by the *Jet Propulsion Lab* for extra-planetary exploration; (b) The PAW (Platform for Ambulating Wheels) robot; and (c) The Octopus robot, developed at EPFL, Lausanne, Switzerland.

of the base height, roll and yaw angles, and the wheels' positions) is either pre-defined as a constant pose or optimized to obtain an equal-distribution of loads between the wheels and maintain stability (Grand et al., 2002). The approach is validated in experiments with the real robot, moving at a average speed of 0.08 m/s on a terrain constituted of slopes with various inclines, as shown in Figure 1.6(b).

Similarly, the motion control system developed in (Wong et al., 2015) for the MHT robot (Figure 1.6(c)) is also based on a velocity-level closed loop inverse kinematics controller and presented good experimental results for stationary posture reconfiguration, but it was not able to cope with uneven terrain.

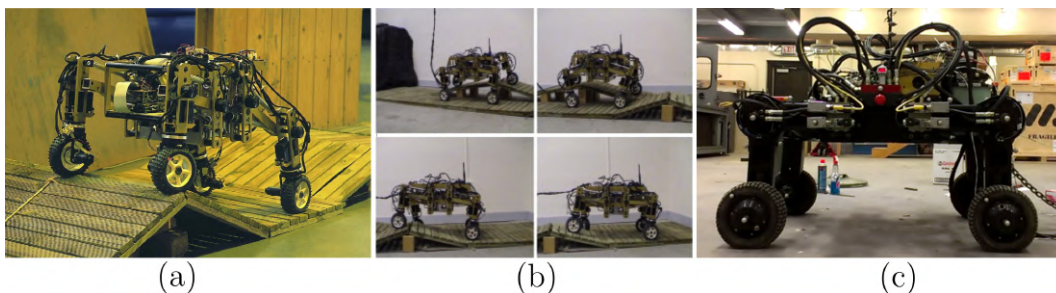


Figure 1.6: Articulated wheeled robots: (a) The Hylos robot, developed by the *Institut des Systèmes Intelligents et de Robotique* (ISIR) of the Université Pierre et Marie CURIE, Paris, France; (b) The Hylos robot navigating on asymmetric inclines; and (c) The MHT (Micro-Hydraulic Toolkit) robot, developed by the Defense Research and Development Canada Suffield Research Centre.

In (Jarrault et al., 2011), the authors present an algorithm for obstacle crossing for the robot Hylos2 that optimizes over the ground reaction forces and the CoM posture to maximize a friction stability measure. The method was validated in simulations with the robot. Experimental validation is presented

further in (Jarrault et al., 2014), that shows the robot crossing a step with a 60° slope, as shown in Figure 1.7. The robot took roughly 70 seconds to complete the maneuver.

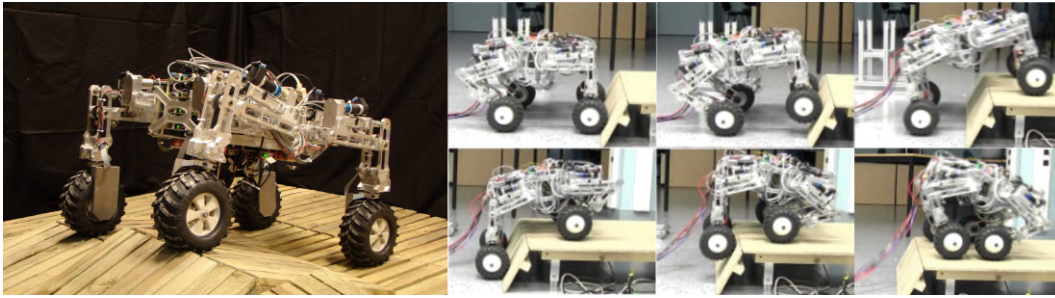


Figure 1.7: On the left, the Hylos2 robot. On the right, the robot crossing a step with 60° slope employing a quasi-static obstacle crossing approach.

In (Reid et al., 2016), an active terrain adaptation algorithm is presented for the MAMMOTH rover, presented in Figure 1.8(a). A kinematic model of the robot is used to control its active articulations to maintain a constant body pose while traversing different terrain obstacles. Instead of using only proprioceptive data, as the previous papers, the robot uses terrain mapping data from an RGB-D sensor to enforce contact between the wheel and the ground. Experimental tests are shown with the rover moving over a bed of rocks at low speeds, between 0.05 m/s and 0.075 m/s. Another rover with an active suspension system is introduced in (Cordes et al., 2018), the SherpaTT, depicted in Figure 1.8(b). It is equipped with a purely reactive control system for active terrain adaptation. The controller uses force measurements at each wheel and the base roll-pitch data to compute the positions of the legs and the center of gravity to maintain permanent wheel-ground contact and optimal force balancing. Experimental results show the rover negotiating slopes from 9.5° to 28° at a speed of 0.04 m/s. For both rovers, a quasi-static condition is assumed.

Other active articulated wheeled robots that employ kinematic reconfigurability methods for traversing uneven terrain can be found in (Wettergreen et al., 2010; Freitas et al., 2010; Inotsume et al., 2013; Jiang et al., 2019). On the other hand, the wheeled-legged robots presented in (Suzumura and Fujimoto, 2012; Giftthaler et al., 2017; Giordano et al., 2009) use a kinematic-based control algorithm to generate velocity commands for the wheels, but they only present results for flat terrain navigation and do not consider the terrain in the planning algorithm.

The work in (Bouton et al., 2016; Bouton et al., 2020) introduces the robot Complios, that stands on four wheels each linked to the chassis via a compliant leg, composed of two pivoting segments and antagonistic springs,

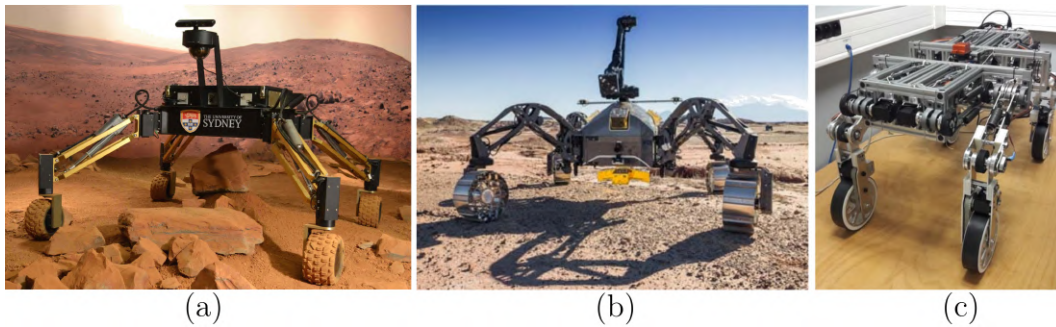


Figure 1.8: Rovers with active suspension: (a) The MAMMOTH rover, developed by the Australian Centre for Field Robotics (ACFR), University of Sydney, Australia; (b) The SherpaTT rover, from the University of Bremen, Germany; (c) The Complios robot, developed by the Université Pierre et Marie CURIE, Paris, France.

and a series elastic actuation system. The robot is able to spontaneously adapt its configuration to uneven terrain, while keeping a reactive control on the load distribution and the platform posture. Experimental results show the robot driving over different asymmetric terrain shapes at a average speed of 0.1 m/s. In addition, some interesting results have been presented by using a reinforcement learning technique for obstacle negotiation for same robot (Bouton et al., 2017). The approach uses external forces measurements on the wheels to detect an obstacle and the controller is trained to choose from a set of pre-defined behaviors to escape from the obstruction. However, the learning framework has only been tested in simulations.

So far, all of the discussed applications focus on driving motions and do not show any instance of wheel lift-offs. Another approach that has been used for computing motion plans for wheeled-legged robots is to switch between walking and driving based on heuristics that rely on the terrain complexity or on commands received from a human operator.

The wheeled humanoid robot DRC-HUBO+ presented in (Bae et al., 2016) switches between walking and driving mode by changing its configuration, as shown in Figure 1.9(a). The robot is teleoperated by three different operators which control the robot by sequencing predefined poses (Zucker et al., 2015). While in driving mode, the robot is in a crouched position and the legs are not used for locomotion or stability, which makes the robot unable to overcome obstacles while driving.

Unlike DRC-HUBO+, the wheeled quadrupedal robot Momaro (Figure 1.9(b)) is equipped with actuated wheels and doesn't require a change in configuration to perform driving motions. (Klamt and Behnke, 2017) presents a motion planning framework for the Momaro robot which uses a perception system to map the environment and, based on a path cost, it chooses between

walking over an obstacle or driving around it. Driving is performed only in flat terrain, while obstacles are overcome by walking motions. The robot uses a kinematic controller architecture (Schwarz et al., 2016) and the robot is shown to overcome stairs and steps at low speed, neglecting dynamic effects and considering only static stability.

An evolution of Momaro, the robot CENTAURO (Klamt et al., 2018) presents a similar approach employing a locomotion framework that switches between stepping or driving. The wheeled-legged robot has four legs ending in steerable wheels and an anthropomorphic upper body, as shown in Figure 1.9(c). The driving task is accomplished using joystick teleoperation, while walking motions are executed by a semi-autonomous controller that provides a set of pre-defined motions that can be triggered by the operator. Experimental results show the robot overcoming obstacles like stones, steps, gaps and attempts to overcome stairs with slow static maneuvers. A further research (Laurenzi et al., 2018) presents a motion planner that generates purely walking motions for the same robot, through the joint optimization of both the CoM trajectory and the footsteps using a linear Model Predictive Control (MPC) framework. However, the experiments are limited to walking motions in flat terrain and the wheels are not used during the motions.

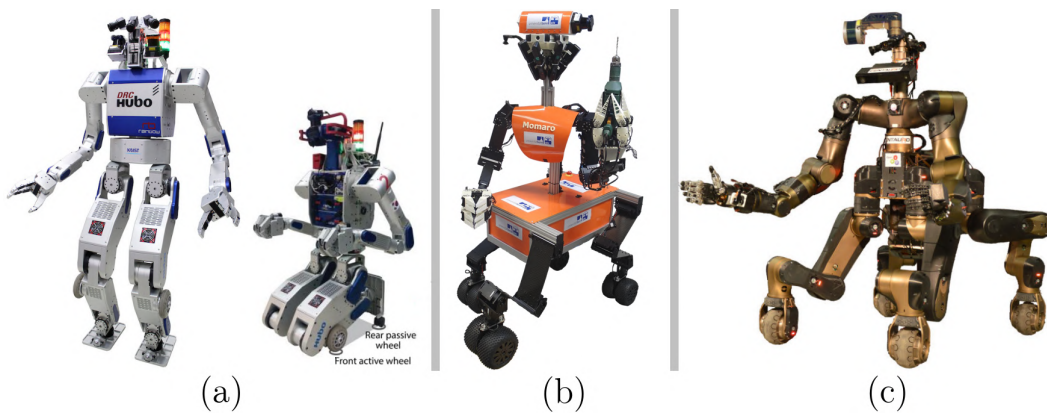


Figure 1.9: Hybrid robots where driving and stepping phase are treated as separate behaviors: (a) The wheeled humanoid robot DRC-HUBO+ in its two configurations: walking mode (left) and driving mode (right); (b) The Momaro robot, developed by the University of Bonn, Germany; (c) The CENTAURO robot, developed by the *Istituto Italiano di Tecnologia* (ITT).

Such approaches do not consider solutions where the robot uses its wheels and legs simultaneously, which limits their ability to overcome obstacles compared to considering the whole-body in a single planning problem. Moreover, a continuous motion without constant switching between driving and walking mode would be more efficient.

On the other hand, the motion planner presented by (Jelavic and Hutter, 2019) solves the whole-body planning problem combining driving and stepping motions, also allowing for gait optimization. The framework, however, focuses on generating kinematically feasible motions for wheeled-legged vehicles performing slow maneuvers, which is the case for heavy machines such as walking excavators. The results are demonstrated with the Menzi Muck M545, shown in Figure 1.10, but no physical validation is presented.



Figure 1.10: The Menzi Muck excavator, on the left. On the right, simulations with the excavator overcoming a step-like obstacle using the wheeled legs and the shovel simultaneously.

As can be seen, most of the previous work focuses on statically stable locomotion for wheeled-legged robots, where a kinematic approach is often employed for the motion planning and control. However, several advances have been recently presented for dynamic motion generation as well.

Dynamic motions in challenging terrain has been demonstrated by the two-wheeled legged robot Handle from Boston Dynamics (Boston Dynamics, 2019), where the wheels are used for driving over flat terrain and the legs are used to jump over high obstacles. However, there are no publications regarding its locomotion framework. The robot Ascento (Klemm et al., 2019) presents a similar behavior focused on navigation in indoor spaces. However, a pre-defined jump motion is triggered by the operator and no trajectory optimization is employed. Both robots are presented in Figure 1.11. Despite the similar architecture, they have very different dimensions, Handle can stand up to 198 cm, while Ascento's height varies from 31 cm to 66 cm.

The work presented in (Geilinger et al., 2018; Geilinger et al., 2020) shows a new tool, called *Skaterbots*, that optimizes the mechanical design of wheeled-legged robots with arbitrary number of legs. Some of the robot designs are presented in Figure 1.12, both with passive and actuated wheels. The framework uses a general trajectory optimization framework that optimizes over several types of motions, such as skating, walking and driving, by solving



(a) The Handle robot, from Boston Dynamics.



(b) The Ascento robot, from ETH Zürich, Switzerland.

Figure 1.11: Two wheeled-legged robots capable of driving and jumping over obstacles.

an NLP problem. However, it assumes the robot is moving on flat ground and it does not include any terrain information. Experimental tests are carried out with different robots moving over flat terrain and in teleoperation mode.



Figure 1.12: Some of the wheeled-legged robots created by the *Skaterbots* framework, which are capable to perform different types of motions in flat terrain.

The quadrupedal robot Robosimian (Satzinger et al., 2014; Hebert et al., 2015; Satzinger et al., 2015) was developed by JPL (Jet Propulsion Lab, Palo Alto, California) for the 2015 DARPA Challenge (DARPA, 2015) and it is presented in Figure 1.13 (on the left). The robot has four identical limbs, each with seven degrees of freedom. The inverse kinematics for a robot with that many degrees of freedom proved quite challenging (Satzinger et al., 2015), but the robot has been shown to perform statically stable walking locomotion over a set of steps. Recently, a single, passive (unactuated) wheel with no sensing was mounted at each of the Robosimian's forearm to transform the robot into a wheeled-legged system, as shown in Figure 1.13 (on the right). A trajectory optimization framework for the wheeled-legged version of the robot is presented

in (Bellegarda and Byl, 2019), which generates dynamic hybrid driving-walking motions taking into account the slippage on the wheels. However, the motions are specific for systems with passive wheels and are only shown in simulation. Moreover, the motions are only performed in flat terrain.

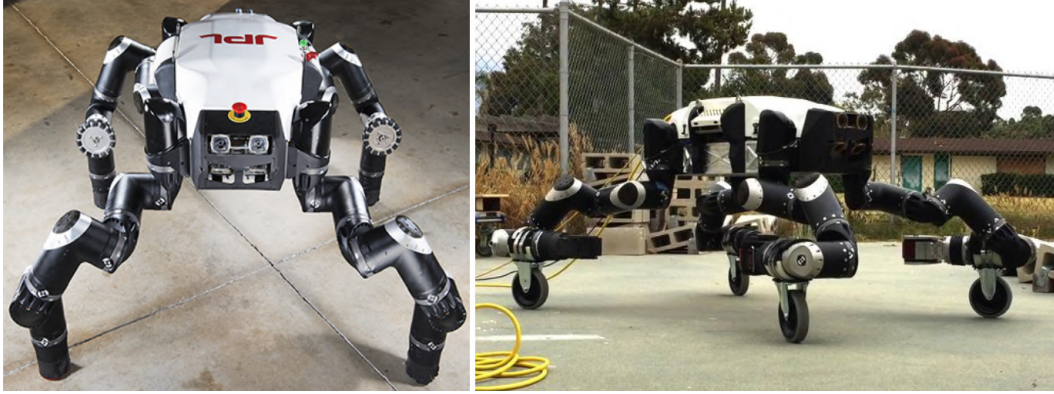


Figure 1.13: The legged Robosimian robot in its default position (on the left) and the robot equipped with passive wheels for skating locomotion (on the right).

All of these approaches, even though can generate dynamic motions, do not take into account terrain information and the flat terrain assumption makes them not very well suited for cases where the terrain contains steps or steep slopes.

In contrast, a very recent work presented in (Sun et al., 2020) introduces the quadrupedal wheeled robot Pholus, developed by the National University of Singapore in cooperation with the IIT, shown in Figure 1.14. It employs a hierarchical control framework designed to perform hybrid locomotion in uneven terrain. The motion planning task is divided into two stages: the foot placement planning and the CoM trajectory optimization, implemented using an MPC approach and considering terrain height information. A numerical inverse kinematics controller is developed for tracking the motions. Experimental results shows the robot overcoming thin obstacles and steps with statically stable hybrid motions at low speeds. Dynamic motions, such as bounding, have only been presented in simulations. In this case, the terrain information is included, but obstacles are traversed in a non-dynamic manner. In addition, several aspects of the motions, such as the base height and the foot positions and velocities, are pre-defined using heuristics based on the terrain map, rather than being optimized for general situations.

Planning for the foot placement position and the base motion in separate steps has been employed for both legged (Kajita et al., 2003; Kalakrishnan et al., 2010; Bellicoso et al., 2018) and wheeled-legged (Bjelonic et al., 2019; Bjelonic et al., 2020; Sun et al., 2020) robots. In such cases, the feet trajectories

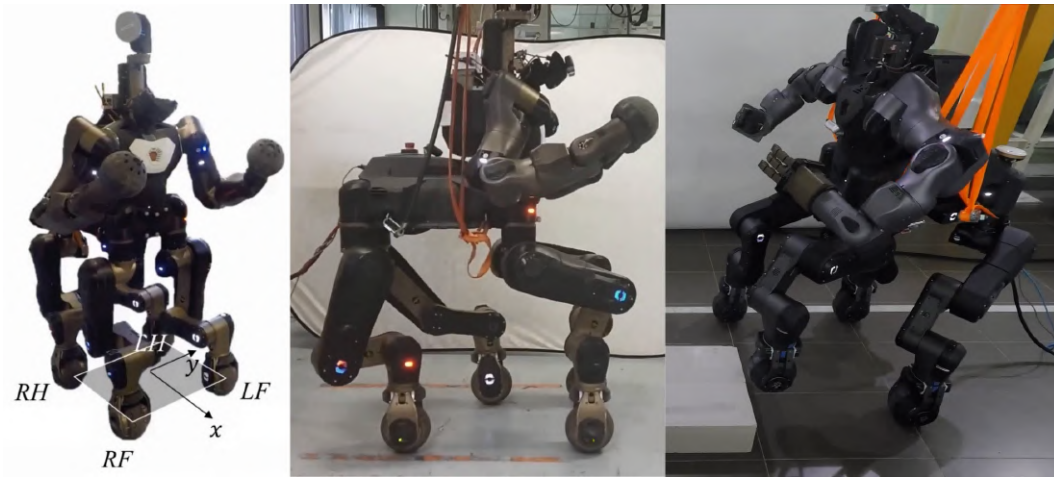


Figure 1.14: The wheeled-legged robot Pholus, a centaur-like robot capable of hybrid locomotion.

are computed firstly based on the desired gait, the desired average speed, and the terrain map, if available. Given the planned foot positions, the robot's CoM motion is then optimized to satisfy some dynamic stability criterion⁴. For that, the most common approach is based on the Zero-Moment Point (ZMP)⁵ (Vukobratović and Borovac, 2004) stability criterion, subject to kinematic and dynamic constraints. However, the gain in computational cost with a divided approach can represent limitations in generality and performance. For instance, the ZMP has a well-known validity limitation: it can only be applied when all contacts are at a constant height. If the robot has to place the feet at different heights to cross uneven terrain, the ZMP no longer has a connection with dynamic stability (Caron et al., 2017). Moreover, handling uneven terrain and force constraints is more complicated. For such cases, it might be more appropriate to use an approach that optimizes over the feet and base motions simultaneously, taking into account the dynamics of the system and how the contact forces affect the base. A unified approach without pre-defined restrictions allows the solver to plan for more complex motions. Moreover, constraints related to friction or unstructured terrains can be easily included in the formulation.

⁴The motion of the robot, at a certain state, is considered to be stable if the torque caused by the ground-reaction forces can prevent the robot from tumbling around any support boundary (Jia et al., 2018).

⁵For a flat ground plane, the ZMP is defined as the point on the ground where the moment induced by the gravito-inertial forces has only a component in the direction of the plane normal. For dynamic stability, the planned ZMP must always lie within the support polygon formed by the planned foot placements.

1.2.1 ANYmal on Wheels

ANYmal (Hutter et al., 2016) is a quadrupedal robot developed by the Robotic System Lab, at ETH Zürich, Switzerland. Its legs feature three actuated joints arranged as a successive hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE). The joints are mounted in a way that permits full rotation, which enables a broad range of motions and allows the robot to climb large obstacles. All the joints are torque controllable and actuated by series-elastic actuators. In (Bjelonic et al., 2019), non-steerable torque-controlled wheels were installed at the end of ANYmal's legs to allow for hybrid driving and walking locomotion. Figure 1.15 depicts the configuration of the joints on the robot and the workspace of the legs for the wheeled version of ANYmal.

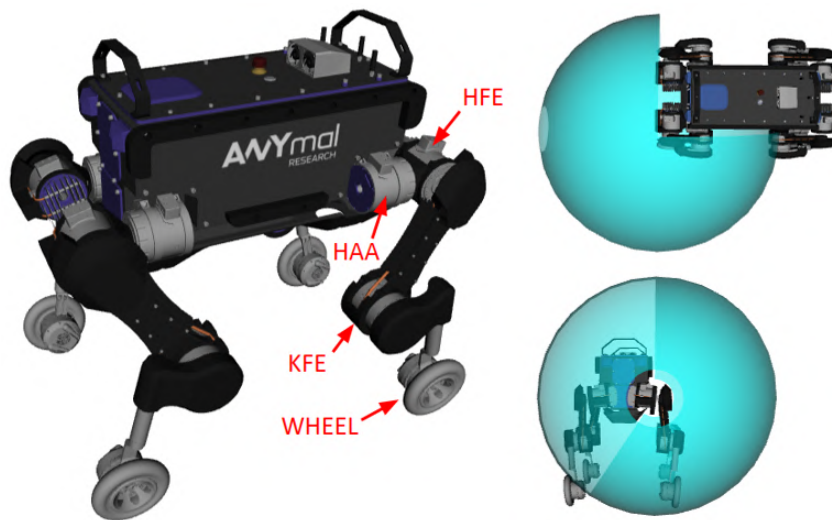


Figure 1.15: On the left, the position of the actuated joints on ANYmal's legs and, on the right, the blue sphere depicts the reachable workspace of one wheel.

ANYmal with wheels is equipped with several sensors for autonomous navigation: an IMU, a LIDAR (Light Detection and Ranging) sensor, a depth camera and a RTK GPS. The locomotion control system of the robot is composed of several modules, including perception, mapping, localization, motion planning and whole-body control. In addition, a comprehensive simulation environment based on Gazebo is available for testing new locomotion controllers for the robot.

Reactive blind locomotion has been implemented for ANYmal with wheels with successful experimental tests in (Bjelonic et al., 2019) and (Bjelonic et al., 2020). In (Bjelonic et al., 2019), a motion planning and control framework is developed for dynamic locomotion. Given the gait pattern and the reference linear and angular velocities for the robot's base, the reference footholds

for each leg are locally computed as a function of the measured and desired velocity of the torso (Bellicoso et al., 2017). Then, the reference trajectories for the CoM are continuously computed by a ZMP optimization. Experimental tests were carried out demonstrating slope negotiation and rough terrain navigation. The robot is shown to execute walking and driving motions, but not simultaneously, the robot needs to stop and switch to a pure walking mode to overcome small steps.

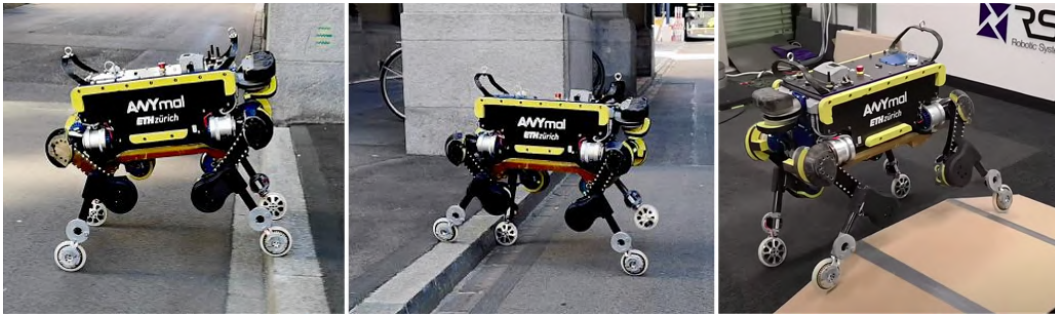


Figure 1.16: The ANYmal with wheels negotiating slopes and small steps with a reactive controller. Reprinted from (Bjelonic et al., 2019).

In contrast, the TO presented in (de Viragh et al., 2019) generates hybrid walking-driving motions for wheeled-legged robots using a linear formulation of the ZMP balance criterion. The approach is tested in simulations with ANYmal equipped with actuated wheels executing several hybrid motions at high speed, but limited to flat terrain.

The trajectory optimization presented in (Bjelonic et al., 2020) also allows for hybrid driving-walking motions. The optimization problem is divided into wheels and base trajectory planning, which allows for online computation in a MPC fashion. Given the gait pattern and the reference velocities for the base, the wheels trajectories are computed first, followed by the base TO which satisfies the ZMP stability criterion. The main difference compared to (Bjelonic et al., 2019) is the inclusion of an optimizer that generates hybrid trajectories for the wheels. Experimental tests are carried out with rough terrain navigation in conditions of bumpy and muddy soil during the DARPA Subterranean Challenge (see Figure 1.17).

In all simulations and experimental tests with ANYmal on wheels, the motion plans are tracked by the WBC developed in (Bjelonic et al., 2019) which is based on a hierarchical optimization framework (Bellicoso et al., 2018). The WBC is formulated as a set of Quadratic Programming (QP) problems composed of linear equality and inequality tasks solved in a prioritized order, presented in Figure 1.18 (priority 1 is the highest). This way, the WBC computes the torque commands for each actuator while taking into account



Figure 1.17: The ANYmal with wheels navigating in an underground environment at the DARPA Subterranean Challenge. Reprinted from (Bjelonic et al., 2020).

the full rigid body dynamics of the robot and several physical constraints, such as the nonholonomic rolling constraint introduced by the wheels, the friction cone, and the actuation torque limits. For details on the implementation of each task, please refer to (Bjelonic et al., 2019).

Priority	Task
1	Floating base equations of motion Torque limits and friction cone Nonholonomic rolling constraint
2	COM linear and angular motion tracking Swing leg motion tracking Swing wheel rotation minimization Ground leg motion tracking
3	Contact force minimization

Figure 1.18: List of prioritized tasks used in the WBC. Tasks in bold are specific for wheeled-legged robots. Reprinted from (Bjelonic et al., 2019).

The WBC tracks the reference trajectories, along with the robot state estimator, at a 400 Hz frequency. ANYmal with wheels features a purely proprioceptive state estimation system based on fusion of IMU and kinematic measurements of the actuators, including the wheels.

Although these current approaches presented good experimental results, the motion planning framework does not take the terrain into account and uses a flat terrain assumption, which violates the validity of the ZMP model when moving over non-flat terrain and renders the approach not amenable for overcoming more challenging terrains, such as steep steps. Indeed, attempts to overcome steep slopes using blind locomotion were not successful, even in simulations, as shown in Figure 1.19.

For planning approaches that require a map of the terrain, a robot-centric elevation map (Fankhauser et al., 2018) can be generated by combining

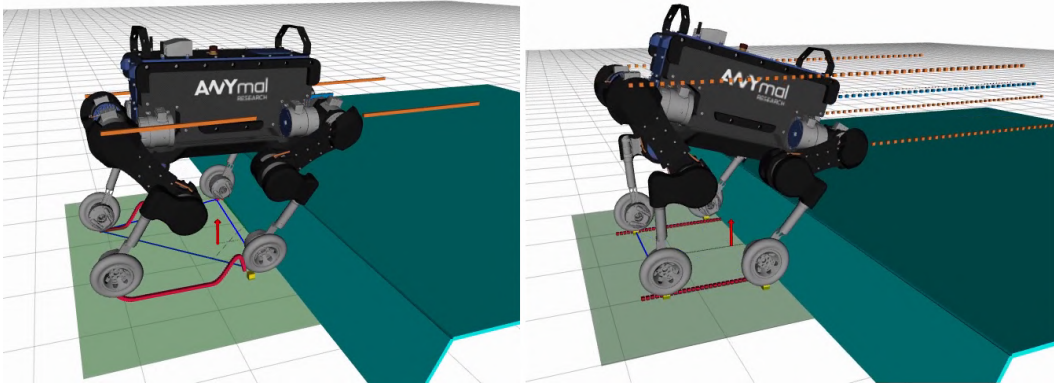


Figure 1.19: Attempts to overcome a step-like obstacle with a 65° slope and 0.2 m height in simulations with purely reactive controller, performing driving motions (right) and hybrid motions (left).

the robot's pose estimation (from Odometry and IMU data) with depth measurements. The elevation map is handled as a grid map, a 2.5-dimensional representation of the environment, where each cell in the grid holds a height value. An example is shown in Figure 1.20 for a step with a linear transition of 65° . From the acquired map, a traversability analysis can be performed based on different terrain characteristics, such as slope, roughness, and step height (Wermelinger et al., 2016).

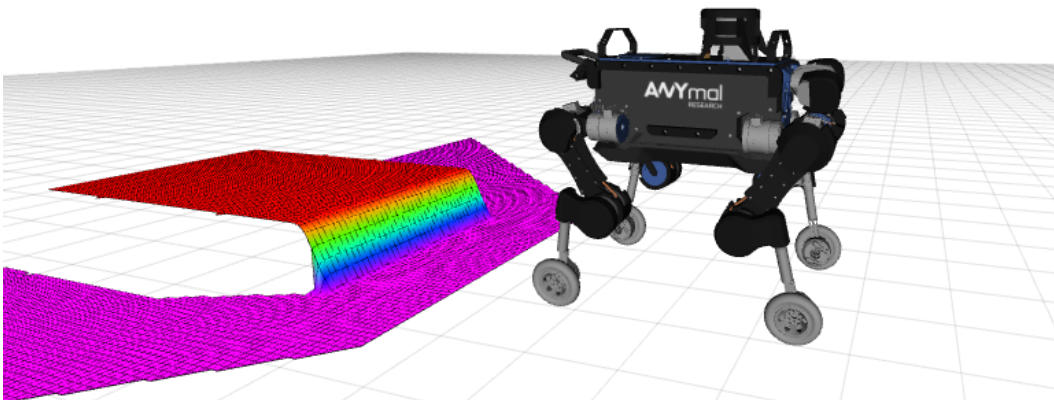


Figure 1.20: The 2.5D height map of the terrain generated from stereo camera data using the ROS package *elevation_mapping* (Fankhauser et al., 2014).

1.3 Contributions

The main contribution of this thesis with respect to the previous work is introducing a motion planning framework for wheeled-legged robots capable of negotiating rough terrain with dynamic motions. On one hand, earlier work

has presented navigation in challenging terrain, but only at low speeds (quasi-static conditions), less than 0.15 m/s. On the other hand, other planning and control algorithms have produced dynamic motions, but employing a flat terrain assumption. This thesis aims to bridge the gap between dynamic motions and motions in rough terrain.

The motion planning and control setup presented in this work breaks the navigation in rough terrain problem into two sub-problems: terrain aware motion planning and perceptive whole-body control. By adding the offline planning component, the robot is able to execute faster motions over steep steps compared to previous approaches. The terrain information is added to the motion control framework to ensure accurate tracking of the planned motions.

The proposed TO framework for wheeled-legged robots optimizes over the 6-DoF base motion (position and orientation) as well as the wheels' positions and contact forces in a single planning problem, accounting for the terrain map and the robot's dynamics. This allows the robot to traverse a variety of challenging terrain, including large steps and drops, with dynamic motions that could not be generated without taking into account the terrain information. Since the contact forces are also decision variables, it is possible to enforce friction constraints and torque limits directly inside the motion planner. A simplified SRBD model is used in the TO, which allows for computing trajectories quickly and for a larger time horizon compared to computing plans with full rigid body dynamics.

Moreover, the approach is general for all terrain types and the robot's base is not restricted to a desired height or orientation, which expands the range of achievable motions, especially for more complex terrains.

This thesis is based on three formulations for wheeled-legged locomotion, each with their dedicated chapter (3, 4, 5). Firstly, a planar trajectory optimization approach is presented for driving motions in conditions where the terrain is symmetric with respect to the traversal direction of motion. The use of a 2D model considerably reduces the computation cost for the approach and allows for the study of suitable cost functions. From that, a general 3D formulation for driving motions is presented with experimental tests on the real robot, showing dynamic driving motions in unstructured terrain. Lastly, a formulation that allows for hybrid locomotion on uneven terrain is developed and discussed. Such approach takes full advantage of the hybrid nature of the system, allowing for dynamic changes in the direction of motion and navigation in discontinuous terrains.

The proposed approaches are evaluated on the robot ANYmal equipped with actuated non-steerable wheels in both simulations and real-world experi-

ments. The results show that ANYmal is able to traverse a variety of terrains with different motions. For driving motions, the testing scenarios include steep inclinations 0.2 m high with 45° and 65° slopes at an average speed of 0.5 m/s and a maximum speed of 0.9 m/s, which is over three times faster than previous approaches for similar obstacles. The hybrid planning framework is verified in several physical simulations that show ANYmal performing hybrid dynamic motions to overcome discontinuous obstacles, such as floating steps and gaps, where instances of wheel's liftoff are necessary. To the best of our knowledge, such a fast negotiation of challenging obstacles by wheeled-legged robots using dynamic motions has not been shown before.

1.4

Thesis Organization

Chapter 2 presents a review on dynamic models for wheeled-legged robots with different levels of simplifications, and a review on the general formulation for TO problems, along with the advantages and disadvantages of the existing solving techniques.

In Chapter 3, an initial formulation of the TO problem is presented to generate dynamic driving motions in challenging terrain using a simplified two-dimensional model of the robot. Chapter 4 extends the formulation for general 3D driving motions in symmetric and asymmetric terrain types.

Chapter 5 presents a TO formulation for hybrid dynamic motions that generates the foot and base trajectories for the robot simultaneously. This way, the planner can generate trajectories with purely driving and purely stepping motions or it can perform both motions at the same time, enabling the negotiation of discontinuous obstacles.

Lastly, Chapter 6 brings suggestions for future work and the conclusions of this thesis.

2

Modeling of Wheeled-Legged Robots

In general, the dynamics of most physical systems is mathematically modeled by a set of Ordinary Differential Equations (ODEs):

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad (2-1)$$

where $\mathbf{x}(t)$ represents the state \mathbf{x} of the system at a given time t and $\mathbf{u}(t)$ is the control input of the system.

Dynamic models have a important role in control, simulation and motion planning for robotic systems. A motion plan is considered physically accurate if the modeled dynamic equations are satisfied at all times. For that reason, this chapter presents a review on dynamic models for wheeled-legged systems along with the constraints and assumptions required to make them physically correct. Furthermore, a review on the general formulation for TO problems and existent solving techniques is presented.

2.1

Full Rigid Body Dynamics

Similar to legged robots, wheeled-legged robots can be modeled as floating-base systems, i.e., the motion of the base cannot be directly controlled through actuators but through external forces resulting from contacts (Hutter et al., 2017). To model such systems, a fictitious fixed base is introduced, and then connected to the floating base via a fictitious joint having six degrees of freedom, as illustrated in Figure 2.1. The six additional joint variables are used to define the position and orientation of the floating base. With the virtual un-actuated base joints, the Equations of Motion (EoM) can be computed via the well known *Newton-Euler* method, which applies the principles of conservation of linear and angular momentum for all links of the robot (Siciliano et al., 2010).

2.1.1

Equations of Motion

Floating base robots can be fully described by the n_b un-actuated base coordinates \mathbf{q}_b and n_j actuated joint coordinates \mathbf{q}_j :

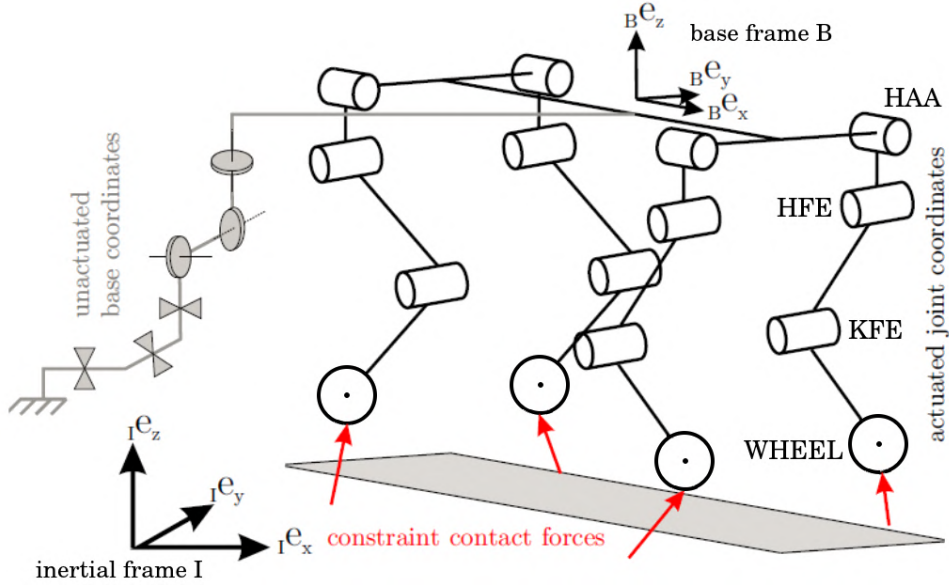


Figure 2.1: Representation of the actuation joints of a wheeled quadrupedal robot, modeled as a floating base B to which the legs and wheels are attached. The coordinate frames are the fixed inertial frame I and the frame B , attached to un-actuated base. Figure adapted from (Hutter et al., 2017).

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_b \\ \mathbf{q}_j \end{bmatrix} \quad (2-2)$$

The floating base is free in translation and rotation:

$$\mathbf{q}_b = \begin{bmatrix} {}^I \mathbf{r}_{IB} \\ \Phi_{IB} \end{bmatrix} \in \mathbb{R}^3 \times SO(3)^1, \quad (2-3)$$

where ${}^I \mathbf{r}_B \in \mathbb{R}^3$ is the position of the base from frame I to frame B with respect to the inertial frame I , and the base orientation Φ_{IB} can be parametrized in different ways, such as *Euler angles* or *unit quaternions* (Siciliano et al., 2010). The number of generalized coordinates of the floating base system is given by $n_b + n_j$ and it depends on the parametrization chosen for the base's rotation, where the minimal number of generalized coordinates for the base is 6.

The generalized velocity vector \mathbf{v} of the system is given by

$$\mathbf{v} = \begin{bmatrix} {}^I \mathbf{v}_B \\ {}^B \boldsymbol{\omega}_{IB} \\ \dot{\mathbf{q}}_j \end{bmatrix} \in \mathbb{R}^{n_v}, \quad (2-4)$$

where ${}^I \mathbf{v}_B \in \mathbb{R}^3$ is the linear velocity of the base B with respect to inertial frame I , and ${}^B \boldsymbol{\omega}_{IB} \in \mathbb{R}^3$ is the angular velocity from frame I to frame B with respect to the frame B . The number of generalized velocity coordinates

¹The special orthogonal Euclidian group that represents rotations in three dimensions.

is $n_v = 6 + n_j$.

Once defined the generalized position and velocity vectors of the system, the dynamics of a floating base rigid-body robot is given by

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{b}(\mathbf{q}, \mathbf{v}) + \mathbf{g}(\mathbf{q}) = \mathbf{S}^T \boldsymbol{\tau} + \mathbf{J}_c^T \mathbf{F}_c, \quad (2-5)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n_v \times n_v}$ is the generalized mass matrix, $\mathbf{b}(\mathbf{q}, \mathbf{v}) \in \mathbb{R}^{n_v}$ is the vector of Coriolis and centrifugal terms and $\mathbf{g}(\mathbf{q})$ is the vector of gravitational terms. $\boldsymbol{\tau} \in \mathbb{R}^{n_j}$ is the vector of generalized torques acting in the direction of the generalized coordinates and the selection matrix $\mathbf{S} = \begin{bmatrix} \mathbf{0}_{n_j \times 6} & \mathbf{1}_{n_j \times n_j} \end{bmatrix}$ selects which joints are actuated. Considering n_c the number of end-effectors in contact, the Jacobian $\mathbf{J}_c = \begin{bmatrix} \mathbf{J}_{C_1}^T & \dots & \mathbf{J}_{C_{n_c}}^T \end{bmatrix}^T \in \mathbb{R}^{3n_c \times n_v}$ maps the contact forces $\mathbf{F}_c = \begin{bmatrix} \mathbf{f}_1^T & \dots & \mathbf{f}_{n_c}^T \end{bmatrix}^T \in \mathbb{R}^{3n_c}$ from the Cartesian space onto the subspace of the generalized coordinates. The Jacobians are derived from the kinematics of the robot, that can be computed using the algorithms presented in (Featherstone, 2007).

Equation (2-5) can be re-written to separate the 6 floating base accelerations and the n_j joint accelerations, as follows:

$$\begin{bmatrix} \mathbf{M}_{bb} & \mathbf{M}_{bj} \\ \mathbf{M}_{jb} & \mathbf{M}_{jj} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}}_b \\ \dot{\mathbf{v}}_j \end{bmatrix} + \begin{bmatrix} \mathbf{b}_b \\ \mathbf{b}_j \end{bmatrix} + \begin{bmatrix} \mathbf{g}_b \\ \mathbf{g}_j \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \mathbf{J}_{c,b}^T \\ \mathbf{J}_{c,j}^T \end{bmatrix} \mathbf{F}_c, \quad (2-6)$$

The first matricial row corresponds to the equations for the un-actuated base motion and the second describes the joint motions, whereby $\mathbf{J}_{c,b}$ indicates the relation between the base motion and the contact forces. Note that the torques from the actuated joints $\boldsymbol{\tau}$ cannot directly influence the base motion and the forces \mathbf{F}_c originated from ground contact are responsible for moving the base.

In this model, the state of the robot is given by the generalized coordinates and velocities for the un-actuated base and joints $\mathbf{x} = \begin{bmatrix} \mathbf{q}^T & \mathbf{v}^T \end{bmatrix}^T$, while the inputs are all the joint torques and the contact forces $\mathbf{u} = \begin{bmatrix} \boldsymbol{\tau}^T & \mathbf{F}_c^T \end{bmatrix}^T$.

2.1.2 Contact Model

Two different methods can be used to model the contact forces. The *soft contact* method models the force as a function of the location and velocity of the contact point, e.g., a spring-damper contact force. This method requires the tuning of the contact parameters for stiffness and damping, which is very difficult to be done accurately.

The alternative is to use the *hard contact* method, in which the contacts are modeled as kinematic constraints (Hutter et al., 2017). For a legged robot with point feet, for instance, if a point C with position \mathbf{r}_c is in contact, it is not allowed to move anymore, which implies in the following kinematic constraints:

$$\dot{\mathbf{r}}_c = \mathbf{J}_c \mathbf{v} = \mathbf{0} \quad (2-7)$$

$$\ddot{\mathbf{r}}_c = \mathbf{J}_c \dot{\mathbf{v}} + \dot{\mathbf{J}}_c \mathbf{v} = \mathbf{0} \quad (2-8)$$

Substituting $\dot{\mathbf{v}}$ in (2-8) with the equations of motion in (2-5), the constraint can be written as

$$\ddot{\mathbf{r}}_c = \mathbf{J}_c \mathbf{M}^{-1} \left(\mathbf{S}^T \boldsymbol{\tau} + \mathbf{J}_c^T \mathbf{F}_c - (\mathbf{b} + \mathbf{g}) \right) + \dot{\mathbf{J}}_c \mathbf{v} = \mathbf{0} \quad (2-9)$$

Then, solving (2-9) for \mathbf{F}_c , the contact force is given by:

$$\mathbf{F}_c = \left(\mathbf{J}_c \mathbf{M}^{-1} \mathbf{J}_c^T \right)^{-1} \left(\mathbf{J}_c \mathbf{M}^{-1} \left(\mathbf{S}^T \boldsymbol{\tau} - (\mathbf{b} + \mathbf{g}) \right) + \dot{\mathbf{J}}_c \mathbf{v} \right) \quad (2-10)$$

As an advantage, this contact model allows for the estimation of the contact forces based on the system dynamics alone, without a further contact force sensor.

Unlike legged robots with point feet, wheeled-legged robots have additional degrees of freedom due to the wheels as end-effectors and the acceleration of the contact point is not zero, which introduces a nonholonomic rolling constraint on the system dynamics, modeled in (Bjelonic et al., 2019) and reviewed next.

2.1.3 Nonholonomic Rolling Constraint

The wheel model used to derive the rolling constraint is illustrated in Figure 2.2. In addition to the inertial frame I and the base frame B , leg-fixed and wheel-fixed coordinate frames are added to the wheel's center and the wheel's contact point. The leg-fixed wheel frame W'_i is located at the wheel's center and does not depend on the joint angle θ_w of the wheel, while the wheel-fixed frame W_i rotates with the wheel and depends on θ_w .

Similarly, the leg-fixed contact frame C'_i is defined with respect to the frame W'_i and does not depend on θ_w . Also, it does not need to have zero velocity, reflecting the motion of the wheel's contact point. In contrast, the wheel-fixed contact frame C_i is defined with respect to the frame W_i and it depends on the joint angle θ_w of the wheel. For both contact frames, the z-

direction is aligned with the terrain normal \mathbf{n} and the x-direction is aligned with the rolling direction of the wheel.

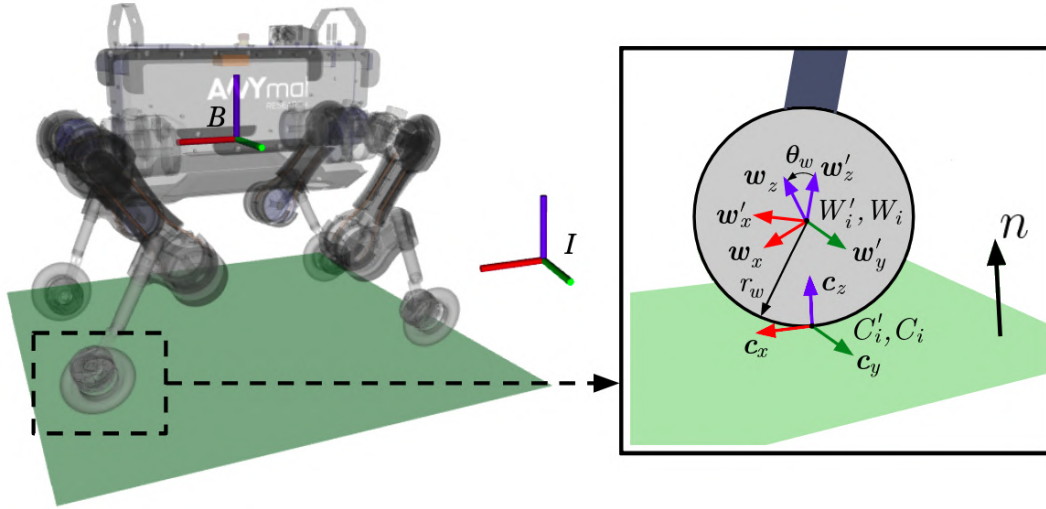


Figure 2.2: The coordinate frames of the robot's wheel.

In contrast to fixed contact points from legged locomotion, the acceleration of the i^{th} wheel contact point with respect to the inertial frame is not constraint to be zero, i.e., ${}^I\ddot{\mathbf{r}}_{IC_i} = \mathbf{J}_{C_i}\dot{\mathbf{v}} + \dot{\mathbf{J}}_{C_i}\mathbf{v} \neq \mathbf{0}$. From (Bjelonic et al., 2019), the resulting contact acceleration is given by

$${}^I\ddot{\mathbf{r}}_{IC_i} = \mathbf{J}_{C_i}\dot{\mathbf{v}} + \dot{\mathbf{J}}_{C_i}\mathbf{v} = \mathbf{R}_{IW_i} \begin{bmatrix} 0 \\ -r_w\dot{\psi}_{IW_i'} \cos(\phi_{IW_i'}) (\dot{\chi}_{IW_i'} + \dot{\theta}_w) \\ r_w(\dot{\chi}_{IW_i'} + \dot{\theta}_w) (\dot{\chi}_{IW_i'} + \dot{\theta}_w + \dot{\psi}_{IW_i'} \sin(\phi_{IW_i'})) \end{bmatrix}, \quad (2-11)$$

where $\mathbf{R}_{IW_i} \in SO(3)$ is the rotation matrix that projects a vector from the wheel frame W_i to the inertial frame I and r_w is the wheel radius. Using an Euler parametrization, the yaw, roll and pitch angle of the leg-fixed frame W_i' with respect to the initial frame I are given by $\psi_{IW_i'}$, $\phi_{IW_i'}$, and $\chi_{IW_i'}$, respectively.

Note that by setting $\phi_{IW_i'}$ and $\psi_{IW_i'}$ to zero, the contact point acceleration is reduced to $\mathbf{R}_{IW_i} \begin{bmatrix} 0 & 0 & r_w(\dot{\chi}_{IW_i'} + \dot{\theta}_w)^2 \end{bmatrix}^T$, which is the centripetal acceleration of the wheel. Moreover, the acceleration in the x-direction is zero since there is no slippage in the wheels.

As an additional comment, the nonholonomic rolling constraint included in the dynamics model implies that a respective rolling constraint must be considered in the motion planning framework to ensure the desired wheel motion is executable.

The full RBD model of wheeled-legged robots, including the contact constraints, is important for a realistic simulation of their locomotion and for computing the actuation torques for the motion control system. However, the RBD is highly complex due to its many degrees of freedom. In particular for motion planning algorithms, the use of a simplified dynamic model makes the formulation simpler and allows for faster convergence of the optimizer compared to using the full RBD.

2.2 Single Rigid Body Dynamics

Wheeled-legged robots can be faithfully represented as a single rigid body with fixed mass and inertia located at the robot's CoM, as depicted in Figure 2.3 for a quadrupedal wheeled robot.

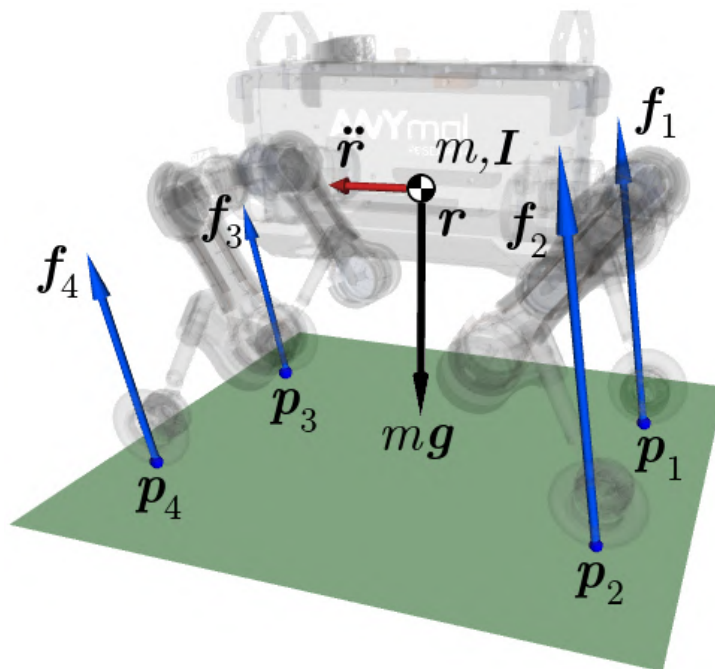


Figure 2.3: Single rigid body representation of a quadrupedal robot. The gray overlaid model of the ANYmal robot with wheels is only for visualization and it is not included in the dynamics, all the system's mass is located at the robot's CoM.

This model assumes that the mass and inertia of the legs are negligible compared to the robot's base and do not contribute significantly to the momentum of the system. In fact, this assumption is reasonable for most quadrupedal robots since a large portion of the mass is in the base of the robot. Each leg is up to an order of magnitude lighter than the base.

With this assumption, the EoM of the system are defined by the Newton-Euler Equations of a single rigid body, in which the robot's CoM linear

acceleration $\ddot{\mathbf{r}}(t) \in \mathbb{R}^3$ is determined by the sum of all the contact forces on the wheels and the gravitational force; and the CoM angular acceleration $\dot{\boldsymbol{\omega}}(t) \in \mathbb{R}^3$ is given by the Euler's rotation equation (Siciliano et al., 2010), described as

$$\begin{aligned} m\ddot{\mathbf{r}}(t) &= \sum_{i=1}^{n_i} \mathbf{f}_i(t) - m\mathbf{g} \\ \mathbf{I}\dot{\boldsymbol{\omega}}(t) + \boldsymbol{\omega}(t) \times \mathbf{I}\boldsymbol{\omega}(t) &= \sum_{i=1}^{n_i} \mathbf{f}_i(t) \times (\mathbf{r}(t) - \mathbf{p}_i(t)) \end{aligned} \quad (2-12)$$

where $\boldsymbol{\omega}(t) \in \mathbb{R}^3$ represents the angular velocity of the robot's base, m denotes the robot's mass, $\mathbf{g} \in \mathbb{R}^3$ the gravity vector and $\mathbf{I} \in \mathbb{R}^{3 \times 3}$ the inertia matrix of the robot, computed around the nominal stance position and assumed fixed. The number of end-effectors is given by n_i , and the contact force and contact position for the i^{th} wheel are given by \mathbf{f}_i and \mathbf{p}_i , respectively.

The SRBD model makes the dynamics of the robot independent of the joint configuration of the legs, reducing the number of variables in the robot's state, while keeping the dynamic effects important to locomotion. The state of the robot is given by the CoM's position \mathbf{r} and orientation $\boldsymbol{\theta}$, its linear and angular velocities, in addition to the wheels' positions and velocities $\mathbf{x} = \left[\mathbf{r}^T \quad \boldsymbol{\theta}^T \quad \dot{\mathbf{r}}^T \quad \boldsymbol{\omega}^T \quad \left[\mathbf{p}_1^T \quad \dots \quad \mathbf{p}_{n_i}^T \right]^T \quad \left[\dot{\mathbf{p}}_1^T \quad \dots \quad \dot{\mathbf{p}}_{n_i}^T \right]^T \right]^T$, while the control inputs are the wheels' contact forces $\mathbf{u} = \left[\mathbf{f}_1^T \quad \dots \quad \mathbf{f}_{n_i}^T \right]^T$. This simplified model is particularly useful for trajectory optimization problems, where reduced computational cost is desirable.

2.3 Trajectory Optimization

The goal of a TO problem is to find a trajectory for some dynamic system that satisfies a set of constraints while minimizing some cost functional. In general, this is achieved by solving the Optimal Control Problem (OCP) described as:

$$\begin{aligned} &\underset{\mathbf{x}, \mathbf{u} \in \mathcal{S}}{\text{minimize}} && J(\mathbf{x}(t), \mathbf{u}(t)) \\ &\text{subject to} && \mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f, \\ &&& \mathbf{c}(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)) \leq \mathbf{0}, \\ &&& \mathbf{h}(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)) = \mathbf{0}, \end{aligned} \quad (2-13)$$

where \mathbf{h} and \mathbf{c} are respectively the set of equality and inequality constraints for the problem, \mathbf{x}_0 and \mathbf{x}_f are initial and final conditions for the state \mathbf{x} , and t_f is the final time for the trajectory. The main equality constraint is the system dynamics, i.e., $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$.

Three different techniques can be considered for solving this OCP: Dynamic Programming, Indirect Methods and Direct Methods (Diehl et al., 2006). The first approach, Dynamic Programming, uses the Bellman principle of optimality (Bertsekas, 2005) to compute recursively a global control policy for all times t and all initial states. It is most commonly applied to unconstrained low-dimensional systems since it requires a discretization of the full state space.

Indirect Methods attempt to solve the OCP by using the “optimize then discretize” philosophy. It requires explicit derivation of the necessary conditions of optimality (Betts, 2009) into a Boundary Value Problem (BVP) which is then solved using a numerical solver. The main strength of such approaches is that they provide solutions that are optimal with respect to the original optimization problem. However, there are several drawbacks. Inequality constraints are particularly difficult to handle using indirect methods, and finding a reasonable initial guess for the problem is quite challenging (Betts, 2009). In addition, such methods tend to be numerically unstable and difficult to implement.

On the other hand, the Direct Method employs the “discretize then optimize” approach by converting the continuous time problem into a finite dimensional NLP, which definition is

$$\begin{aligned}
 & \underset{\boldsymbol{\xi} \in \mathbb{R}^n}{\text{minimize}} && \bar{J}(\boldsymbol{\xi}) \\
 & \text{subject to} && \bar{\mathbf{c}}(\boldsymbol{\xi}) \leq \mathbf{0} \\
 & && \bar{\mathbf{h}}(\boldsymbol{\xi}) = \mathbf{0} \\
 & && \boldsymbol{\xi}_L \leq \boldsymbol{\xi} \leq \boldsymbol{\xi}_U
 \end{aligned} \tag{2-14}$$

where $\boldsymbol{\xi} \in \mathbb{R}^n$ is the set of optimization variables, and $\boldsymbol{\xi}_U$ and $\boldsymbol{\xi}_L$ are respectively the upper and lower bounds for the variables.

The resulting NLP is then solved by state-of-the-art numerical optimization methods. Algorithms for solving NLP problems include Interior Point (Mehrotra, 1992), Sequential Quadratic Programming (SQP) and Trust-Region (Nocedal and Wright, 2006).

The advantages of the Direct Method are several: the formulation and treatment of the inequality constraints is easier; the initialization is more intuitive since the decision variables are physical quantities; and there is a broad options of NLP numerical solvers available for efficiently solving large scale problems, which is specially important for long prediction horizons in complex terrains. The drawback is that such methods are commonly prone to local minima and they only guarantee a locally optimal solution.

Having considered the advantages and disadvantages of the aforementioned solving techniques, this work will focus on the application of Direct Methods to solve the TO problem.

2.3.1

Transcription Methods

There are several methods for transcribing the TO problem into a NLP. In general, they can be divided into two broad categories: *sequential methods* and *simultaneous methods*. In sequential approaches, only the control input $\mathbf{u}(t)$ is parametrized through a set of discrete variables. The input is then applied to dynamic system and the state trajectory $\mathbf{x}(t)$ is computed by a forward simulation with the help of an ODE solver. The NLP optimizes over the discretized control input and verifies if the resulting states fulfill all the constraints. Thus, simulation and optimization iterations proceed sequentially, one after the other, until the control input is found for which the state trajectory minimizes the cost and respects the constraints. In this approach, the NLP is relatively smaller since only the control input is optimized, reducing the computational cost. However, ill-conditioned problems can be difficult to handle, since small changes in the control input can cause large variations in the state trajectory (Kelly, 2017a).

In simultaneous methods, both the input control $\mathbf{u}(t)$ and the states $\mathbf{x}(t)$ are parameterized over time and the NLP optimizes over $\mathbf{u}(t)$ and $\mathbf{x}(t)$ simultaneously. The system dynamics is no longer simulated, but enforced through nonlinear equality constraints in the optimization, evaluated only at special points along the trajectory. Such methods can be initialized with a guess for the state trajectory $\mathbf{x}(t)$, which may be easier to determine than an initial $\mathbf{u}(t)$. The problem is relatively larger, but changes in the input only affects the state at that time instance, which reflects in the sparse structure of the problem and allows for faster solutions (Posa et al., 2014).

A commonly used simultaneous method is the *Direct Collocation* (Hargraves and Paris, 1987). The state trajectory and the control input are represented as piecewise cubic polynomials and the system dynamics and constraints are enforced at the knots (discretization points) between the polynomials. The decision variables in the optimization are the values for the control and state at each knot point. Once the state values and derivatives are determined for all knots, the cubic polynomials between them are fully defined. The resulting trajectory is guaranteed to be continuous and to have continuous first derivative. This is the transcription method used in this thesis to formulate the 3D trajectory optimization problem.

3 2D Trajectory Optimization

Wheeled-legged robots offer a solution for versatile locomotion in unstructured terrain. For such systems, driving motions are particularly interesting, since they allow for faster and more stable locomotion, and obstacle negotiation is enabled through the presence of the legs.

When planning for driving motions for quadrupedal wheeled robots, common scenarios often include straight trajectories in terrains with variations only in the longitudinal direction. Such scenarios can benefit from a simplification of the planning problem and can be handled with a two-dimensional approach. To this end, an initial formulation of the TO problem is presented for a simplified 2D model of the robot.

Such formulation presents several advantages:

- Considerably lower computational cost, which allows for fast solutions even for trajectories with a long time horizon;
- Easier tuning and assessment of the optimization parameters;
- Simple visualization of the motions with MATLAB®;
- Reduced implementation effort, which helps with the study of suitable stability metrics and cost functions.

In addition, a simplified planar trajectory optimization framework for wheeled-legged robots can be useful for teaching purposes.

3.1 2D Robot Model

For wheeled-legged robots to move safely on challenging terrain, it is important that the motion planner produces only trajectories that are physically feasible and dynamically stable. To ensure that, the trajectory optimization must take into account how the wheels' positions and forces affect the robot's state. In this approach, the robot is approximated by a single rigid-body with mass and inertia located at the robot's CoM, as illustrated in Figure 3.1.

In the 2D formulation, the wheels are assumed to be skid-steered, so only forces in the XZ plane are considered. The displacement in the y -direction,

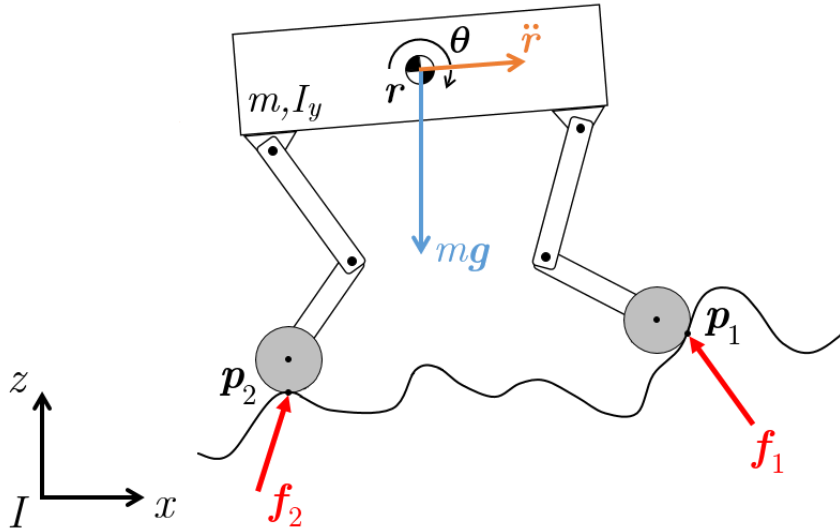


Figure 3.1: Planar model of the ANYmal robot traversing rough terrain.

and the roll and yaw angles of the robot's base are assumed to be zero for the entire motion. Moreover, we assume that the contact between the wheel and the ground can be reduced to a single virtual contact point and the friction coefficient of the soil is known or can be estimated.

Using this simplified model, the dynamics of the robot is given by:

$$\begin{bmatrix} m\ddot{\mathbf{r}} \\ I_y\ddot{\theta} \end{bmatrix} = \begin{bmatrix} \mathbf{1}_{2 \times 2} & \\ & \mathbf{1}_{2 \times 2} \end{bmatrix} \begin{bmatrix} (r^z - p_1^z) & -(r^x - p_1^x) \\ (r^z - p_2^z) & -(r^x - p_2^x) \end{bmatrix} \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} - \begin{bmatrix} m\mathbf{g} \\ 0 \end{bmatrix} \quad (3-1)$$

where the CoM position and orientation are given by \mathbf{r} and θ , respectively. The contact points on the wheels are given by \mathbf{p}_1 and \mathbf{p}_2 , m denotes the robot's mass, \mathbf{g} the gravity vector and I_y the inertia of the robot around the Y axis. The contact forces on each wheel are given by \mathbf{f}_1 and \mathbf{f}_2 . All the forces and positions vectors are defined with respect to the inertial frame, in \mathbb{R}^2 . The orientation is simply the pitch angle of the base, defined in \mathbb{R} . The right superscript denotes a component of the vector, e.g., p_i^x represents the the x component of the contact position of the i^{th} wheel.

For the trajectory optimization, the terrain profile must be provided to the planner. With the terrain information, the contact force on each wheel \mathbf{f}_i can be accurately decomposed into a normal force N_i and a tangential force F_{T_i} (traction), considering a local frame C_i located in the point of contact between the wheel and the ground, as shown in Figure 3.2. Assuming no slippage in the wheels, which is enforced during the optimization, the torque τ in each wheel can be obtained by multiplying the traction force for the wheel radius r_w .

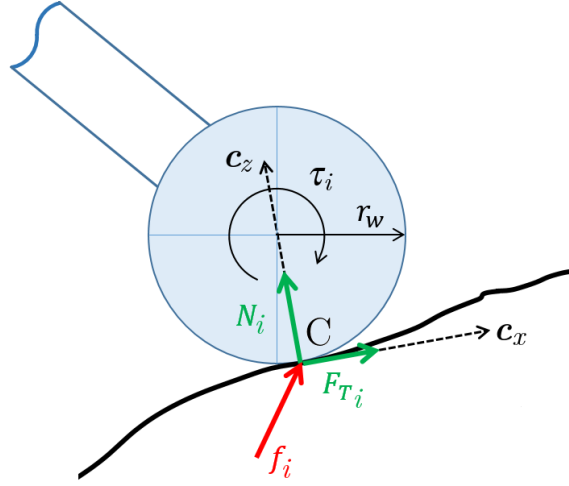


Figure 3.2: Contact forces on each wheel, considering a frame C_i located at the wheel's contact point with the terrain. The axis \mathbf{c}_z is aligned with the terrain normal and \mathbf{c}_x is aligned with the rolling direction of the wheel.

Given the terrain normal at the i^{th} wheel contact point $\mathbf{n}_i = [n_i^x \ n_i^z]^T \in \mathbb{R}^2$, the 2D rotation matrix $\mathbf{R}_{C_i I}$ from the inertial frame I to the contact frame C_i is given by:

$$\begin{aligned} {}^{C_i} \mathbf{f}_i &= \mathbf{R}_{C_i I} {}^I \mathbf{f}_i \\ {}^{C_i} \mathbf{f}_i &= \begin{bmatrix} F_{T_i} \\ N_i \end{bmatrix} = \begin{bmatrix} n_i^z & -n_i^x \\ n_i^x & n_i^z \end{bmatrix} {}^I \mathbf{f}_i \end{aligned} \quad (3-2)$$

3.2 Problem Formulation

The trajectory optimization is formulated as an OCP of the form:

$$\begin{aligned} & \underset{\mathbf{x}(t), \mathbf{u}(t)}{\text{minimize}} && J(\mathbf{x}(t), \mathbf{u}(t)) \\ & \text{subject to} && \mathbf{c}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \\ & && \mathbf{h}(\mathbf{x}(t), \mathbf{u}(t)) = \mathbf{0}, \\ & && \mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(T) = \mathbf{x}_f, \end{aligned} \quad (3-3)$$

where $\mathbf{x}(t)$ is the set of variables that defines the motion of the robot's CoM and the wheel's contact positions

$$\mathbf{x}(t) = [\mathbf{r}(t) \ \dot{\mathbf{r}}(t) \ \theta(t) \ \dot{\theta}(t) \ \mathbf{p}_{1,2}(t) \ \dot{\mathbf{p}}_{1,2}(t)]^T,$$

$\mathbf{u}(t)$ are the contact forces on the wheels $\mathbf{f}_{1,2}(t)$, \mathbf{x}_0 and \mathbf{x}_f are respectively the initial and final state of the robot, and T is the total time duration of the trajectory.

Firstly, this continuous-time optimization problem needs to be formulated as an NLP, with a finite number of decision variables, as follows

$$\begin{aligned}
 & \underset{\boldsymbol{\xi} \in \mathbb{R}^n}{\text{minimize}} && \bar{J}(\boldsymbol{\xi}) \\
 & \text{subject to} && \bar{\mathbf{c}}(\boldsymbol{\xi}) \leq \mathbf{0} \\
 & && \bar{\mathbf{h}}(\boldsymbol{\xi}) = \mathbf{0} \\
 & && \boldsymbol{\xi}_L \leq \boldsymbol{\xi} \leq \boldsymbol{\xi}_U
 \end{aligned} \tag{3-4}$$

For that, the time horizon is discretized in fixed time intervals ΔT , including the initial state, generating $n = \text{floor}(T/\Delta T) + 1$ nodes, as illustrated in Figure 3.3. Each node k is defined by the optimization variables at the time $t_k = k\Delta T$ of the trajectory. Thus, the NLP decision variables are the values for the robot's state and contact forces on all nodes:

$$\boldsymbol{\xi} = [\mathbf{u}_0 \quad \mathbf{x}_0 \quad \mathbf{u}_1 \quad \mathbf{x}_1 \quad \dots \quad \mathbf{u}_{n-1} \quad \mathbf{x}_{n-1}]^T \tag{3-5}$$

where $\mathbf{u}_k = \mathbf{u}(t_k)$ and $\mathbf{x}_k = \mathbf{x}(t_k)$.

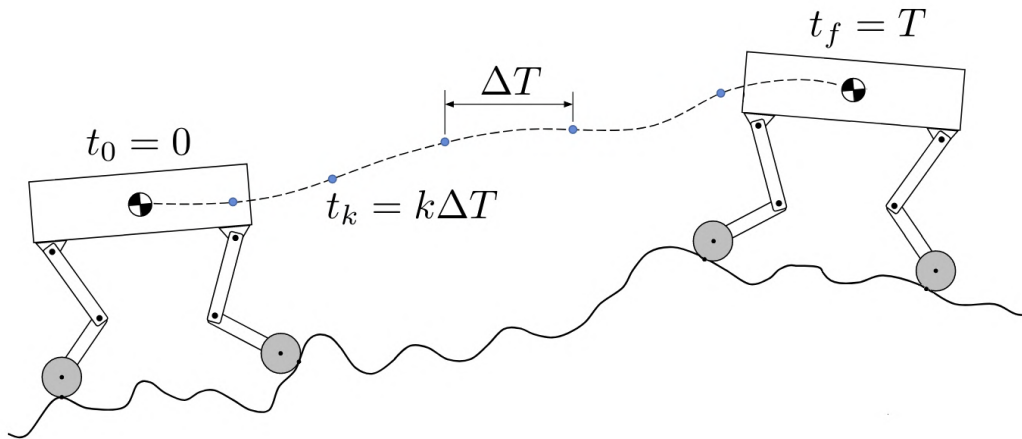


Figure 3.3: Illustration of the discretization points (nodes) for the robot's trajectory.

In order to ensure the consistency of the derivatives, the following constraints are imposed between the nodes:

$$\dot{\mathbf{q}}_k = \frac{\mathbf{q}_{k+1} - \mathbf{q}_k}{\Delta T}, \quad k = 0, \dots, n-2, \tag{3-6}$$

where $\mathbf{q}_k = [\mathbf{r} \quad \theta \quad \mathbf{p}_1 \quad \mathbf{p}_2]_k^T$, the set of position variables at node k .

Having defined the decision variables for the discretized problem, the most straightforward approach is to enforce the equality and inequality constraint at all nodes:

$$\bar{\mathbf{c}}(\boldsymbol{\xi}_k) \leq \mathbf{0}, \quad k = 0, \dots, n-1, \quad (3-7)$$

$$\bar{\mathbf{h}}(\boldsymbol{\xi}_k) = \mathbf{0}, \quad k = 0, \dots, n-1, \quad (3-8)$$

where $\boldsymbol{\xi}_k = [\mathbf{u}_k \quad \mathbf{x}_k]^T$.

Note that such an approach does not guarantee that the constraints are satisfied on the entire interval between the nodes $[t_k, t_{k+1}]$, but only at the discretization points. Once the problem is solved, the continuous solution for each variable is obtained by a simple linear interpolation between the nodes, which gives a feasible continuous trajectory, assuming ΔT sufficiently small.

The complete TO formulation for the 2D motion optimization problem with all the constraints is presented in Figure 3.4. For all the variables, the right superscript denotes a component of the vector and the right subscript indicates which wheel is being considered. The optimization constraints are discussed in details in the next section.

find	$\mathbf{r}(t), \dot{\mathbf{r}}(t) \in \mathbb{R}^2$	(CoM linear position)
	$\theta(t), \dot{\theta}(t) \in \mathbb{R}$	(CoM angular position)
	$\mathbf{p}_{1,2}(t), \dot{\mathbf{p}}_{1,2}(t) \in \mathbb{R}^2$	(wheels' contact positions)
	$\mathbf{f}_{1,2}(t) \in \mathbb{R}^2$	(wheels' contact forces)
s.t.	$[\mathbf{r}, \theta](0) = [\mathbf{r}_0, \theta_0]$	(initial state)
	$[\mathbf{r}, \theta](T) = [\mathbf{r}_g, \theta_g]$	(goal state)
	$[\ddot{\mathbf{r}}, \ddot{\theta}]^T = \mathbf{F}_d(\mathbf{r}, \theta, \mathbf{p}_{1,2}, \mathbf{f}_{1,2}),$	(dynamic model)
	$\alpha(\mathbf{r}, \theta, \mathbf{p}_{1,2}, \mathbf{f}_{1,2}) > \alpha_{min},$	(stability measure)
	$\mathbf{p}_i(t) \in \mathcal{R}_i(\mathbf{r}(t), \theta(t)), \quad i = 1, 2$	(kinematic constraints)
	$p_i^z(t) = h_{terrain}(p_i^x(t)), \quad i = 1, 2$	(terrain height)
	$N_i(t) \geq 0, \quad i = 1, 2$	(normal force)
	$F_{Ti}(t) \leq \tau_{max}/r_w, \quad i = 1, 2$	(maximum torque)
	$-\mu N_i(t) \leq F_{Ti}(t) \leq \mu N_i(t), \quad i = 1, 2$	(no slippage)

Figure 3.4: Decision variables and constraints of the two-dimensional TO formulation.

As an additional note, a similar formulation can also be employed for torque optimization in hazardous conditions, in which the robot is close to losing stability and tipping over, as demonstrated for wheeled robots in (Medeiros et al., 2019). In this case, the contact positions and the position of the CoM are provided by the user and the solver computes the optimal wheels' torques for improving stability.

3.2.1 Constraints

The 2D formulation is only applicable for generating driving motions, which implies that the wheels are in contact with the terrain during the entire trajectory. This can be translated in the problem formulation as constraints on the wheel's contact positions and on the normal contact forces, as:

$$p_i^z = h_{terrain}(p_i^x), \quad i = 1, 2 \quad (3-9)$$

$$N_i \geq 0, \quad i = 1, 2 \quad (3-10)$$

where $h_{terrain}$ is the continuous 2D height map of the terrain. The normal component N_i is extracted from the contact force \mathbf{f}_i by applying the transform presented in (3-2). For that, the terrain normal at the contact position $\mathbf{n}(p_i^x)$ is obtained through the derivative of the continuous terrain map at that point.

In order to ensure no slippage in the wheels, the traction force is constrained to be less or equal than the normal force times the friction coefficient μ of the terrain:

$$-\mu N_i \leq F_{T_i} \leq \mu N_i, \quad i = 1, 2 \quad (3-11)$$

where F_{T_i} is obtained from (3-2) as well.

Additionally, the traction forces are limited to a saturation value correspondent to the maximum torque of the wheel's motor:

$$-\frac{\tau_{max}}{r_w} \leq F_{T_i} \leq \frac{\tau_{max}}{r_w}, \quad i = 1, 2 \quad (3-12)$$

The dynamic consistency of the trajectory is ensured by imposing the equations (3-1) to all nodes. For that, the base accelerations are computed via numerical differentiation of the base velocities.

$$\ddot{\mathbf{r}}_k = \frac{\dot{\mathbf{r}}_{k+1} - \dot{\mathbf{r}}_k}{\Delta T}, \quad k = 0, \dots, n-2, \quad (3-13)$$

$$\ddot{\theta}_k = \frac{\dot{\theta}_{k+1} - \dot{\theta}_k}{\Delta T}, \quad k = 0, \dots, n-2, \quad (3-14)$$

The base accelerations are also constrained to a maximum value for avoiding abrupt motions. In most cases, the velocities for the initial and final nodes are constrained to be zero to prevent any feedforward torque at the beginning or end of the motion.

An additional requirement for motion consistency is to ensure the wheels' positions remain within the reachable workspace of the legs. The exact way to do this is to enforce the joint limits and compute the end-effector position

through forward kinematics. However, forward kinematics introduces more non-linear constraints on the formulation, which could affect the solver's performance and should be avoided. Instead, the feasible workspace for each wheel is represented by a rectangle of fixed size centered in the nominal position of the wheels relative to the robot's base, similar to (Winkler et al., 2018; Bjelonic et al., 2020). The size of the rectangle is defined based on the robot's joints limits, as depicted by the cyan-marked region in Figure 3.5. Thus, the constraint is computed as:

$$-\mathbf{b}/2 < [\mathbf{R}(\theta)(\mathbf{p}_i - \mathbf{r}) - \bar{\mathbf{p}}_i] < \mathbf{b}/2, \quad i = 1, 2, \quad (3-15)$$

where $\mathbf{R}(\theta)$ is the 2D rotation matrix with respect to θ , $\bar{\mathbf{p}}_i \in \mathbb{R}^2$ is the nominal position of the i^{th} wheel relative to the robot's base, and $\mathbf{b} = [b_x \ b_z]^T$ is the vector of the rectangle dimensions.

Lastly, this planner takes into account the stability of the robot along the trajectory, enabling motion in complex and highly uneven terrain. The stability criterion used for the optimization is based on the Force-Angle Stability Measure presented in (Papadopoulos and Rey, 2000), adapted for a planar case. Considering the illustration shown in Figure 3.5, the stability measure α is given by the minimum of the two angles β_1 and β_2 .

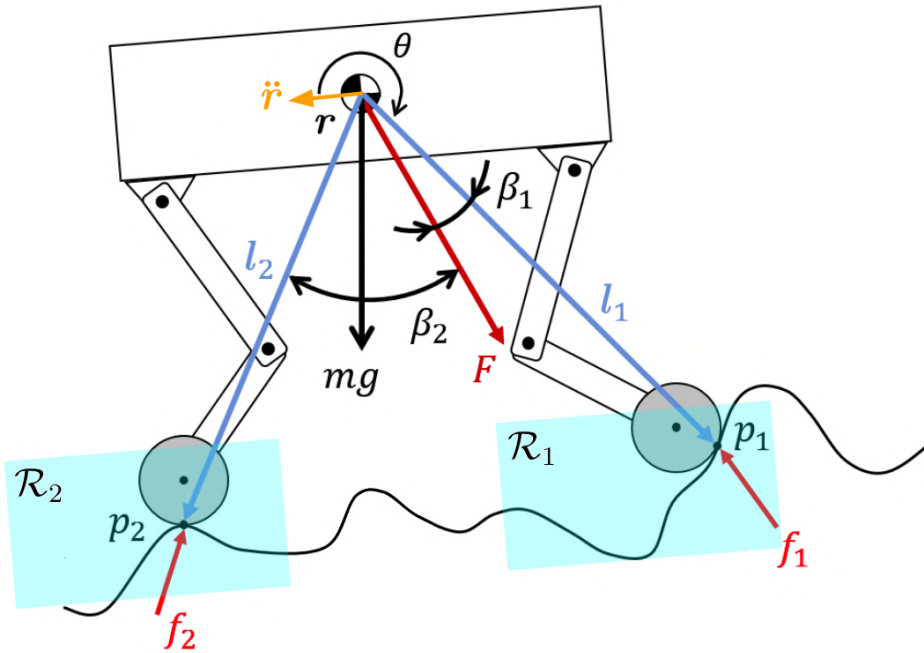


Figure 3.5: Illustration of the stability measure for the planar case.

The two tipover axis normals are represented by the vectors \mathbf{l}_1 and \mathbf{l}_2 , defined as

$$\mathbf{l}_i = \mathbf{r} - \mathbf{p}_i, \quad i = 1, 2 \quad (3-16)$$

Letting $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$ for any vector \mathbf{v} , the tipover angles β_1 and β_2 and the tipover stability measure α are computed by:

$$\beta_i = \cos^{-1}(\hat{\mathbf{F}} \cdot \hat{\mathbf{l}}_i), \quad i = 1, 2 \quad (3-17)$$

$$\alpha = \min(\beta_1, \beta_2), \quad (3-18)$$

where \mathbf{F} is the sum of all forces acting on the robot's CoM that contribute to a tipover motion instability, including gravitational loads, inertial forces¹ and external disturbances.

If the total force \mathbf{F} is coincident with either one of the vectors $\mathbf{l}_{1,2}$, α becomes zero and the robot is on the eminence of tipping over. When $\alpha < 0$, the robot is already in a tipover condition. Thus, it is desirable that the value of α remains large and positive, to maintain the robot in the stability region. The stability measure can be used either in the objective function for maximization or limited to a minimum value as a constraint of the optimization problem.

3.2.2 Objective Function

Since our problem formulation includes all the forces and positions of the wheels, the objective function for the trajectory optimization can be numerous: maximize the stability measure, to avoid tipping and loss of wheel contact with the ground; minimize the traction difference between the wheels, which can be very helpful in conditions where the friction coefficient of the terrain changes abruptly; or minimize the difference between the wheels normal forces to improve stability in uneven terrains. However, the use of a cost function can be highly costly and increase the amount of tuning parameters for the optimization, especially in this formulation, where the value of the objective function must be computed as a sum of the costs of all nodes. For that reason, it is also considered the option to formulate the optimization as a *feasibility problem*, similar to (Winkler et al., 2018). In this case, there is no cost function to minimize. The objective is simply to find the decision variables that fulfill all the constraints, which should speed up the optimization considerably. The influence of the cost functions on the generated motions is evaluated in details in the next section.

¹The fictitious force that appears to act on an object due to its inertia when the frame of reference used to describe the object's motion is accelerating compared to a non-accelerating frame.

3.3 Results

This section discusses the implementation and testing of several motions generated by the 2D motion planning framework. The presented 2D TO formulation was tested in different terrain types including a half-pipe, a terrain with a sine profile, a step 0.2 m high, among others. An initial assessment of the 2D trajectories is performed via visualization in MATLAB[®], with a script that illustrates the robot motion together with the contact forces on the wheels, the base acceleration and the stability margin. Further, the trajectories are validated in physical simulations using a real model of the ANYmal robot equipped with non-steerable torque-controlled wheels.

3.3.1 Implementation

The 2D trajectory optimization was implemented in C++² using the Ifopt (Winkler, 2018) interface for the interior-point solver Ipopt (Wächter et al., 2006). Ifopt provides an intuitive interface for constructing the optimization problem that allows the user to define independent sets of variables and constraints, and the overall problem is automatically built from these sets using an automatic index management feature. This way, each node of the trajectory is defined as a variable-set that all the constraints are imposed on, rather than defining one huge set with all the variables. The derivatives (Jacobians) for all the constraints and costs are provided analytically, which significantly reduces the computational cost of the solver. The Hessian matrix of second derivatives, on the other hand, is iteratively approximated by Ipopt through a limited-memory quasi-Newton algorithm (L-BFGS) (Wächter et al., 2006).

The time interval between the nodes ΔT was chosen as 0.1 s, which is refined enough to ensure physically feasible and dynamically consistent motions. This means that a problem with a 4.0 s time horizon has 720 optimization variables, 485 equality constraints and 813 inequality constraints. The robot parameters used for the optimization were adapted from the quadrupedal robot ANYmal with wheels and are presented in Table 3.1.

For all terrains, the position of the base is initialized by a linear interpolation between the initial and desired final position of the robot. The positions of the wheels are initialized assuming the default stance position of the robot's

²The same formulation was implemented and tested on MATLAB[®] using the free optimization toolbox OPTI (Currie and Wilson, 2012). However, its use was proved impracticable since the computational time for solving the optimization problem was around dozens of seconds per trajectory.

Table 3.1: Robot's parameters used for the 2D trajectory optimization.

Parameter	Symbol	Value	Unit
Mass	m	18.2747	kg
Inertia	I_y	0.8842	kg m ²
Motor's saturation torque	τ_{max}	32.0	Nm
Wheel's radius	r_w	0.05	m
Kinematic limit in x	b_x	0.15	m
Kinematic limit in y	b_y	0.10	m
Terrain friction coefficient	μ	0.5	-

legs with respect to the base along the entire trajectory. The velocities on each node are initialized with the average speed of the robot, given by the total displacement of the trajectory divided by the total time duration. The contact forces are initialized with the same value at all nodes: the maximum allowed traction force in the x -direction and half the weight of the robot's CoM in the z -direction.

3.3.2 Suitable Cost Functions

Several tests are conducted in flat terrain to evaluate how the use of a cost function would affect the trajectories and the computational cost of the framework. Four objectives were considered for the analysis: minimize the difference between the traction forces on the wheels, minimize the difference between the normal forces on the wheels, maximize stability, and no objective function. The respective cost functions to be minimized for each case are presented in the following equations:

$$\bar{J} = \sum_{k=0}^{n-1} (F_{T_1} - F_{T_2})_k^2, \quad (3-19)$$

$$\bar{J} = \sum_{k=0}^{n-1} (N_1 - N_2)_k^2, \quad (3-20)$$

$$\bar{J} = - \sum_{k=0}^{n-1} \alpha_k, \quad (3-21)$$

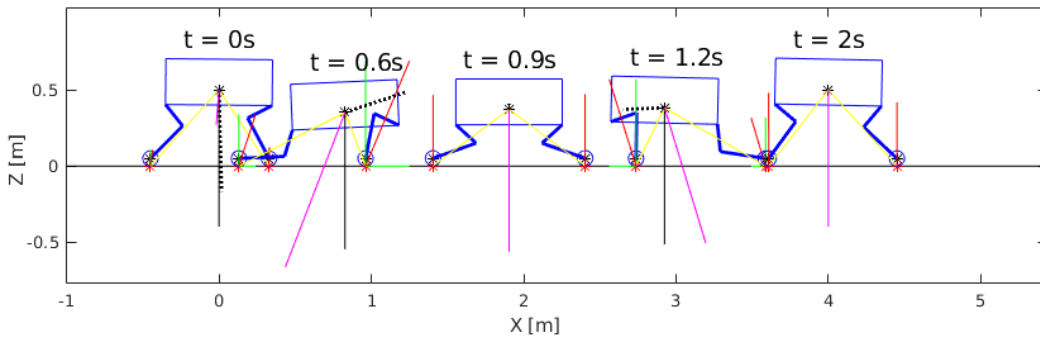
$$\bar{J} = 0, \quad (3-22)$$

Except for the case where it is explicitly maximized, the stability measure is added as an inequality constraint in the optimization problem, being constrained to remain above a minimum threshold on all nodes:

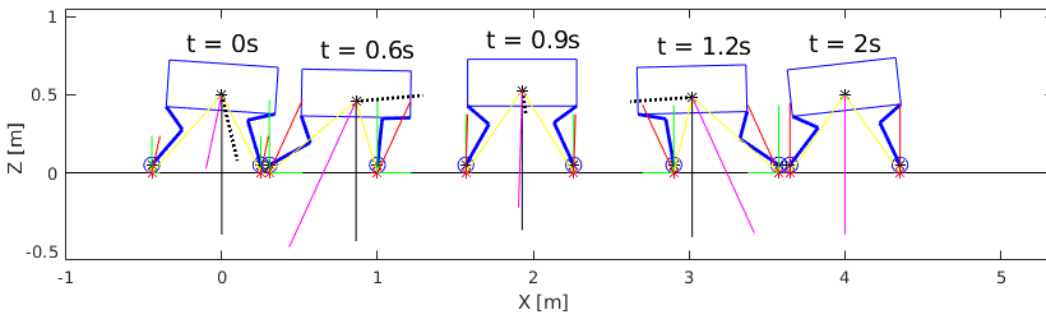
$$\alpha_k > 6.3^\circ, \quad k = 0, \dots, n - 1 \quad (3-23)$$

At this point, the threshold limit is chosen arbitrarily based on empirical testing. Further evaluation on an ideal threshold is carried out in Chapter 4.

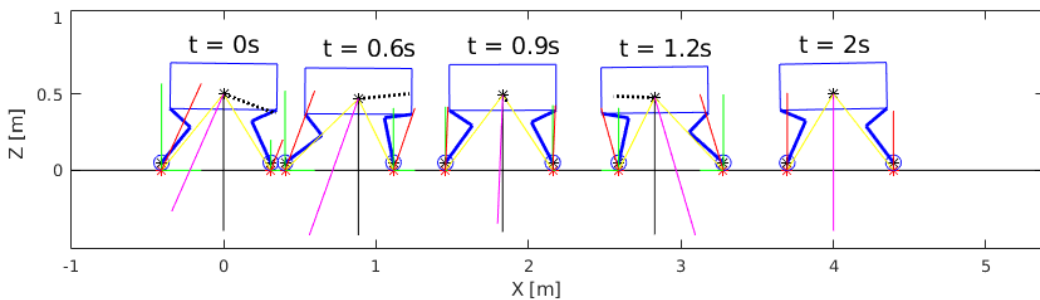
In flat terrain, tipover instability can occur at high acceleration conditions, which is why the acceleration limits for the base and the wheels are set to $\pm 10 \text{ m/s}^2$ for the following tests. Figure 3.6 shows the 2D optimized trajectories for the robot moving on a flat terrain from $x = 0.0 \text{ m}$ to $x = 4.0 \text{ m}$ in 2.0 s, starting and finishing at zero velocity.



(a) Maximize stability measure.



(b) Minimize difference between normal forces.



(c) No cost function.

Figure 3.6: 2D optimization results for the robot driving on flat terrain with different cost functions. The red lines are the contact forces, the green lines are the decomposed normal and traction forces, the back full lines are the gravity force and the black dotted lines indicate the direction of the CoM's acceleration. The pink line is the total force acting on the robot's CoM and the yellow lines represent the tipover axis normals, the limits for stability.

The effect of each cost function on the robot's motion can be visually

identified from the figure. When maximizing stability (Figure 3.6(a)), the CoM is maintained at a lower height and the wheels are positioned farther from each other to increase the area of the polygon defined by the contact points. When accelerating ($t = 0.6$ s), the base is moved forward and the torque is higher on the front wheel. Similarly, the base is moved backwards when de-accelerating ($t = 1.2$ s), and the torque is higher on the hind wheel.

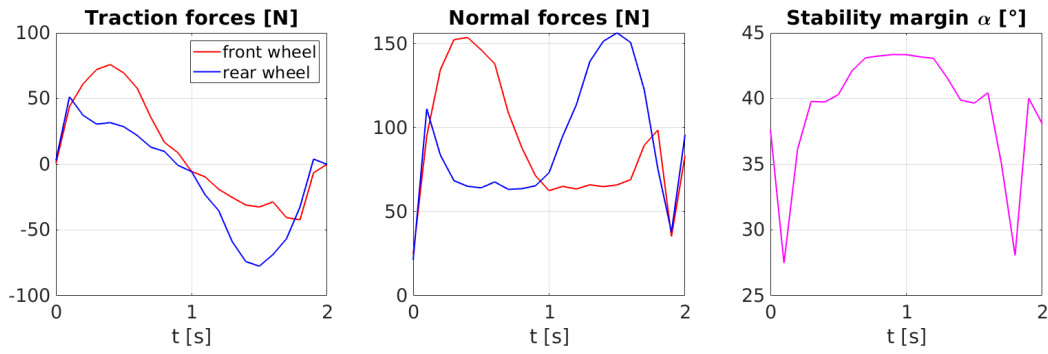
In contrast, there was no significant difference between the results for minimizing the traction forces or the normal forces on the wheels, costs presented in (3-19) and (3-20). The motion was similar for both cases and it is depicted in Figure 3.6(b). As it can be seen, the wheels' positions and the base orientation are constantly adjusted to keep a more evenly distribution of forces between the wheels. As a consequence, the stability measure also remains positively large along the trajectory.

Lastly, Figure 3.6(c) shows the trajectory generated for the feasibility problem. In this case, since there is no cost function, the solver converges to the closest feasible solution that fulfills all the constraints.

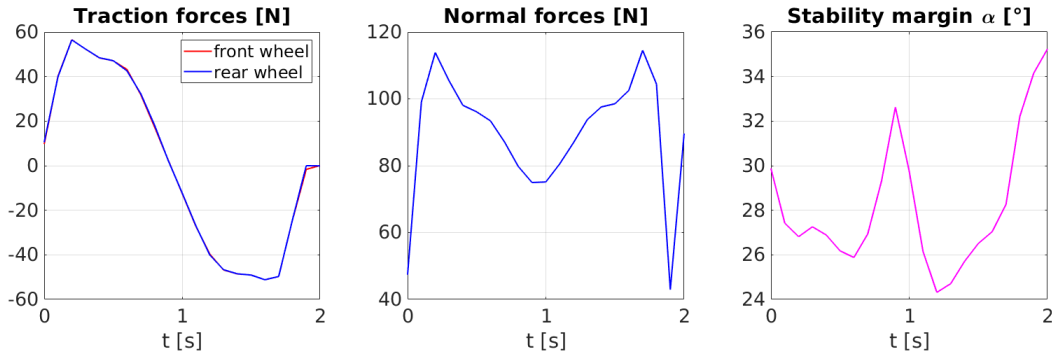
For a quantitative analysis, Figure 3.7 shows the plots of the traction forces, normal forces and stability measure for each of the motions presented in Figure 3.6. Figure 3.7(a) shows that the stability margin never reaches a value lower than 27° when maximizing the stability measure. In Figure 3.7(b), the difference between the normal forces on the front wheel and the hind wheels is so small that is imperceptible on the plot. In addition, the traction forces on both wheels are also almost the same. The solution for the optimization with no cost function (Figure 3.7(c)) presents a lower average stability margin during the motion. The traction force is higher on the hind wheel when accelerating and higher in the front wheel when braking, which contributes to the lower stability margin, but it is still within the limits.

The computational time that it takes to compute the trajectory in each case is also an important factor to be evaluated, especially considering that the flat terrain is the simplest case and the computational cost will most likely be higher for more complex terrains. All computational times stated in this thesis were obtained on a processor Intel® Core™ i7-7500U, 2.7GHz, dual-core, 64-bit. The trajectory optimization for flat terrain took 0.075 s without a cost function, 0.114 s when minimizing the difference between the normal forces, and 0.622 s for maximizing stability. Optimizing the stability measure (3-18) adds a non-linear non-convex cost function to the problem, which causes the computational cost of the framework to increase in over 8 times compared to not using an objective function.

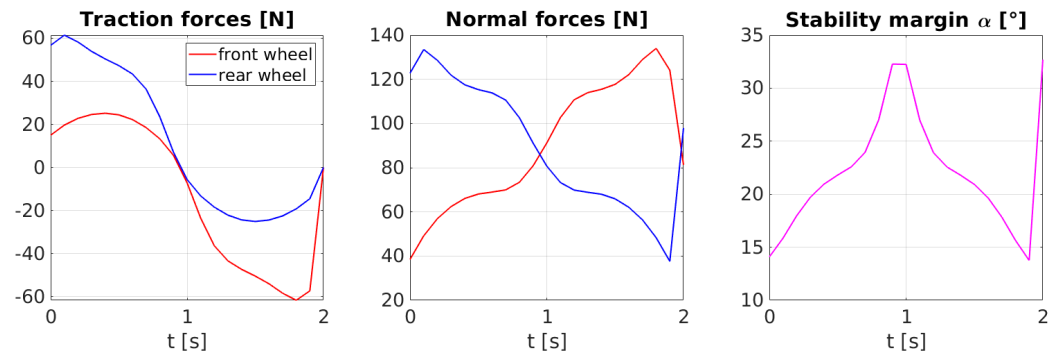
The general goal of a motion planning framework is to generate a



(a) Plots for the motion that maximizes the stability measure.



(b) Plots for the motion that minimizes the difference between normal forces.



(c) Plots for the feasibility problem solution, i.e., no cost function.

Figure 3.7: Plots for the contact forces on the wheels and the stability measure α for each motion.

physically feasible trajectory for the system, which is well accomplished by solving the feasibility problem (3-22) and it is much faster. For that reason, this is the approach used for the trajectory optimization problem in this thesis. The inclusion of the stability measure is maintained as an inequality constraint to ensure safety in all conditions.

3.3.3 Trajectories

Once the best approach for the cost function was investigated, further testing was carried out in different terrain types. The solver computation time

for the planner depends on the complexity of the terrain and the optimization parameters, but it remained in average 34.7 ms per second of trajectory. This represents a Real-Time Factor (RTF) of 30.2, which means the computational time for the solver is in average 30.2 times shorter than the planning horizon. Table 3.2 presents an illustration of the terrains used for the tests and the respective computation time for planning the trajectory, given a time horizon and the desired final position for the robot.

Figure 3.8 shows the motions for the half-pipe terrain and the slope terrain. In both cases, the terrain profile has positive and negative slopes that require the robot to adjust its behavior accordingly for traversing it. For the half-pipe terrain, there is an abrupt descend at the beginning of the motion and the robot starts adjusting the positions of the CoM and the front wheel prior to the terrain decline, which ensures a stable transition. This is only possible because the motion planner takes into account the terrain information and the dynamics of the robot, which allows for the robot to be already in a proper configuration when faced with the obstacle. In such cases, a purely reactive controller could fail in adjusting the robot's configuration in time to prevent a tipover condition, which is why most reactive approaches employ quasi-static assumptions and low speed limits for the motions.


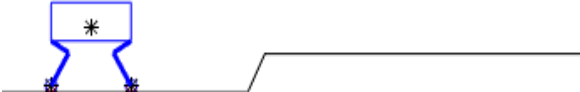
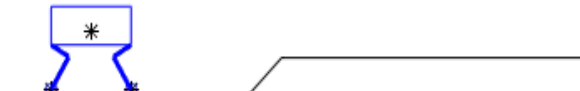



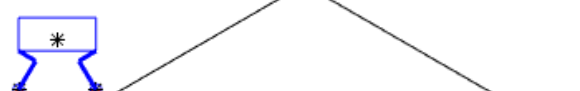

The same predictive behavior is easily perceived when the robot has to drive up steep obstacles, such as the 65° step. Figure 3.9 shows the motion generated by the 2D TO framework for driving up the steep step. Initially, the robot maintain its base slightly backwards and move the front wheels towards the step. Next, it moves the hind legs closer to the center of mass so it has more traction to pull the front wheels up. Lastly, it moves the base closer to the ground and pulls the hind legs up, completing the maneuver. Such motions cannot be generated without knowledge of the terrain.

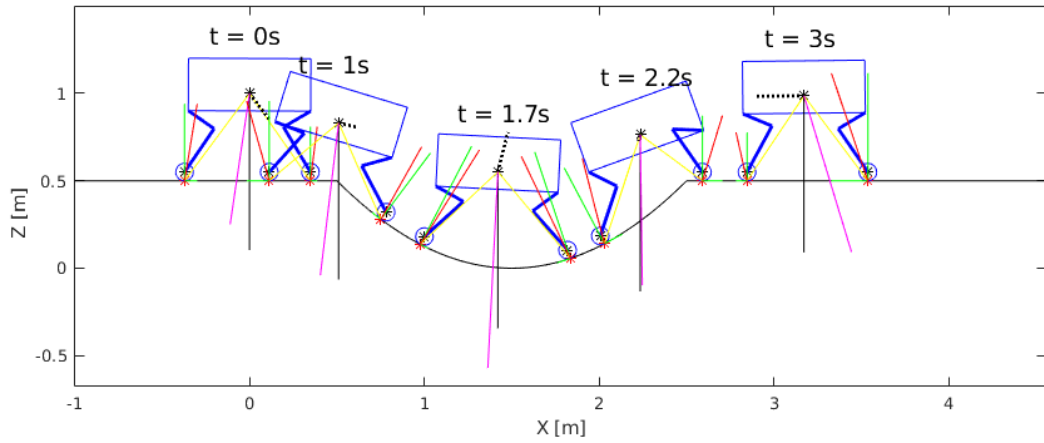
The analysis of the motions in MATLAB[®] demonstrated the dynamic consistency of the motion plans in different conditions. However, further validation is performed by verifying if the trajectories can be reliably tracked by a low-level motion controller that generates the inputs to a real wheeled-legged robot to execute the planned driving motions.

3.4 Simulations

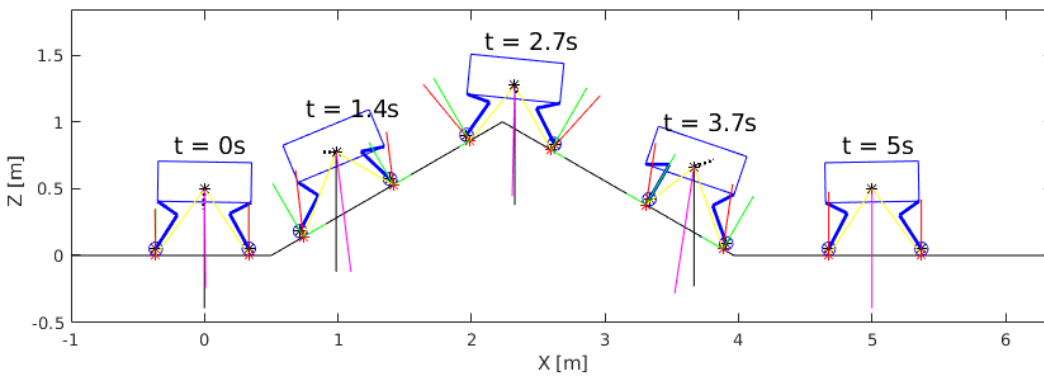
The simulations are carried out in the robot simulation environment Gazebo (Koenig and Howard, 2004) with ODE (Russell, 2008) as the physics engine, using the full rigid body dynamics of the real quadrupedal robot ANYmal, equipped with actuated non-steerable wheels. The reader is encouraged

Table 3.2: Different terrains used to test the 2D TO framework.

Terrain Description	Time horizon	Distance (in x)	Comp. time
Flat terrain 	2.0 s	4.0 m	0.056 s
Step with a 0.2 m height with a linear transition of 65° 	4.0 s	2.0 m	0.154 s
Step with a 0.2 m height with a linear transition of 45° 	3.0 s	2.0 m	0.079 s
Sine function with 5.0 rad/s frequency and 0.1 m amplitude combined with a 12° slope 	4.0 s	4.0 m	0.116 s
Sine function with 9.0 rad/s frequency and 0.06 m amplitude 	3.2 s	3.2 m	0.152 s
Sine function with 2.0 rad/s frequency and 0.2 m amplitude 	3.7 s	7.4 m	0.137 s
Increasing and then decreasing ramp with 30° slope and 2.0 m length 	5.0 s	5.0 m	0.220 s
Parabola-modeled gap with 0.5 m height and 2.0 width 	3.2 s	3.2 m	0.139 s



(a) 2D motion for the parabola-modeled half-pipe terrain.



(b) 2D motion for the increasing/decreasing slope.

Figure 3.8: 2D motions for terrains with ascending and descending trajectories.

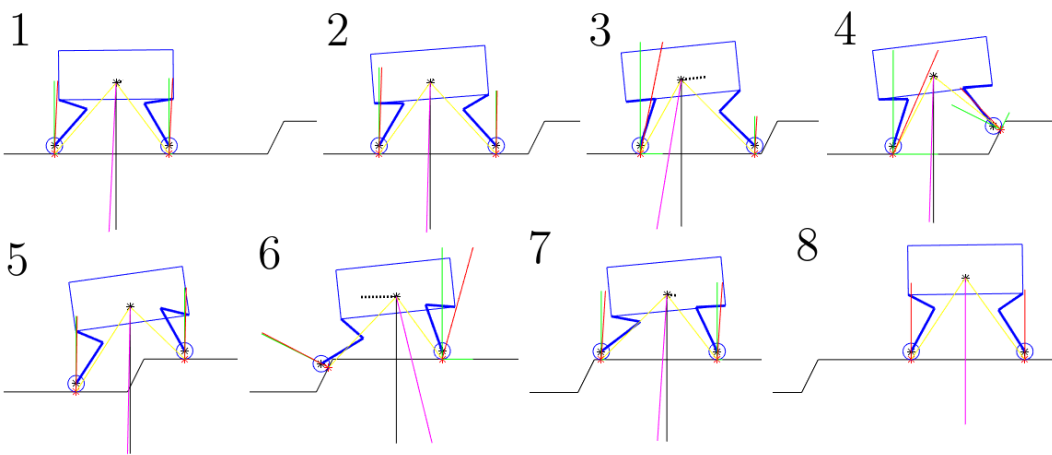


Figure 3.9: 2D motion for the robot driving up a step 0.2 m high and a 65° slope.

to watch the video available in <https://youtu.be/1ELr4stekhQ>, that shows all the motions in different terrains.

The simulation framework for ANYmal is fully ROS-compatible and it

can be used to reflect the physical behavior of a robot with high fidelity. A graphical user interface is available for monitoring and visualization of the simulated motion, with a logging module that records all data for further plotting in MATLAB®. In addition, it runs the whole-body controller, the state estimator and the mapping modules in the same way it runs on the real robot, but with inputs from simulated data.

3.4.1

From 2D to 3D motions

For simulations with the actual robot, the 2D motions are transcribed into 3D by fixing the y -coordinate of the robot's CoM, as well as the roll and yaw angles of the base, as shown in Figure 3.10. For the wheels' positions, the default stance y -position is maintained for the entire motion. The trajectory of the wheels is assumed to be mirrored between the left and right wheels, which means the optimized contact forces are divided by two between them, keeping no contact force in the lateral direction.

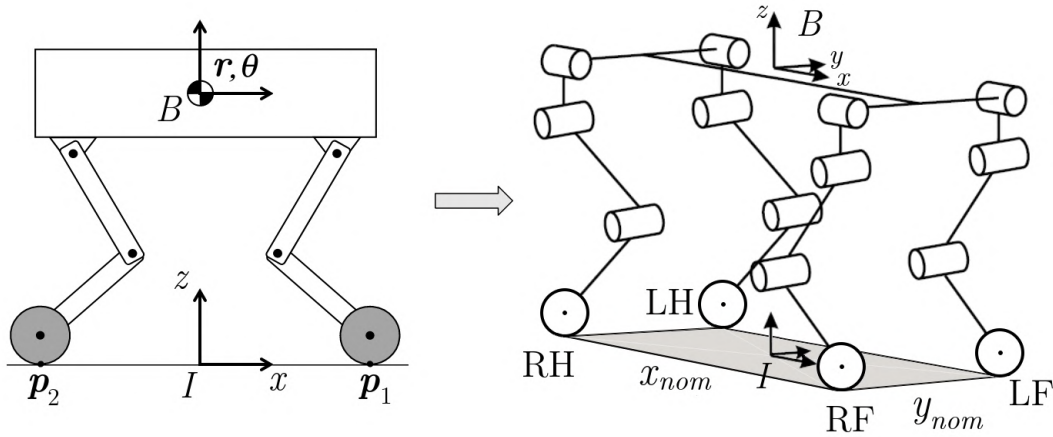


Figure 3.10: Trajectories for the planar case transcribed into 3D trajectories for simulations with the actual robot. For the wheels, the first letter indicates the left (L) or right (R) and the second, indicates front (F) or hind (H).

Thus, the 6-DoF base motion, composed by its position ${}^I\mathbf{r}_B(t) \in \mathbb{R}^3$ and orientation $\Phi_{IB}(t) \in \mathbb{R}^3$ (represented in Euler angles), is given by:

$$\begin{aligned} {}^I\mathbf{r}_B(t) &= [r^x(t) \quad r_0^y \quad r^z(t)]^T, \\ \Phi_{IB}(t) &= [0 \quad \theta(t) \quad 0]^T, \end{aligned} \quad (3-24)$$

where ${}^I\mathbf{r}_0 \in \mathbb{R}^3$ is the initial position for the base, usually given by ${}^I\mathbf{r}_0 = [0 \quad 0 \quad z_0]^T$.

The wheels' motions are defined as

$$\begin{aligned}
{}^I \mathbf{p}_{LF}(t) &= [p_1^x(t) \quad r_0^y - y_{nom}/2 \quad p_1^z(t)]^T \\
{}^I \mathbf{p}_{LH}(t) &= [p_2^x(t) \quad r_0^y - y_{nom}/2 \quad p_2^z(t)]^T \\
{}^I \mathbf{p}_{RF}(t) &= [p_1^x(t) \quad r_0^y + y_{nom}/2 \quad p_1^z(t)]^T \\
{}^I \mathbf{p}_{RH}(t) &= [p_2^x(t) \quad r_0^y + y_{nom}/2 \quad p_2^z(t)]^T
\end{aligned} \tag{3-25}$$

where y_{nom} is the nominal distance between the left and right wheels.

Lastly, since the robot's CoM does not move in the y -direction, the wheels' contact forces are equally divided between left and right wheels:

$$\begin{aligned}
{}^I \mathbf{f}_{LF}(t) &= [f_1^x(t)/2 \quad 0 \quad f_1^z(t)/2]^T \\
{}^I \mathbf{f}_{LH}(t) &= [f_2^x(t)/2 \quad 0 \quad f_2^z(t)/2]^T \\
{}^I \mathbf{f}_{RF}(t) &= [f_1^x(t)/2 \quad 0 \quad f_1^z(t)/2]^T \\
{}^I \mathbf{f}_{RH}(t) &= [f_2^x(t)/2 \quad 0 \quad f_2^z(t)/2]^T
\end{aligned} \tag{3-26}$$

3.4.2 Extended Whole-Body Controller

The 3D trajectories (base and feet motion) are provided as input to the hierarchical WBC described in (Bjelonic et al., 2019), that generates the actuation torques for the joints and the wheels while accounting for the full rigid-body dynamics of the robot, and several constraints, such as actuator limits, rolling and friction cone constraints. To allow the negotiation of steep obstacles with driving motions, we extend the existing controller with the capability to leverage terrain normals.

For tracking the desired motion for the wheels, the WBC locally models the terrain as a three-dimensional plane, which is estimated by fitting a plane through the most recent wheels' center locations, computed from forward kinematics. Then, the estimated plane is moved along its normal by the radius of the wheel to estimate the contact points. Figure 3.11 illustrates the procedure. This approach, although efficient for flat or inclined terrain, doesn't work very well with terrains that have an abrupt change in height, since the contact angle between the wheel and the terrain is misrepresented.

We extend that approach to use the knowledge of the terrain map to compute the terrain normal and contact point for each wheel separately. Since the friction cone axis is defined from the terrain normal, this mitigates incorrect estimation of the friction cones when traversing step obstacles, as shown in Figure 3.12. In this case, the lack of a terrain-aware motion controller or a contact force optimization prevents the robot from driving up the step, which is enabled by our framework.

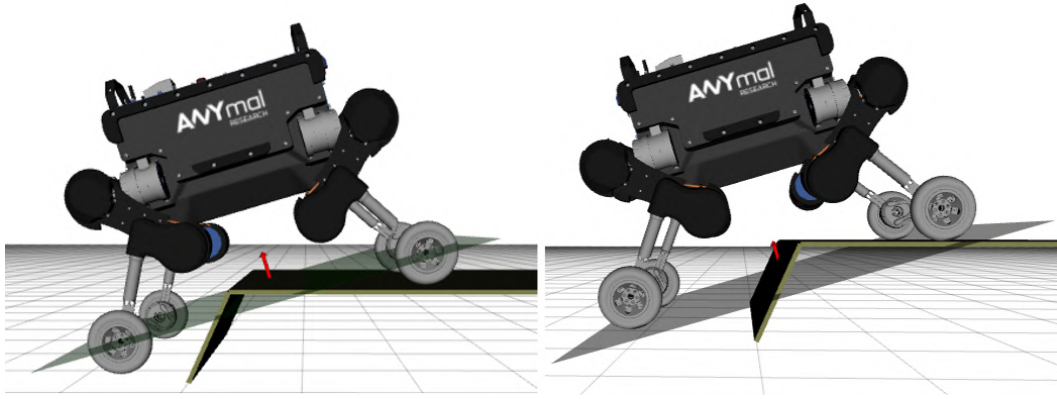


Figure 3.11: The estimation of the terrain shape by the reactive controller currently implemented on ANYmal with wheels. The terrain estimate is obtained by fitting a plane through the most recent wheel's positions and moving it by the wheel radius along the plane normal.

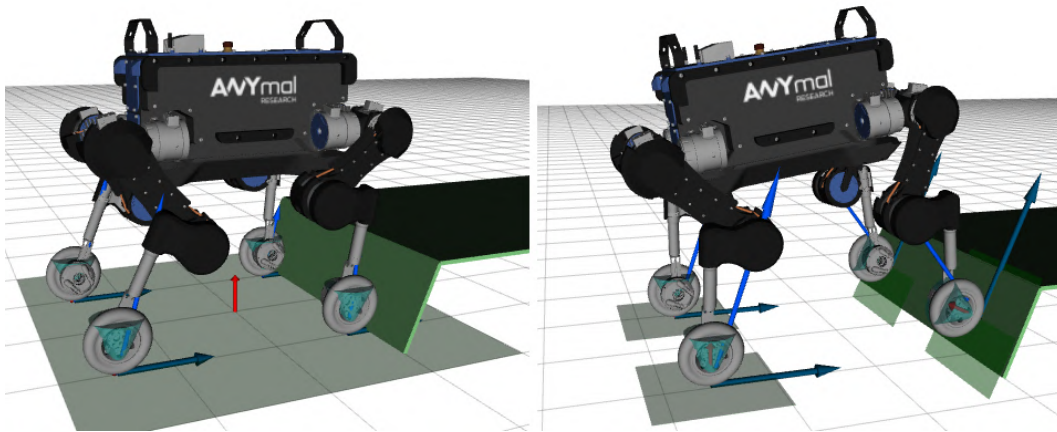


Figure 3.12: Our extended WBC uses the terrain information to estimate the wheels contact points. (a) The approach described in (Bjelonic et al., 2019) estimates the shape of the terrain by fitting a plane through the most recent wheels' positions. Note that the frictions cones are oriented with the normal plane, which is not accurate in this case. (b) Using the knowledge of the terrain normals, we are able to predict the contact points and the friction cones along the terrain map.

3.4.3 Simulation Results

Figure 3.13 shows snapshots for the robot crossing the 0.5 m deep half-pipe at an average speed of 1.0 m/s. For this terrain, the TO takes 165 ms to optimize the motion for a 3.2 s time horizon. Note that the robot maintains a kinematic feasible leg configuration by pitching the torso, which is only possible because we consider the base and wheels' motions in a single planning problem. Figure 3.14 depicts the desired motions compared with the measured positions obtained from the simulations, confirming the successful tracking of

the extended WBC. The actual accuracy of the tracking can be measured using the Root-Mean-Square-Error (RMSE), computed by

$$RMSE = \sqrt{\frac{\sum_{k=1}^n (q_{d_k} - q_{m_k})^2}{n}} \quad (3-27)$$

where \mathbf{q}_d are the desired positions and \mathbf{q}_m are the measured positions from the simulations.

The extended WBC is able to track the desired motions for the gap terrain with a RMSE of 5 mm for the base and 14 mm for the wheels.

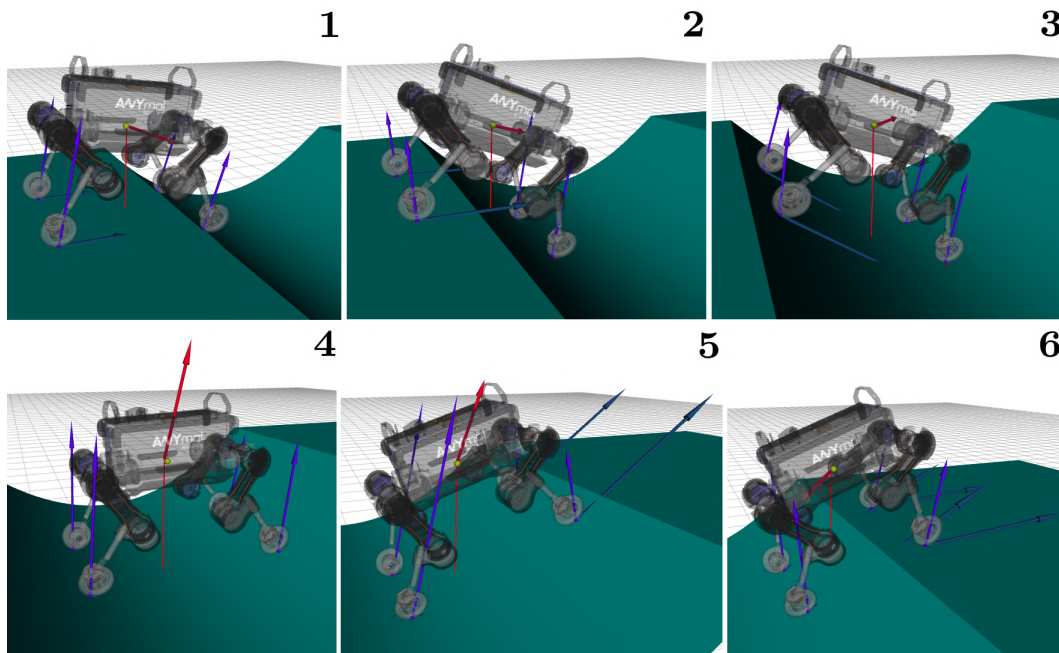


Figure 3.13: The ANYmal robot driving over a 0.5 m half-pipe at an average speed of 1.0 m/s. The light blue arrows on the wheels are the contact forces and the dark blue arrows are the wheels' linear velocity. The yellow sphere is the robot's CoM and the red arrow is base's linear acceleration.

Attempts to overcome the same terrain with a reactive controller and a desired speed of 1.0 m/s for the base nearly fails both going down and up the half-pipe, as shown in Figure 3.15, but the robot is able to reach the other side of the gap. This is because the reactive WBC runs at a high frequency and the state-of-the-art joint actuators have very fast actuation. A robot with low bandwidth actuators could not be able to perform the adjustment to the terrain in time to prevent tipover. However, the reactive controller fails completely to overcome the terrain at a average speed of 2.0 m/s or higher.

Figure 3.16 shows the robot driving over the two consecutive ramps with 30° slope at an average speed of 1.0 m/s. As expected, the robot reduces its speed when reaching the peak of the slopes so it can safely adapt to the abrupt change in the inclination. This can be clearly seen in the plots for the base's

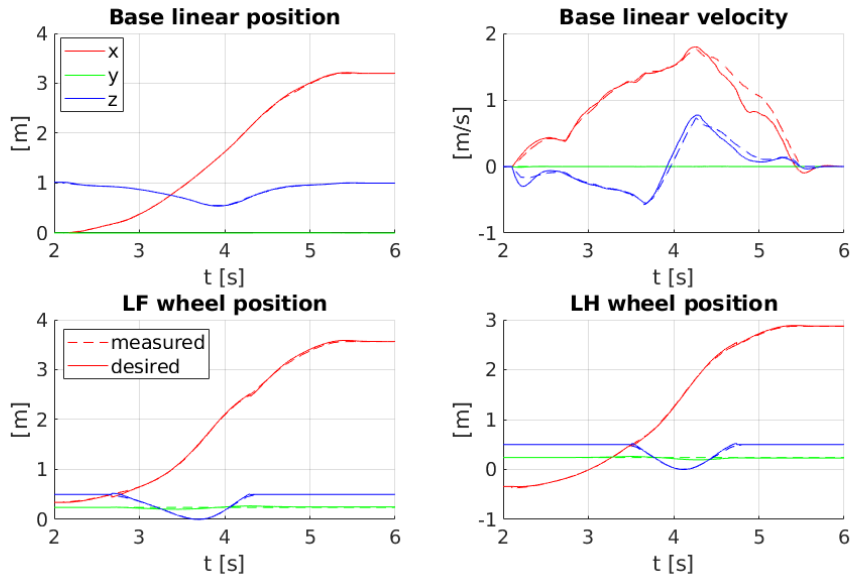


Figure 3.14: The desired positions provided as input to the extended WBC (dashed line) and the simulated measured positions of the robot's base and wheels (full line) while traversing the half-pipe terrain.

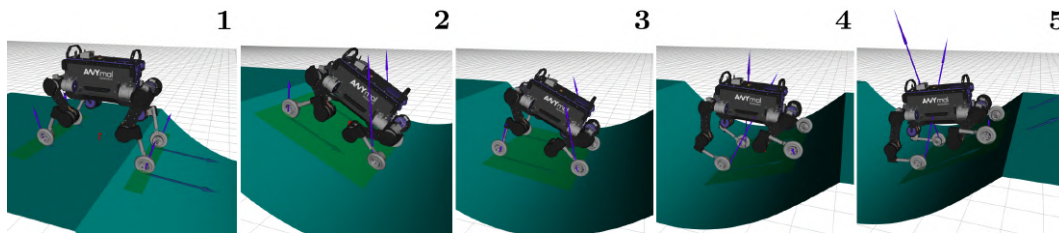


Figure 3.15: ANYmal with wheels attempting to traverse the half-pipe terrain with blind locomotion at 1.0 m/s. As soon as the controller perceives the drop in the terrain (1), it drastically lowers its CoM for stabilization (2). Immediately after the descend, there is a rising on the terrain (3), but the robot is not able to adjust the pitch angle of the base in time, causing the hind legs to reach their kinematic limits and bending in the other direction (4). The robot reaches the other side of the gap in that configuration.

desired velocity on Figure 3.17. The base speed starts to decrease seconds before reaching the top of the terrain, which happens at around $t = 7.1$ s, according to the graph for the base linear position.

The previous knowledge of the impending decreasing slope is essential for crossing such terrain at this speed. In simulations with a purely reactive controller and a desired base speed of 1.0 m/s, the robot barely reaches the top of the terrain and it is unable to overcome it because the hind legs collide with the ramp. With a speed of 2.0 m/s, the robot reaches the top of the terrain a little easier, but it is not able to adapt to the change in inclination fast enough to prevent a fall, as seen Figure 3.18.

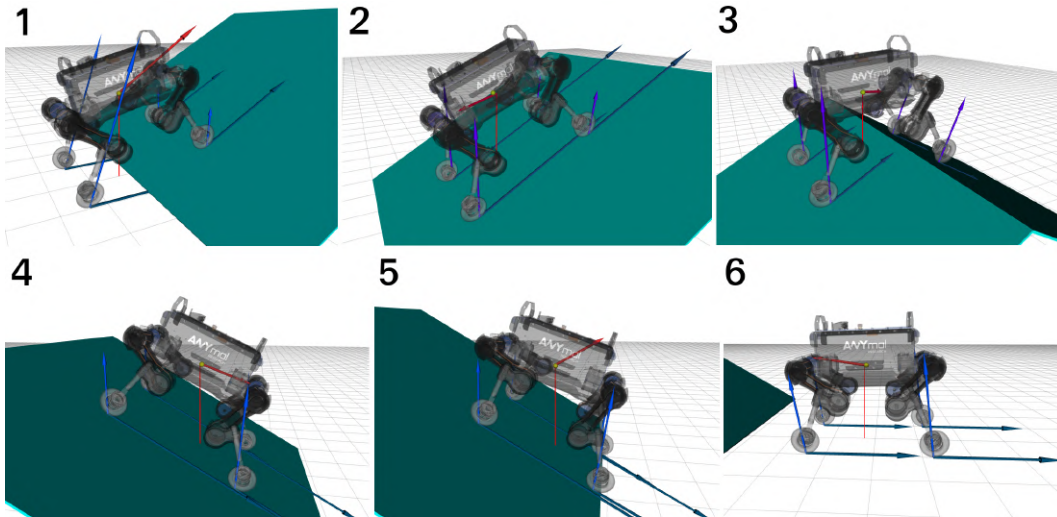


Figure 3.16: Simulation of the robot driving over two increasing and then decreasing ramps with a 30° slope.

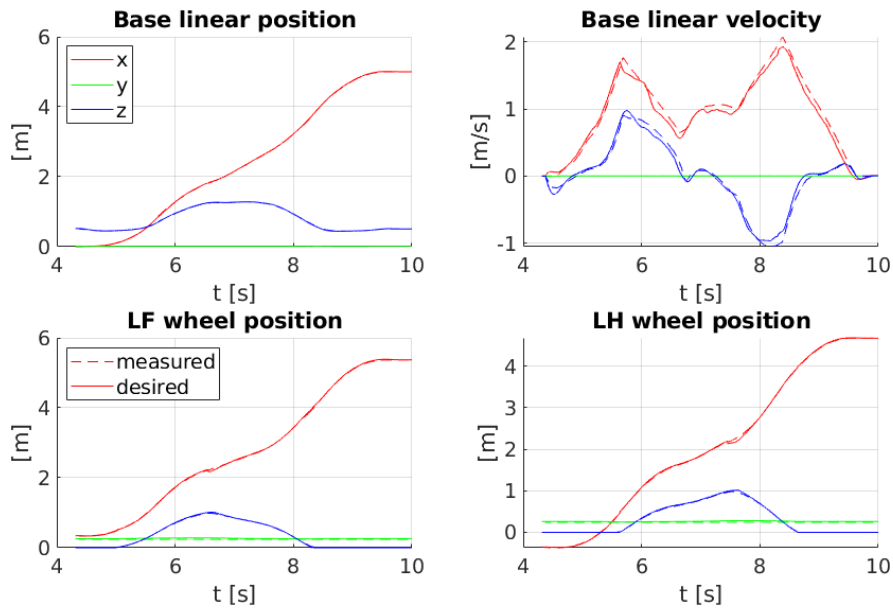


Figure 3.17: The planned trajectory for the terrain with opposite slopes and the measured data from the simulation.

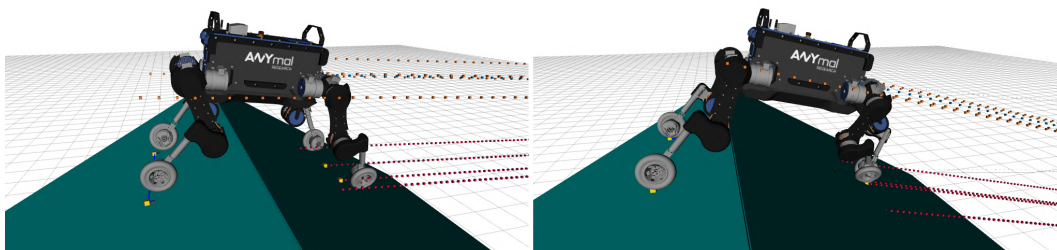


Figure 3.18: Failed attempts to overcome the slopes with blind locomotion at 1.0 m/s speed (on the left), and at 2.0 m/s speed (on the right).

Figure 3.19 shows the simulation of the robot driving over the 65° step, tracking the motion presented in Figure 3.9. A similar result has been shown in our previous work in (Medeiros et al., 2019). The height of the step is 0.2 m, which represents 40% of the legs' length. As it can be seen in Figure 3.20, the whole-body controller was able to track the desired motions for the base and the feet with high accuracy. The RMSE for the base's motion is 3 mm and for the wheels' trajectories is less than 12 mm. Moreover, the maneuver is performed in an average 0.5 m/s speed and reaches a maximum of 1.0 m/s, which is as fast as walking up motions. Attempts on crossing the 65° step with blind locomotion failed in all simulation tests with different reference velocities for the base, from 0.3 m/s to 2.0 m/s.

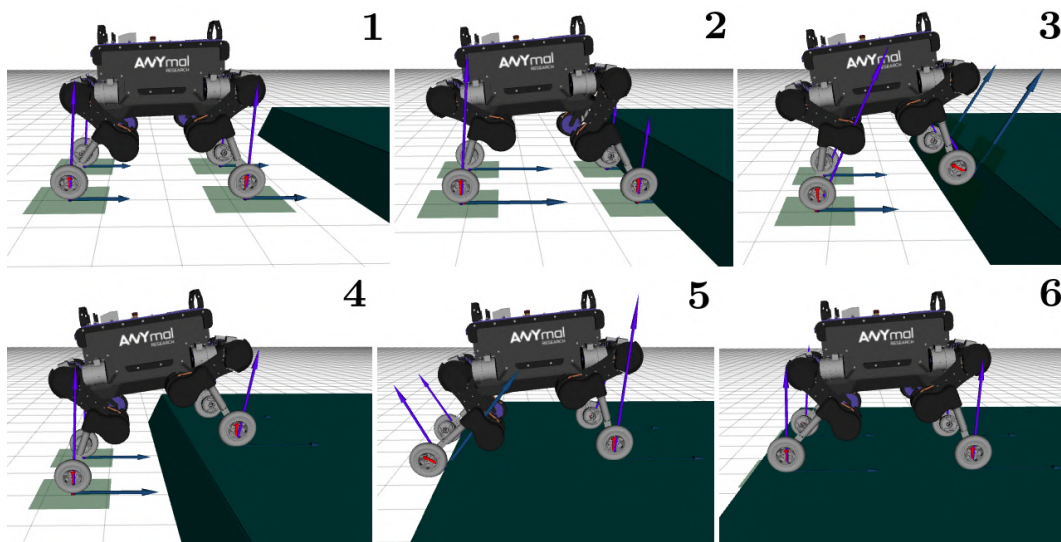


Figure 3.19: Simulation of the robot driving up a step 0.2 m high with a 65° slope.

Even for terrains that the robot is able to traverse driving blindly with the current motion control framework, the simulations show that the motions generated by the presented TO are more stable. Figure 3.21 shows the simulation results for the terrain which profile is a composition of a sine function with 5.0 rad/s frequency and a 12° slope. Note that, when moving blindly, the robot gets dangerously close to a tipping over condition, since the normal forces on the front wheels are considerable smaller than the ones on the hind wheels. The optimized trajectory, however, shows a better distribution of the normal forces, contributing for a more stable motion.

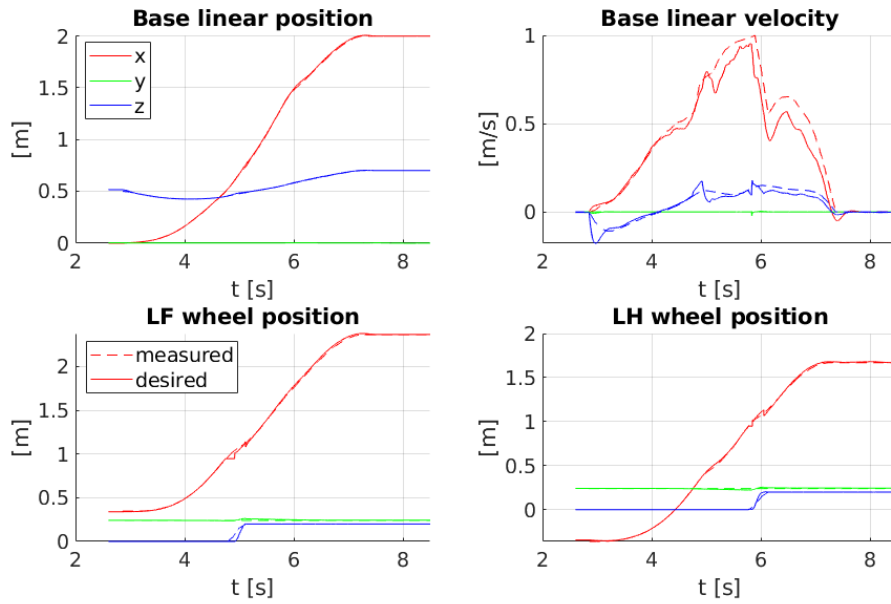


Figure 3.20: The desired and measured linear velocity for the robot's base and left wheels while driving up the 65° step.

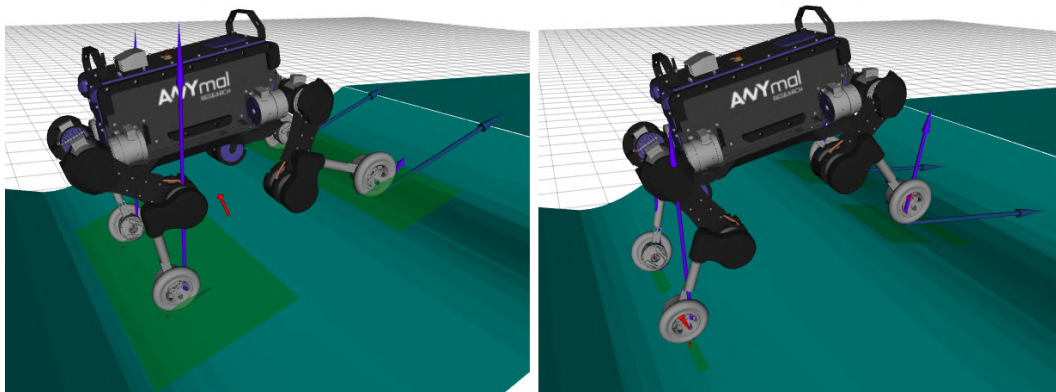


Figure 3.21: On the left is the robot moving on a wavy slope with a purely reactive controller, while on the right is the robot traversing the same terrain using the TO framework.

3.5 Conclusions

The presented 2D TO framework is able to generate optimized motions and interaction forces for wheeled-legged robots driving over challenging terrain by taking into account the robot's dynamics, the terrain profile and the stability of the robot along the trajectory. The feasibility of the motion plans is demonstrated in simulations with ANYmal in different terrain types and further comparison with a purely reactive controller.

In all the motions, the robot is moving with a speed of 0.5 m/s or higher, even in terrains with abrupt variations in height, which is enabled by the

knowledge of the terrain map.

The simulation results indicate that our approach should present good results in experimental tests, since the simulator is quite realistic. The initial assumption of mass-less legs is strong and could affect the dynamic consistency of the motions, specially considering the actuators placed on the knee joints; however, other motions using the same assumption have been successfully deployed on ANYmal (Medeiros et al., 2020; Bellicoso et al., 2018).

The simplified 2D model allows for a fast computation time, less than 35 ms/s of trajectory in average, which is suitable for an online receding-horizon implementation. This would make the approach more robust to uncertainties in the model and the terrain map.

For increasing the amount of scenarios in which the robot can navigate with the TO, a 3D trajectory optimization approach is developed in the next section with experimental tests using steps with different slopes in several configurations.

3D Trajectory Optimization for Driving Motions

The simulation for the planar wheeled-legged locomotion showed promising results that indicates that driving motions are a viable solution for negotiating challenging terrain. However, there are several limitations inherent to a 2D formulation. The mirrored trajectories between left and right wheels restricts its application to terrains with the same height in the y-axis and imposes restrictions on the orientation of the robot's base. Moreover, such a simplified model could not handle the generation of hybrid motions in which each leg has a different state depending on the desired gait. For real world applications, a complete 3D formulation may be a more suitable approach.

In addition to the increased number of variables, one of the main differences compared to the 2D formulation is the addition of a rolling constraint on the wheel's motions to ensure consistence with driving locomotion, which is not necessary for the planar case. Moreover, some motions can only be discovered with a 3D model of the robot, such as driving up a step one wheel at a time, which is also discussed here.

This chapter formalizes the TO problem for generating 3D dynamic driving motions for wheeled-legged robots and discusses its formulation as an NLP. Further, it presents the results for the simulations and experiments carried out with a real wheeled-legged robot in different terrain types to validate the motion plans.

4.1

TO Formulation for Driving Motions

This section describes in details the TO problem solved to compute the 3D driving motions, which formulation is summarized in Figure 4.1. The decision variables are robot's base linear position $\mathbf{r}(t) \in \mathbb{R}^3$ and orientation $\boldsymbol{\theta}(t) \in \mathbb{R}^3$ (roll-pitch-yaw Euler angles), the wheels' contact positions $\mathbf{p}_i(t) \in \mathbb{R}^3$ and contact forces $\mathbf{f}_i(t) \in \mathbb{R}^3$. The high-level user inputs are the initial and final state of the robot, and the total time duration T of the trajectory which is defined based on the desired average speed for the robot's base.

Figure 4.2 depicts the coordinate frames of the robot used in the formulation: I denotes the inertial frame, B denotes the base frame, and C_i

denotes the i^{th} wheel contact frame, located on the wheels contact point on the ground. For all the variables, the right superscript denotes a component of the vector and the left superscript indicates the coordinate frame, e.g. ${}^{C_i}f_i^z(t)$ represents the the z component of the contact force on the i^{th} wheel expressed in the i^{th} wheel contact frame.

$$\begin{aligned}
&\text{find } {}^I\mathbf{r}(t) \in \mathbb{R}^3 && \text{(base linear position)} \\
&{}^I\boldsymbol{\theta}(t) \in \mathbb{R}^3 && \text{(base Euler angles)} \\
&\text{for every wheel } i : \\
&{}^I\mathbf{p}_i(t) \in \mathbb{R}^3 && \text{(wheels' contact positions)} \\
&{}^I\mathbf{f}_i(t) \in \mathbb{R}^3 && \text{(wheels' contact forces)} \\
&\text{s.t. } [{}^I\mathbf{r}, {}^I\boldsymbol{\theta}](0) = [{}^I\mathbf{r}_0, {}^I\boldsymbol{\theta}_0] && \text{(initial state)} \\
&[{}^I\mathbf{r}, {}^I\boldsymbol{\theta}](T) = [{}^I\mathbf{r}_g, {}^I\boldsymbol{\theta}_g] && \text{(goal state)} \\
&\mathbf{F}_d({}^I\mathbf{r}, {}^I\boldsymbol{\theta}, {}^I\mathbf{p}_i, {}^I\mathbf{f}_i) = \mathbf{0}, \quad i = 1 : 4 && \text{(dynamic model)} \\
&s({}^I\mathbf{r}, {}^I\boldsymbol{\theta}, {}^I\mathbf{p}_i, {}^I\mathbf{f}_i) > \beta_{min}, \quad i = 1 : 4 && \text{(stability measure)} \\
&\text{for every wheel } i : \\
&{}^I\mathbf{p}_i(t) \in \mathcal{R}_i({}^I\mathbf{r}(t), {}^I\boldsymbol{\theta}(t)) && \text{(kinematic model)} \\
&{}^I p_i^z(t) = h_{terrain}({}^I\mathbf{p}_i^{x,y}(t)) && \text{(terrain height)} \\
&{}^{C_i}f_i^z(t) > 0 && \text{(normal force)} \\
&\| {}^{C_i}f_i^x(t) \| \leq f_{max} && \text{(maximum torque)} \\
&{}^I\mathbf{f}_i(t) \in \mathcal{F}(\mu, \mathbf{n}, {}^I\mathbf{p}_i^{x,y}(t)) && \text{(friction cone)} \\
&{}^{C_i}\dot{p}_i^y(t) = 0 && \text{(rolling constraint)}
\end{aligned}$$

Figure 4.1: Decision variables and constraints of the TO formulation. The constraints marked as blue are specific for optimizing driving motions over rough terrain.

The formulation of the TO problem was based on the one proposed in (Winkler et al., 2018) for legged robots with point feet. In comparison with this formulation for wheeled robots performing driving motions, the main differences are that the velocity of the end-effector's contact point is no longer forced to zero and a constraint on the rolling direction of the wheels is included to ensure consistency with wheeled locomotion. Such constraint was not necessary for the previous 2D formulation since it already optimizes over planar motions. The presented formulation allows to generate dynamic driving motions for all types of non-flat terrain, including gaps and steep slopes. Additionally, since we focus on planning driving motions, the contact between the wheels and the ground is enforced during the entire trajectory and a stability constraint ensures that the robot can safely drive over obstacles.

Note that this formulation can be extended for robots with steerable

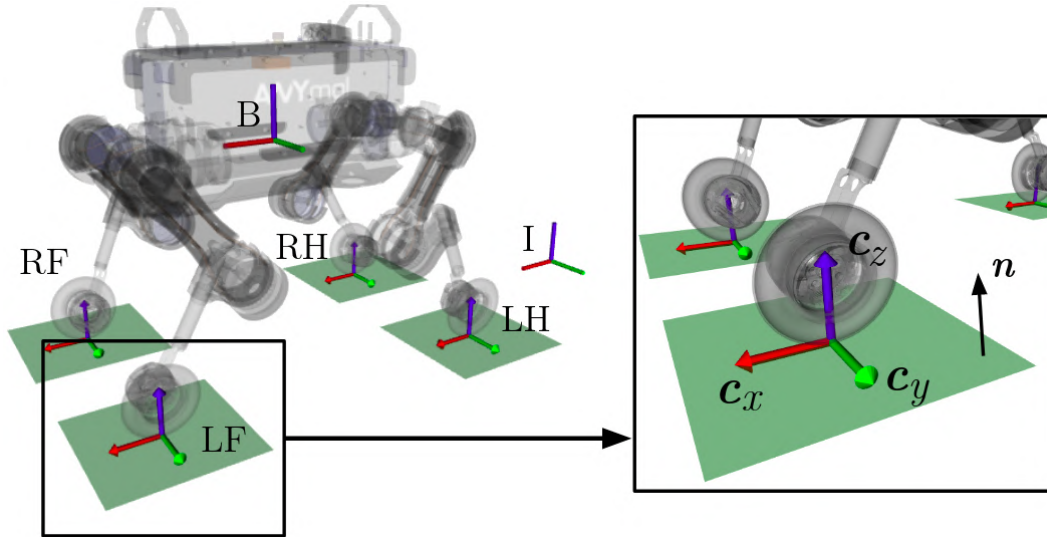


Figure 4.2: Coordinate frames used for the motion planning. The frame B is attached to the CoM of the robot and each wheel has a frame attached to their contact point with the ground. On the right, there is a detailed view of the wheel's contact frame C_i . The axis c_z is aligned with the terrain normal \mathbf{n} and the c_x axis is aligned with the rolling direction of the wheel. In the wheels' frames, the first letter indicates the left (L) or right (R) and the second, indicates front (F) or hind (H).

wheels by including the steering angle for each wheel in the set of optimization variables. This variable will impose an additional rotation along the z axis of the wheel's contact frame when computing the rolling direction of each wheel. This formulation is also generic enough to allow for motion planning for robots with different number of wheels.

4.1.1 Optimization Variables Parametrization

The OCP is transcribed into an NLP problem by optimizing over the decision variables in discrete times t_k sampled at a fixed interval ΔT along the trajectory, called *nodes*. All the optimization variables define each node at the time $t_k = k\Delta T$ of the trajectory, including the initial state, generating $n = \text{floor}(T/\Delta T) + 1$ nodes. Once all the decision variables are optimized, each dimension of the variables is represented in continuous time by connecting the nodes with third order polynomials, that can be fully defined by the value x and its derivative at the adjacent nodes and its time duration ΔT :

$$\begin{aligned} x(t) &= a_0 + a_1t + a_2t^2 + a_3t^3, \\ a_i &= f(x_k, \dot{x}_k, x_{k+1}, \dot{x}_{k+1}, \Delta T) \quad \forall k \in [0, n-1] \end{aligned} \quad (4-1)$$

where x_k is the state of the robot at the k^{th} node.

This is called the Hermite parametrization, illustrated by Figure 4.3, and allows to optimize over the state of the robot directly instead of the polynomial coefficients, which is far more intuitive for the problem formulation. This simplifies the formulation and the implementation of the NLP. The procedure to obtain the polynomial coefficients through the robot's state can be found in Appendix A.

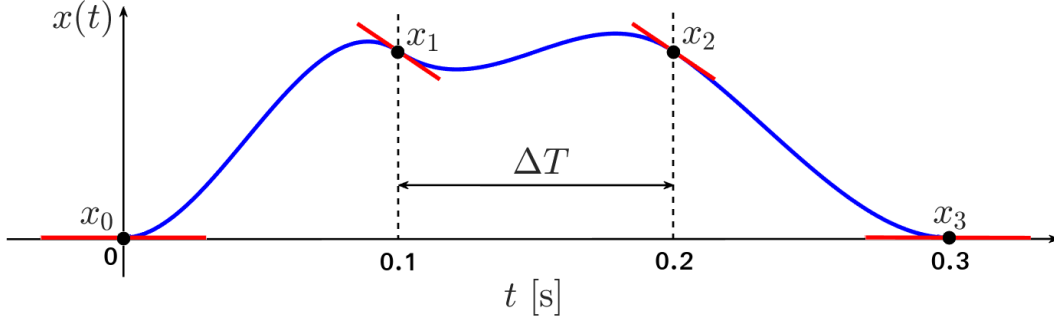


Figure 4.3: Hermite spline parametrization. Decision variables inside the optimization are the black dots (nodes) and the red lines (state derivatives at each node). The blue curves are the polynomials defined by two consecutive nodes and their derivatives.

Since both the contact forces (control input) and the base and wheels' motions (robot's state) are decision variables represented by piece-wise cubic polynomials, this approach qualifies as a *Direct Collocation* (Hargraves and Paris, 1987; Posa et al., 2014). The constraints are then enforced at a fixed time interval along the trajectory. The frequency of which the constraints are evaluated affects the computational cost of the solver.

4.1.2 Dynamic and Kinematic Constraints

The robot's dynamic model used for the TO is a Single Rigid Body model, in which the robot is approximated by a single rigid-body with mass and inertia located at the robot's CoM, as presented in Section 2.2.

Using this simplified model, the robot's CoM linear acceleration ${}^I\ddot{\mathbf{r}}(t) \in \mathbb{R}^3$ and angular acceleration ${}^I\dot{\boldsymbol{\omega}}(t) \in \mathbb{R}^3$ are given by

$$m {}^I\ddot{\mathbf{r}}(t) = \sum_{i=1}^4 {}^I\mathbf{f}_i(t) - m {}^I\mathbf{g}, \quad (4-2)$$

$${}^I\dot{\boldsymbol{\omega}}(t) + {}^I\boldsymbol{\omega}(t) \times {}^I\boldsymbol{\omega}(t) = \sum_{i=1}^4 {}^I\mathbf{f}_i(t) \times ({}^I\mathbf{r}(t) - {}^I\mathbf{p}_i(t)),$$

where ${}^I\boldsymbol{\omega}(t) \in \mathbb{R}^3$ represents the angular velocity of the robot's base in the world frame, m denotes the robot's mass, $\mathbf{g} \in \mathbb{R}^3$ the gravity vector and

$\mathbf{I} \in \mathbb{R}^{3 \times 3}$ the inertia matrix of the robot, computed around the nominal stance position. The transformation (Gehring et al., 2016) from the Euler angles $\boldsymbol{\theta}$ that define the orientation of the base (yaw, pitch, roll) and derivatives $\dot{\boldsymbol{\theta}}$ to the angular velocities in the inertial frame is given by

$$\begin{aligned} {}^I\boldsymbol{\omega} &= \mathbf{C}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \begin{bmatrix} \cos(\theta_y) \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \cos(\theta_y) \sin(\theta_z) & \cos(\theta_z) & 0 \\ -\sin(\theta_y) & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_x \\ \dot{\theta}_y \\ \dot{\theta}_z \end{bmatrix} \\ {}^I\dot{\boldsymbol{\omega}} &= \dot{\mathbf{C}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} \end{aligned} \quad (4-3)$$

As for the kinematic constraints, the joint angles cannot be directly constrained to prevent the legs from reaching its kinematic limits, since they are not included explicitly in the formulation. Instead, we constrain the wheels positions to remain within a feasible workspace that moves together with the robot's base, approximated by a parallelepiped with fixed size located on the nominal position of the wheel relative to the robot's base, as depicted in Figure 4.4. Such an approximation is common for robots with knee joints and fully articulated legs (at least 3 degrees-of-freedom per leg) (Bellicoso et al., 2018; Bjelonic et al., 2019; Winkler et al., 2018). The size of the parallelepiped must be defined by the user by taking into account the position limits of the joints. The constraint is given by

$$-\mathbf{b} \leq \mathbf{R}_{BI}(\boldsymbol{\theta}(t))({}^I\mathbf{p}_i(t) - {}^I\mathbf{r}(t)) - {}^B\mathbf{p}_{in} \leq \mathbf{b}, \quad (4-4)$$

where $\mathbf{R}_{BI} \in \mathbb{R}^{3 \times 3}$ is the rotation matrix from the inertial frame to the base frame, ${}^B\mathbf{p}_{in} \in \mathbb{R}^3$ is the nominal position of the i^{th} wheel in the base frame and $\mathbf{b} = [b_x \ b_y \ b_z]^T$ is the vector of parallelepiped dimensions.

4.1.3 Wheels' Contact Constraints

For the entire trajectory, the robot is assumed to be in driving phase, i.e., the robot is not stepping, which imposes physical constraints on the wheels' motion and contact forces. Firstly, all the wheels must be in contact with the ground, which is enforced by

$${}^I p_i^z(t) = h_{terrain}({}^I \mathbf{p}_i^{x,y}(t)) \quad (4-5)$$

where $h_{terrain}$ is the continuous 2.5D height map of the terrain (Fankhauser et al., 2018).

Assuming constant contact also implies that the normal force on the

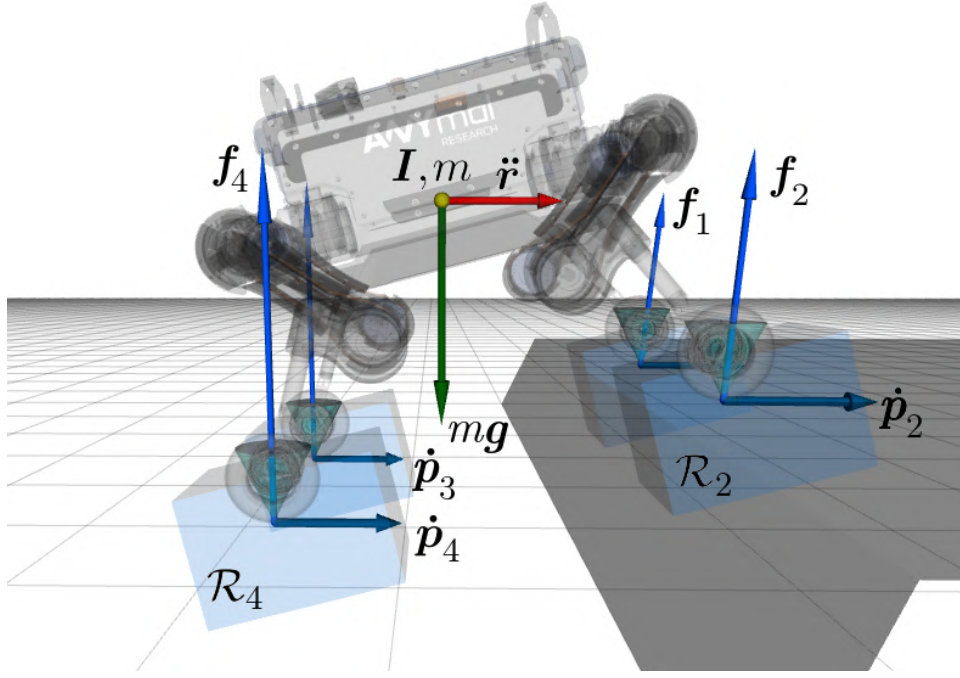


Figure 4.4: The robot model for the optimization. The joint limits of the robot are assumed not violated if the wheel's contact point \mathbf{p}_i is inside the parallelepiped \mathcal{R}_i . The contact forces \mathbf{f}_i on the wheels are constraint to remain inside the friction cone.

wheels must always be positive. Since the contact forces are explicit decision variables, the forces can be directly constrained as

$$f_{n_i}(t) = C_i f_i^z(t) \geq 0, \quad (4-6)$$

where $C_i f_i^z(t)$ is the z -component of the contact force on the i^{th} wheel expressed in the i^{th} contact frame.

To ensure no slippage of the wheels' contact points, we constrain the tangential forces to remain inside the Coulomb friction cone defined by the terrain friction coefficient μ . In our implementation, the friction cone is approximated by a friction pyramid, depicted in Figure 4.5, which makes the constraint linear and thus, speeds up the computation. The constraint is given by

$$\begin{aligned} -\mu f_{n_i}(t) &\leq C_i f_i^x(t) \leq \mu f_{n_i}(t) \\ -\mu f_{n_i}(t) &\leq C_i f_i^y(t) \leq \mu f_{n_i}(t) \end{aligned} \quad (4-7)$$

Additionally, the traction forces are limited to a saturation value correspondent to the maximum torque of the wheel's motor, which is equivalent to limit the component of the contact force aligned with the rolling direction of the wheels $C_i f_i^x(t)$ as

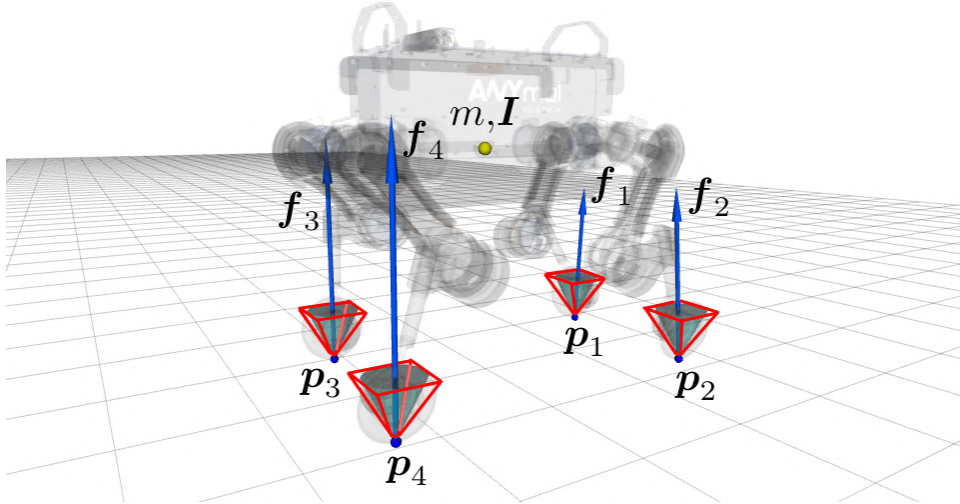


Figure 4.5: Friction pyramids (in red) used as an approximation for the friction cones.

$$-\tau_{max}/r_w \leq C_i f_i^x(t) \leq \tau_{max}/r_w, \quad (4-8)$$

Since we constrain the contact forces in a way that there is no slippage on the wheels, the maximum traction force is defined by the maximum torque on the wheel's motor $\tau_{max} \in \mathbb{R}$ divided by the wheel's radius r_w .

Unlike for the legged robots with point feet, contact points for wheeled-legged robots can have a non-zero speed or acceleration. This introduces a rolling constraint, necessary to ensure the consistency of a driving motion. We constrain the the wheels' velocity to be aligned with the rolling direction of the wheels, i.e., no lateral velocity is allowed on the wheels during the motion. This is achieved by constraining the y -velocity of the wheels' contact point in the contact frame to be zero:

$$C_i \dot{p}_i^y(t) = 0 \quad (4-9)$$

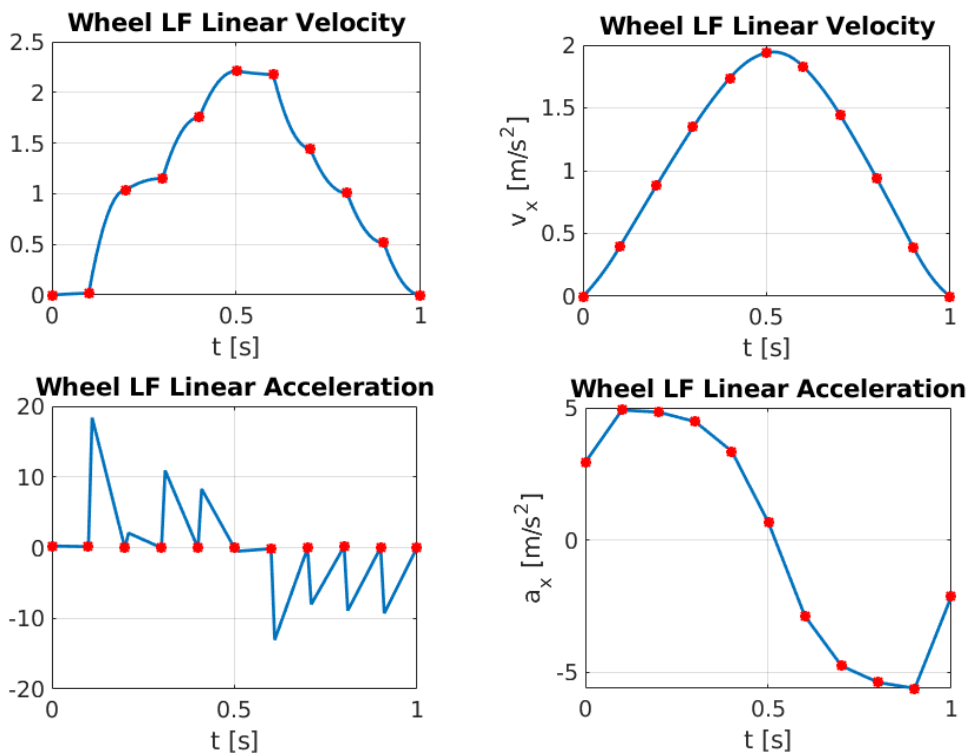
The rotation matrix $\mathbf{R}_{C_i I} \in \mathbb{R}^{3 \times 3}$ from the inertial frame to the contact frame of the i^{th} wheel is obtained by the following

$$\begin{aligned} \mathbf{e}_y &= \mathbf{n}({}^I \mathbf{p}_i^{x,y}(t)) \times (\mathbf{R}_{IB}(\boldsymbol{\theta}(t))^B \mathbf{b}_x) \\ \mathbf{e}_x &= \mathbf{e}_y \times \mathbf{n}({}^I \mathbf{p}_i^{x,y}(t)) \\ \mathbf{e}_z &= \mathbf{e}_x \times \mathbf{e}_y \\ \mathbf{R}_{C_i I} &= \begin{bmatrix} \mathbf{e}_x/|\mathbf{e}_x| & \mathbf{e}_y/|\mathbf{e}_y| & \mathbf{e}_z/|\mathbf{e}_z| \end{bmatrix} \end{aligned} \quad (4-10)$$

where $\{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$ are the respectively the $\{x, y, z\}$ components of the contact frame of the i^{th} wheel, $\mathbf{n}({}^I \mathbf{p}_i^{x,y}(t)) \in \mathbb{R}^3$ is the normal vector of the terrain at the position ${}^I \mathbf{p}_i^{x,y}(t) \in \mathbb{R}^3$, ${}^B \mathbf{b}_x$ is the x component of the basis frame expressed

in the base frame and \mathbf{R}_{IB} is the rotation matrix from the base frame to the inertial frame.

The Hermite parametrization (4-1) ensures continuously differentiable velocities and positions profiles for the base and the wheels, since the same node used as start-point for one polynomial is used as end-point for the previous one (see Figure 4.3). The accelerations, however, are not explicit decision variables and are computed from the derivative of the velocity polynomials, which allows for discontinuities in the acceleration profile. To avoid that, we constrain the wheel's accelerations to be equal between two polynomial junctions nodes, so there are no jumps in the wheels' acceleration that could cause jumps in the wheels' contact forces and consequently in the wheels' torque, which is not desired. For the same reason, the base's linear and angular accelerations are also constrained to be equal between the nodes (Winkler et al., 2018). To illustrate, Figure 4.6 shows the optimization results for the LF wheel's motion in the x -direction, with and without the aforementioned constraint, when the robot is asked to drive 1.0 m for 1.0 s in flat terrain.



(a) Wheel's motion generated **without** the acceleration continuity constraint.

(b) Wheel's motion generated **with** the acceleration continuity constraint.

Figure 4.6: Computed acceleration and velocity of the robot's LF wheel while driving 1.0 m in 1.0 s in flat terrain with and without the acceleration continuity constraint. The red dots are the times in which the decision variables are optimized over and the polynomial junctions, spaced 0.1 s apart in this case.

This wheels' acceleration continuity constraint needs to be included specifically for driving motions, since the contact points motions are continuous. In addition, to prevent abrupt motions, the acceleration in the world frame on all wheels is limited to a maximum value.

4.1.4 Stability Constraint

Since the contact forces are decision variables of our TO, it is possible to include a stability measure for the robot in the formulation. Similar to the previous planar case, the stability is based on the so-called Force-Angle Stability Measure (Papadopoulos and Rey, 2000), which 3D definition is briefly reviewed here. Consider the illustration shown in Figure 4.7, that represents a quadrupedal with the left wheels on a different level than the right.

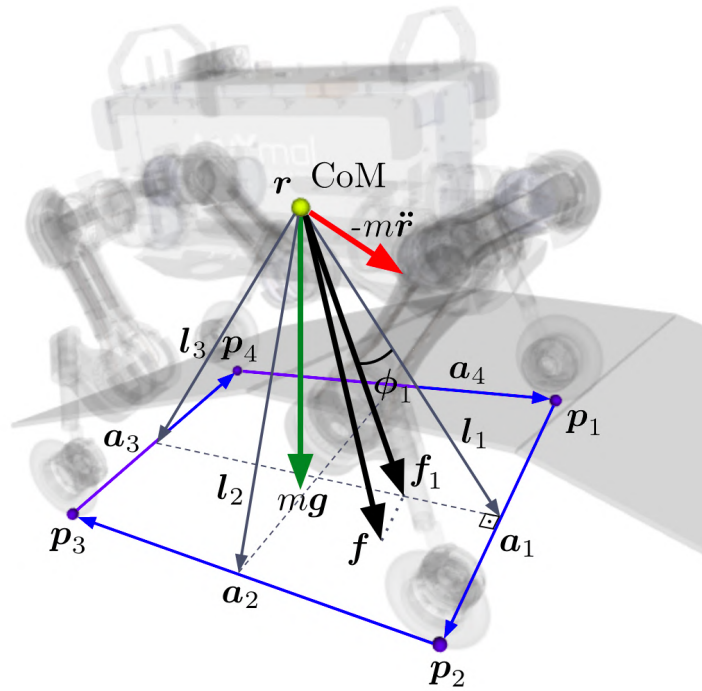


Figure 4.7: Illustration of the Force-Angle Stability Measure for a quadrupedal robot. \mathbf{f} is the sum of all forces and angular loads acting on the CoM and \mathbf{f}_1 is the component of \mathbf{f} that acts on the 1st tipover axis; ϕ_1 is the stability angle w.r.t. to the first tipover axis. Same procedure is carried out to determine the correspondent \mathbf{f}_i and ϕ_i for all tipover axes. All the vectors are represented in the inertial frame.

The wheels' contact points are represented by \mathbf{p}_i and the vectors joining the contact points are the tipover axes \mathbf{a}_i , given by

$$\begin{aligned} \mathbf{a}_i &= \mathbf{p}_{i+1} - \mathbf{p}_i, \quad i = 1, \dots, 3 \\ \mathbf{a}_4 &= \mathbf{p}_1 - \mathbf{p}_4. \end{aligned} \quad (4-11)$$

The tipover axis normals are the vectors that intersect the tipover axes and the robot's CoM, denoted by \mathbf{l}_i . They are obtained by the orthogonal projection of the vector $(\mathbf{p}_{i+1} - \mathbf{r})$ in the direction perpendicular to \mathbf{a}_i , given by

$$\mathbf{l}_i = (\mathbf{1} - \hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^T)(\mathbf{p}_{i+1} - \mathbf{r}) \quad (4-12)$$

where $\mathbf{1}$ is the 3×3 identity matrix and $\hat{\mathbf{a}}_i$ is the normalized vector $\hat{\mathbf{a}}_i = \mathbf{a}_i / |\mathbf{a}_i|$.

The force \mathbf{f} is the sum of all forces and angular loads acting on the CoM, computed by the gravitational force and the forces and moments exerted at the contact points. For each tipover axis \mathbf{a}_i , it is possible to obtain the component of \mathbf{f} that acts on the tipover axis, given by

$$\mathbf{f}_i = (\mathbf{1} - \hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^T) \mathbf{f}, \quad i = 1, \dots, 4 \quad (4-13)$$

The stability angles ϕ_i are then defined as the angle between tipover axis normal \mathbf{l}_i and force \mathbf{f}_i , i.e.,

$$\begin{aligned} \phi_i &= \sigma_i \cos^{-1}(\hat{\mathbf{f}}_i \cdot \hat{\mathbf{l}}_i), \\ \text{where } \sigma_i &= \begin{cases} +1, & (\hat{\mathbf{f}}_i \times \hat{\mathbf{l}}_i) \cdot \hat{\mathbf{a}} > \mathbf{0} \\ -1, & \text{otherwise} \end{cases} \end{aligned} \quad (4-14)$$

Lastly, the stability measure β is given by

$$\beta = \min(\phi_i), \quad i = 1, \dots, 4 \quad (4-15)$$

Critical tipover stability occurs when β approaches zero, which happens when the force \mathbf{f}_i is aligned with one of the tipover axes \mathbf{l}_i . If \mathbf{f}_i lies outside the polygon defined by the contact points, ϕ_i becomes negative and tipover is in progress. Thus, for the vehicle to remain in a stable condition, β must be positive and as large as possible. Note that this definition of the stability margin not only gives a measure of the tipover stability margin, but also identifies the currently probable tipover axis. Additionally, this measure is not limited to flat terrain and it can be applied to vehicles with any number of end-effectors, including manipulators. The stability measure can be used either in the objective function for maximization or limited to a minimum value as a constraint of the optimization problem, which is the approach used here. This way, the stability is ensured without significant increase in the computation cost of the optimization. As demonstrated with the 2D formulation in Section 3.3.2, the use of an objective function comes with

higher computational effort and the amount of tuning parameters increases. Therefore, we limit ourselves to solving feasibility problems.

4.2 Results

This section discusses the implementation and testing of several motions generated by our 3D TO framework. We verify our planning and control pipeline in physical simulation on a variety of different terrains such as steep slopes and stairs. We also perform experiments using ANYmal robot equipped with non-steerable torque-controlled wheels for steps with 45° and 65° slopes. All the motions are showed in details in the video available at <https://youtu.be/D1JGFhGS3HM>.

4.2.1 Implementation

The TO framework is implemented in C++ using the Ifopt (Winkler, 2018) interface for the interior-point method solver Ipopt (Wächter et al., 2006). The implementation is based on the software TOWR presented in (Winkler et al., 2018) for legged robots. The simulations are carried out in the robot simulation environment Gazebo (Koenig and Howard, 2004) with ODE (Russell, 2008) as the physics engine, using the full rigid body dynamics of the real quadrupedal robot ANYmal, equipped with actuated non-steerable wheels. The base and wheel motions are provided as input to the an extended version of the WBC described in (Bjelonic et al., 2019), updated to use a map of the terrain normals to compute the exact contact points and terrain planes at each wheel. For details, see Section 3.4.2.

The presented formulation requires a continuous 2.5D height map of the terrain. The height map can be either manually specified if the objects in the environment are known or generated from on-board sensors. Grid map representations such as *Octomap* (Hornung et al., 2013) or a *Gridmap* (Fankhauser et al., 2016) (shown in Figure 1.20) can be adapted to comply with the planner interface. Since our tests were carried out in known scenarios, the terrain map was analytically defined for each of the terrains.

For both simulations and experimental tests, the motion planner first reads the current state of the robot and then computes the trajectories for the desired time horizon. Figure 4.8 gives an overview of the motion planning framework. The extended WBC tracks the reference trajectories, along with the robot state estimator, in a 400Hz loop. The state estimator module (Bloesch et al., 2012) fuses kinematic measurements from each actuator with the data

from an IMU to compute the state estimation of the robot. Similarly to the extended WBC, the map of terrain normals was included in the state estimation algorithm so the wheels' contact points and velocities are computed in the right direction, which improves the robot's state estimation and limits the effects of drifts in the estimation on the trajectory tracking.

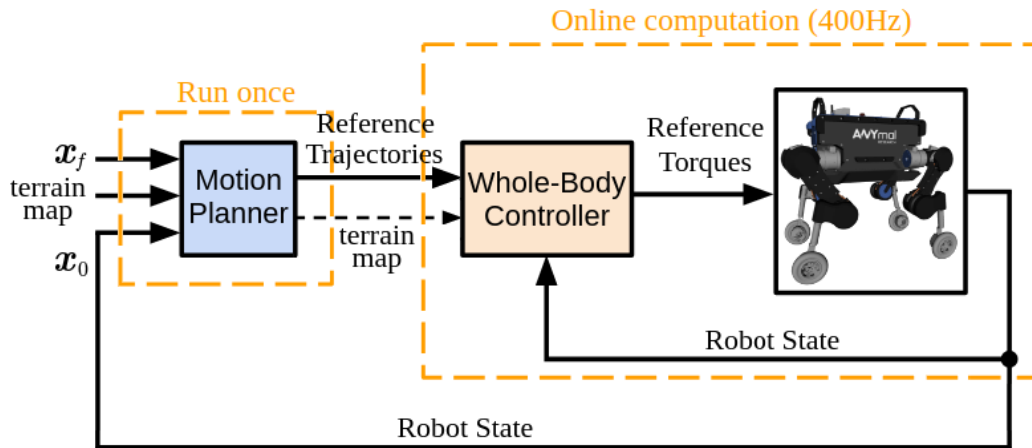


Figure 4.8: Overview of the motion planning framework. The motion planner computes the reference trajectories for a specified time horizon that is given as input to the WBC, that computes the reference torques for the robot.

For the motion planning, the kinematic constraints are enforced at a time interval of 0.08 s while all the other constraints are enforced every 0.1 s. Consequently, a trajectory with a 4.0 s time horizon has 2269 optimization variables, 1290 equality constraints and 2262 inequality constraints which is roughly three times larger than the 2D formulation. All the derivatives of the constraints are provided analytically to the solver, which drastically improves the solver's performance. The time interval in which the constraints are enforced could be decreased to improve dynamic consistency along the trajectory, but since we use a simplified dynamics model, this would result in more computation time and not necessarily generate more efficient trajectories.

4.2.2 Initial Guess

Interior point methods for non-linear programming problems are prone to local minima and can be sensitive to the initial choice of variables (Gertz et al., 2004), which is why providing a reasonable initial guess causes a significant reduction in the number of iterations necessary for convergence. For that reason, the velocities on each node are initialized with the average speed of the robot, given by the total displacement of the trajectory divided by the total time duration. The position of the base is initialized by a linear interpolation

between the initial and desired final position and the positions of the wheels are initialized assuming the default stance position of the robot's legs with respect to the base along the entire trajectory.

4.2.3 Simulations

The simulations were carried out in the robot simulation environment Gazebo with different terrain types. We verify the effect of the tipover stability criterion introduced in Section 4.1.4 as well as the ability to traverse various challenging terrain.

The effect of the stability constraint on the robot's motion is shown in Figure 4.9, where snapshots of the robot's driving on a slope with 30° inclination at an average speed of 1.0 m/s with different stability thresholds. As expected, as the stability angle threshold increases, the trajectory presents a higher pitch angle for the base and the wheels position are adjusted to minimize the difference between the normal forces on the front and hind wheels. As the threshold increases, the size of the feasible solution set is reduced and the computation cost for the optimization increases proportionally. As a compromise, the minimum stability angle was constrained to 10° for all trajectories.

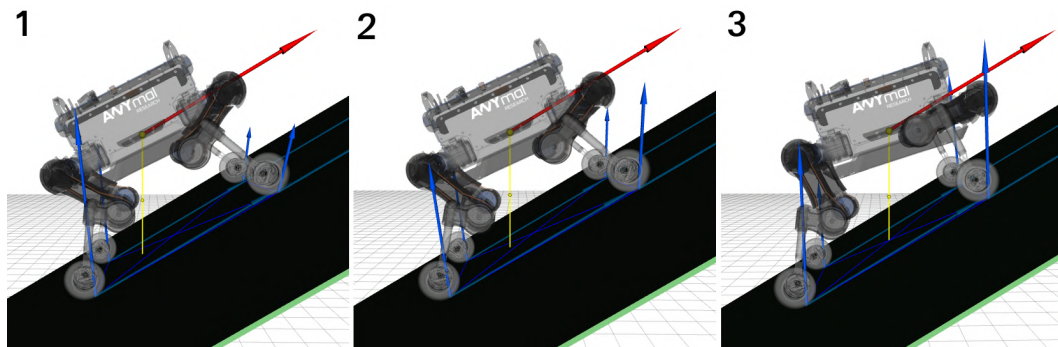


Figure 4.9: ANYmal drives on a steep incline with a 30° inclination at average speed of 1.0 m/s. The dark blue arrows on the wheels are the wheels' speed, the light blue arrows are the contact forces, the red arrow on the base is its linear velocity and yellow line is the CoM position vertically projected on the terrain. Results for: (1) $\beta \geq 0^\circ$; (2) $\beta \geq 10^\circ$; (3) $\beta \geq 20^\circ$

Figures 4.10 and 4.11 show snapshots for two challenging driving motions: crossing a 0.5 m deep half-pipe at an average speed of 1.2 m/s and traversing two consecutive increasing and decreasing slopes at an average speed of 1.5 m/s. For the gap, the motion is very similar to the one optimized by the 2D TO formulation, where the robot adjusts the pitch angle of the base and the wheels' positions to overcome the inclination of the terrain. For the slopes,

since they are placed either on the right or left side of the path, the robot tilts its base in the roll direction in order to drive over the obstacles, which is only possible because we consider the whole body planning problem in our approach. Due to the height difference in the y -direction, this motion would not be possible to generate using the simplified 2D model presented in Chapter 3. Figure 4.12 illustrates the desired motion (base and wheel) provided as input to the extended WBC along with the measured positions obtained from the simulation, and the stability angles (4-15) throughout the trajectory, which are maintained always within the stability margin of 10° . The average RMSE for the motions' tracking was 3.5 mm for the base and 9.2 mm for the wheels.

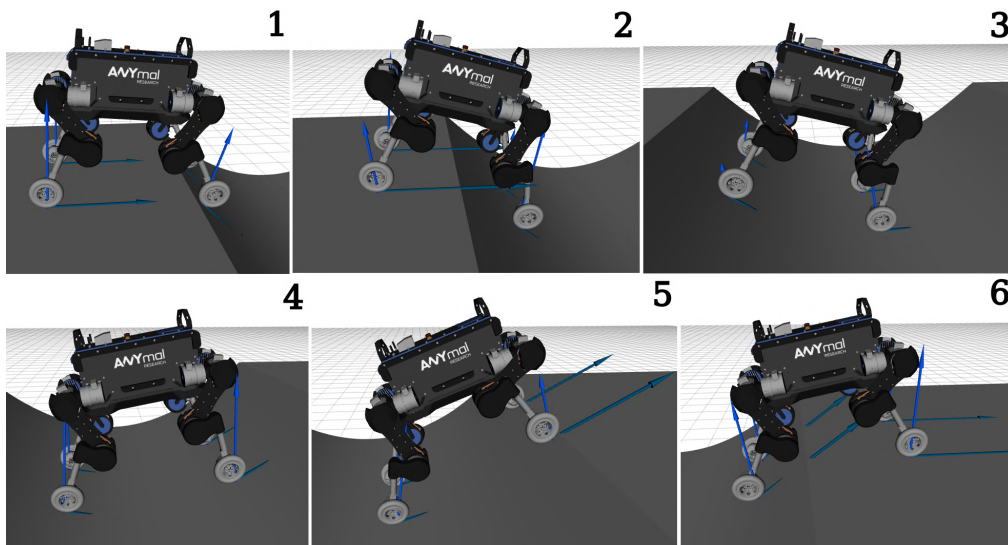


Figure 4.10: The ANYmal robot driving over a 0.5 m half-pipe at an average speed of 1.2 m/s.

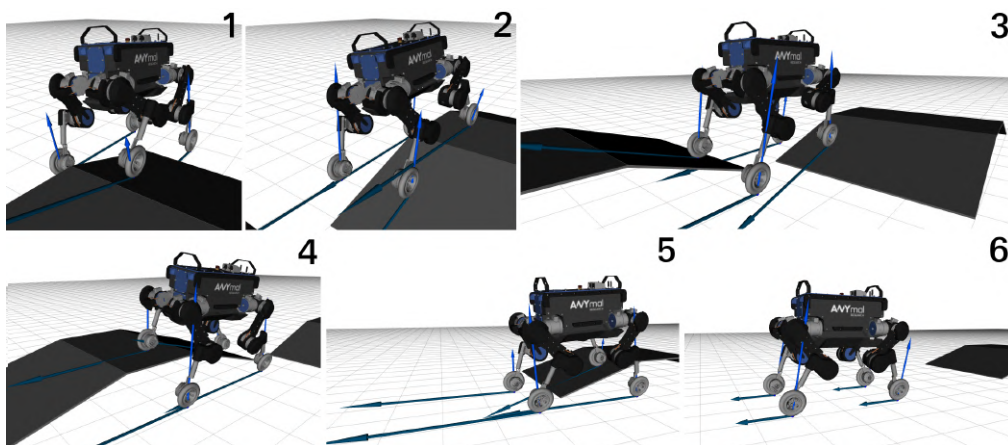


Figure 4.11: The ANYmal robot driving over two consecutive slopes with a height of 0.2 m at an average speed of 1.5 m/s.

Another motion that could only be generated by a 3D approach is presented in Figure 4.13. Two long obstacles with 0.1 m width and 0.15 m

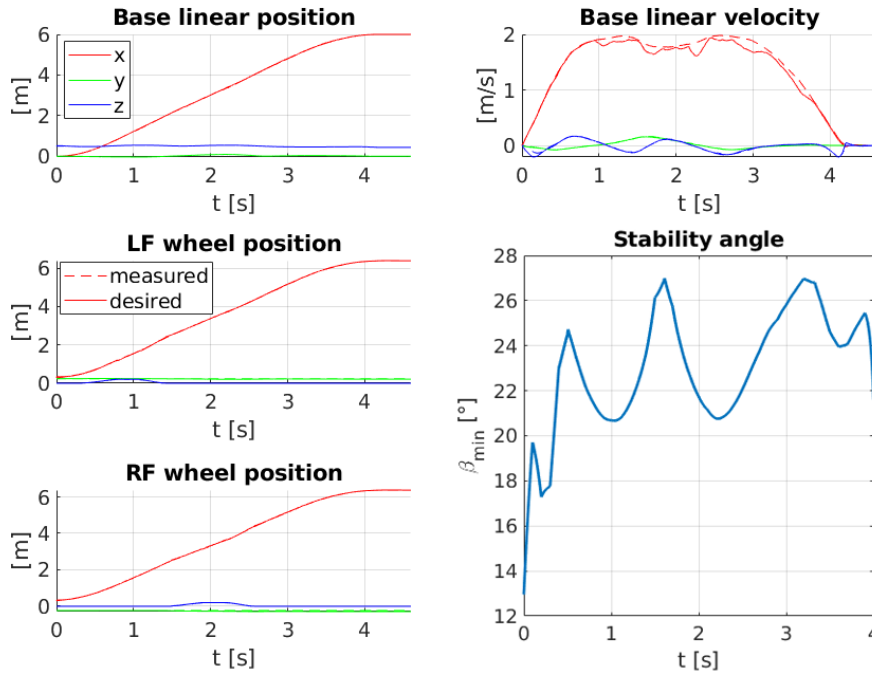


Figure 4.12: On the top, the linear position and velocity of the robot’s base and, on the left, the right and left front wheel positions. The graphs show the desired motions provided as input to the extended WBC and the measured positions from the simulations of the two slopes maneuver. On the bottom right, the variation of the stability angle along the trajectory. The robot reaches a maximum speed of nearly 2.0 m/s.

height are placed on each side of the terrain. The slope on the obstacles is 65° . For crossing this terrain, the robot must drive the first obstacle with the left wheels and then drive over the second obstacle with the right wheels. The obstacles are placed at a 1.0 m distance from each other, but there is a quick instant (Frame 4 on Figure 4.13) where both the LH and the RF wheels are on the obstacles, relying on the traction forces on the obstacle’s walls to maintain stability.

One of the most important results of this work is the robot’s gained ability to drive up steps. The proposed approach can handle multiple obstacles in succession such as the stairs shown in Figure 4.14. The stairs are composed by steps with 0.2 m in height, 0.4 m distance between them and transition with a 65° slope. Note that, as soon as the front wheels reach the second step, the hind wheels are pulled up the first step to stay within the kinematic limits of the legs. A set of five steps is completed with an average speed of 0.5 m/s, which is faster than a legged robot with point feet could do. We achieve such a speed only because we compute plans for the robot before starting the maneuver which gives the controller more information for tracking compared to just reactive locomotion with flat terrain assumption. Trying to go up the steps

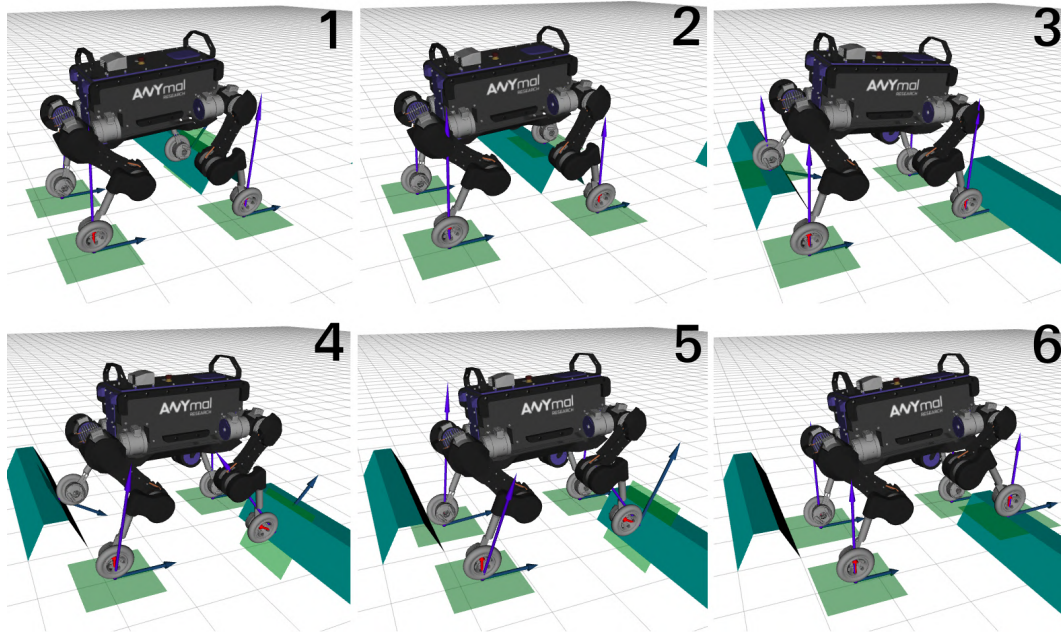


Figure 4.13: The ANYmal robot driving over two consecutive obstacles on different sides of the path at an average speed of 0.5 m/s.

with a reactive controller (Bjelonic et al., 2019) has shown no success, even in simulation.

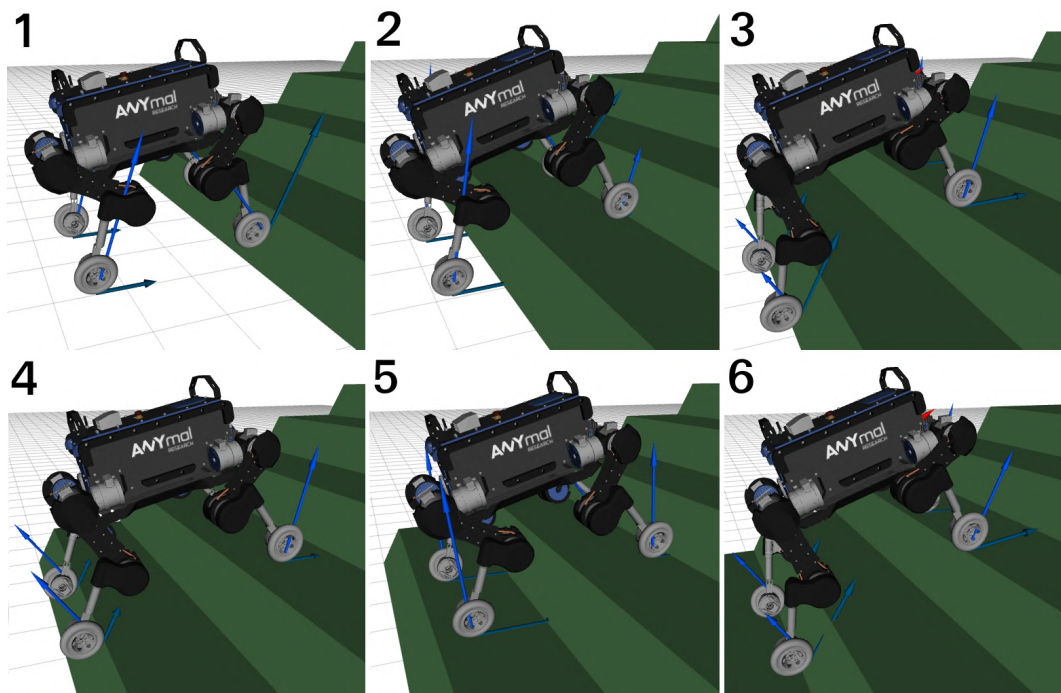


Figure 4.14: ANYmal drives over stairs composed by steps with 0.2 m in height and a 0.4 m distance between them.

Table 4.1 presents the computation times for planning trajectories in different challenging terrains. The solver computation time depends on the terrain complexity, but remained in average 2.1 times shorter than the plan-

ning horizon. All the derivatives of the constraints are provided analytically to the solver, which improves the solver’s performance.

Table 4.1: Computation times for some of the terrains used to test the 3D TO framework.

Terrain description	Time horizon	Distance (in x)	Computation time
Flat terrain	3.0 s	8.0 m	1.28 s
Step with a 0.2 m height with a linear transition of 65°	4.0 s	1.8 m	1.76 s
Step with a 0.2 m height with a linear transition of 45°	4.0 s	1.8 m	1.56 s
Sine function with 5.0rad/s frequency and 0.1 m amplitude combined with a 12° slope	3.5 s	4.0 m	1.41 s
Parabola-modeled gap with 0.5 m height and 2.0 width	3.0 s	3.5 m	1.56 s
Two slopes with 0.2 m height and 17° slope on each side of the terrain	4.0 s	6.0 m	1.87 s
Two obstacles with 0.1 m width and 0.15 m height on each side of the terrain	6.0 s	3.0 m	2.58 s
Five consecutive steps with 0.2 m in height, 0.4 m width and a linear transition of 65°	7.2 s	3.6 m	8.61 s

An important contribution of the motion planner is that we consider the whole-body planning problem (6-DoF base motion, wheels’ motion and contact forces) in a single trajectory optimization framework, while accounting for the terrain information and the dynamics of the robot. This combination allows for generating trajectories that warrant a predictive behavior to the robot. For instance, the robot starts adjusting the base position (height and orientation) prior to the obstacle so it can overcome it in a dynamic manner at a higher speed. Such motions cannot be generated with a purely reactive approach. In addition, we include the terrain map information in the tracking controller, allowing it to accurately track the desired driving motions.

Generating 3D base and wheels motions in a single planning problem using terrain information and without restrictions in speed or pose offer the advantage of an increased range of achievable motions for more complex terrains as a trade-off for the increased size of the optimization and its computational cost. Relaxation of some constraints and a decrease in the number of variables would reduce the cost for the optimization, but could decrease its applicability.

4.2.4

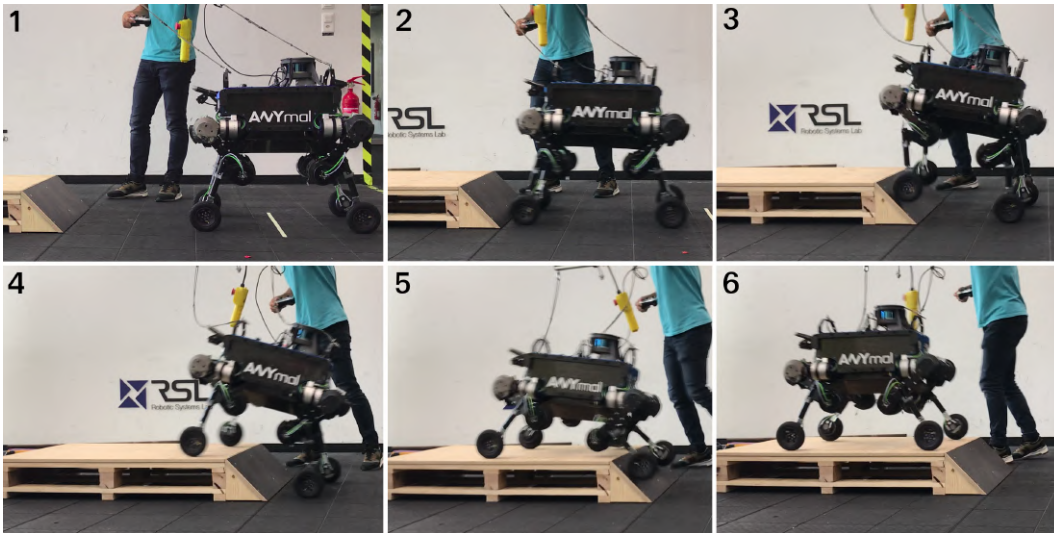
Experimental Results

We conducted experimental tests in four different configurations: two steps 0.2 m high (40% of the legs length) with a 45° and a 65° incline either in the center of the path or only on the right side of the robot. The optimization parameters were the same in all tests, with the exception of the robot's goal pose. All the obstacles are overcome with an average speed of 0.5 m/s.

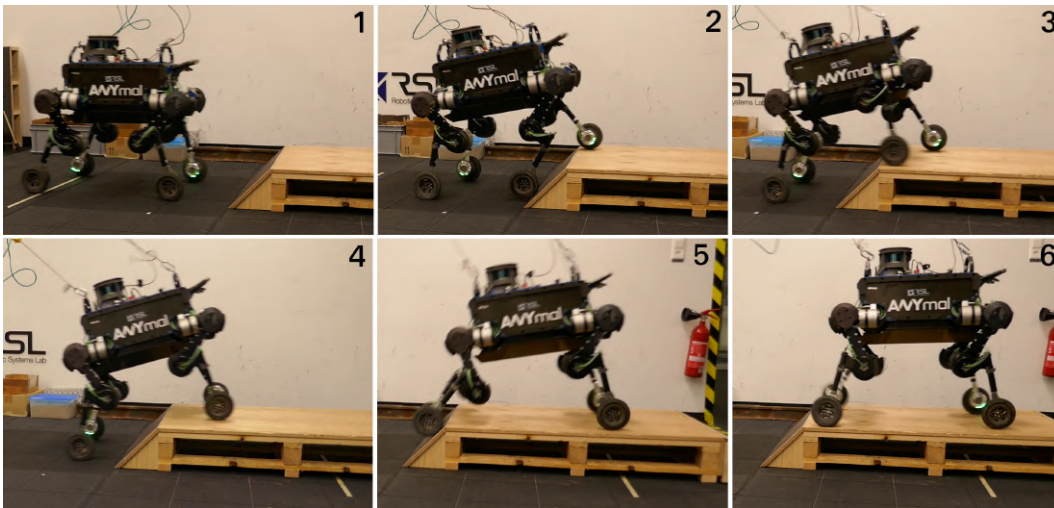
Figure 4.15 shows the robot driving over the step with a 45° slope in two different ways: both wheels at the same time and one wheel at a time. In the first case, the base of the robot is kept in a lower position and with a higher distance between the front and the hind wheels to improve stability. In the second, the robot drives up the left wheel first and, once both front wheels are on the platform, the hind wheels are driven up together to complete the motion. In this case, the transition over the step is more stable and the robot's CoM is maintained in a higher position.

It is relevant to point out that the two different motions for the same terrain can only be obtained with different initial solutions. This is due to the absence of a cost function in our formulation and the fact that the initial solution provided to the solver is a linear interpolation between the initial and final position of the wheels, that is mirrored between the left and right wheels. For that reason, a solution with the wheels going up separately is obtained in this formulation by providing an initial solution with a small shift between the left and right wheels' trajectories. In this case, the initial solution provided for the planner had the left wheels' positions shifted forwards 0.1 m in relation to the right wheels. The "one wheel at a time motion" represents a trajectory that can only be discovered combining terrain knowledge with a 3D model of the robot. As an advantage, such motions are less sensitive to the friction coefficient on the step slope and require less joint torques.

In Figure 4.16(a), the platform with a 65° slope is positioned on the right side of the robot, that drives up the obstacle in less than 0.4 s, showing that terrains with different heights in the y -direction are also successfully



(a) The robot goes up the step by driving both wheels up at the same time.

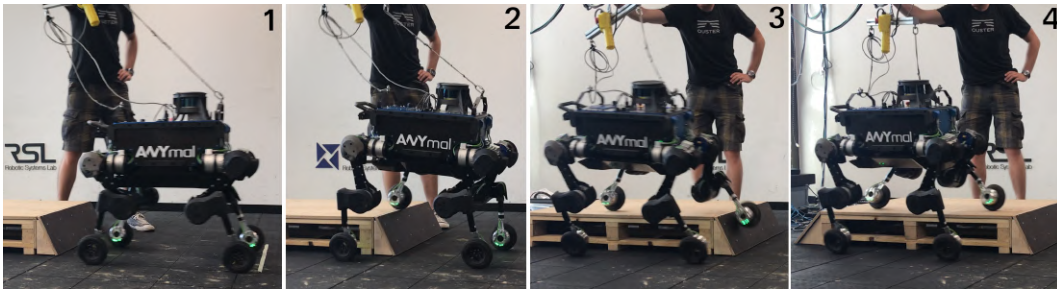


(b) The robot drives up one wheel at a time.

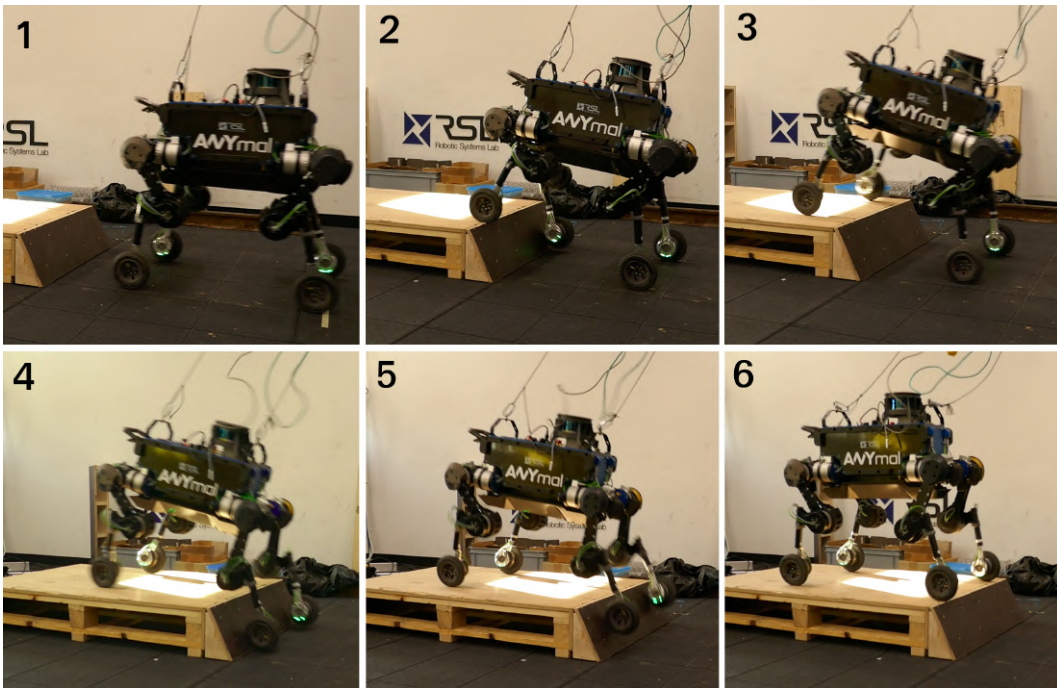
Figure 4.15: ANYmal drives over the step with a 45° slope in two different ways. Both motions are performed in average speed of 0.5 m/s.

negotiated with our TO. To go up, the robot shifts and rolls its base to respect the kinematic limits. In Figure 4.16(b) the robot successfully drives up the the same platform now positioned in the center of its path, proving that our approach can handle steep slopes.

Figure 4.17 depicts the desired motions compared with the measured positions obtained from the experiments for two cases: ANYmal driving up the 45° and the 65° ramp in the center of its path. Our extended version of WBC is able to track the desired motions on the real robot with an average RMSE of 6.5 mm for the base and 22.6 mm for the wheels, in both motions. The errors are computed using the robot's state computed by the state estimator module, which can present some drift if compared to ground truth measurements. Note that the base moves slower while the front wheels go up to maintain enough



(a) The obstacle is positioned on the right side of the path.



(b) The step is aligned with the center of the robot's base.

Figure 4.16: Snapshots of the robot driving up a step 0.2 m high (40% of the legs length) and a 65° linear transition using the TO framework.

traction on the hind wheels; and moves faster when the hind wheels are going up, achieving a maximum speed of 0.88 m/s.

Considering the successful experiments with the 65° slope crossing, it is expected that the robot be able to traverse the stairs presented in Figure 4.14. However, since the planning horizon is larger for such task, the lack of online adaptation of the trajectories and accumulated errors in the tracking controller could prevent the robot of climbing all the steps. One way to mitigate these problems is an implementation of the planner in a receding horizon fashion.

4.3

Conclusions

This chapter presented a motion planning framework that generates driving motions for wheeled quadrupedal robots, optimizing over the base

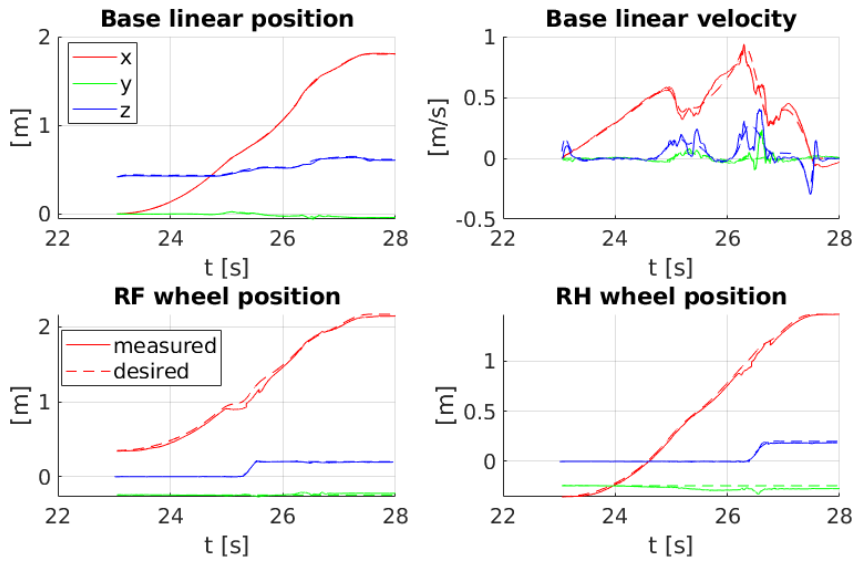
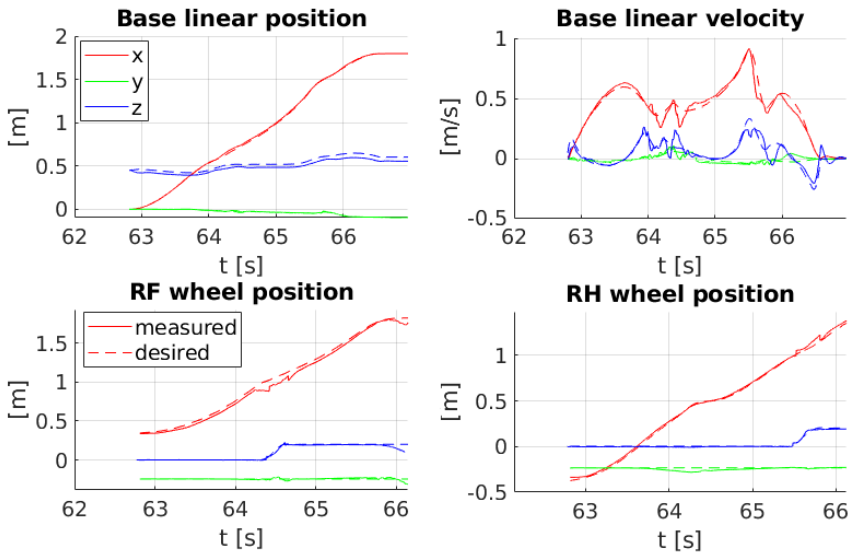
(a) Results for the 45° step.(b) Results for the 65° step.

Figure 4.17: The desired positions of the robot's base and wheels provided as input to the extended WBC (dashed line) and the measured positions (full line) obtained during the experimental tests.

motion and the wheels' positions, velocities and contact forces. The framework consists of a terrain-aware planner and a terrain-aware controller that is an extension of the WBC presented in (Bjelonic et al., 2019). Computing plans with terrain information enables us to generate driving motions over steep obstacles in a non-static manner, achieving an average speed of 0.5 m/s or higher. The feasibility of the trajectories is demonstrated in experiments with the ANYmal robot equipped with wheels driving over different terrains. The results show that we are able to achieve perceptive motion planning over multiple seconds horizons in challenging terrain. Despite model mismatches,

sensor noise and torque tracking inaccuracies, the motion plans are robust enough to be tracked on a real system. Moreover, blind locomotion could not overcome the obstacles or would traverse them in a static step-by-step fashion at a average speed much lower than 0.5 m/s. By using the map of the terrain to plan the trajectory before start moving, we can plan for faster motions.

As for the limitations of the approach, the trajectories are planned for a long time horizon in complex terrains which can be sensitive to drift and tracking error. For that reason, it is recommended a further investigation on the implementation of the motion planner in a receding horizon fashion by using on-board state estimation of the robot. A major concern for making this possible is reducing the cost of the NLP solving. This could be achieved by employing smarter initialization strategies, similar to (Melon et al., 2020), that showed a significantly lower computational cost for a similar NLP formulation.

Even though driving motions can be very efficient, the contact between the wheels and the terrain must be enforced at all times, which precludes its application for discontinuous terrain types, such as small gaps or floating steps, where walking motions are necessary. One possible solution would be to switch between walking and driving based on the characteristics of the terrain. However, this approach does not consider a clear advantage of hybrid wheeled-legged robots, which is the possibility of using wheels and legs simultaneously for locomotion, increasing the robot's speed and mobility. Another improvement of hybrid locomotion is the possibility to perform trajectories with lateral displacement and changes in the robot's heading direction. Such maneuvers are difficult to achieve with purely driving motions for robots with non-steerable wheels due to the restriction on the wheels' rolling direction. With hybrid motions, the wheels' position and direction can be changed while not in contact with the terrain.

Planning hybrid motions for wheeled-legged robots is an interesting and challenging task that involves several constraints to be fulfilled by the planned trajectories. The forces that move the robot can only be created when the wheel is in contact with the terrain. As a consequence, the wheel that is being lifted off does not produce any forces and, therefore, does not significantly contribute to moving the base. Also, the contact forces are unilateral, i.e., the robot can only push on the ground, not pull. These strong constraints on the contact forces and on the interaction between them and the robot's base are better handled with an approach that considers how the interaction forces affect the base motion during the planning task.

In this context, the previous NLP formulation for purely driving motions is extended to generate dynamic hybrid driving-walking motions, still taking into account the terrain information and the dynamics of the robot. This way, driving and stepping motions are both considered in a single planning problem that can generate trajectories with purely driving and purely stepping motions or perform both motions simultaneously.

5.1 TO formulation for Hybrid Motions

For optimizing hybrid walking-driving motions for wheeled-legged robots, the decision variables remain the same as for the previous driving formulation, i.e., the 6-DoF base motion (position and orientation), the wheels' contact positions and contact forces. The main difference is that, besides the robot's initial and final state and the total time duration T of the trajectory, the user must provide a contact schedule to the optimizer. The contact schedule indicates the sequences and durations of the contact phases, as shown in the example in Figure 5.1 for a quadruped robot performing a static¹ walking gait.

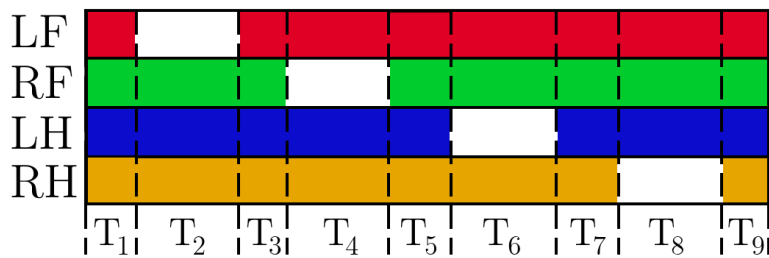


Figure 5.1: Example of a contact schedule for a quadrupedal robot performing a static walking gait. Phases where the wheels are in contact are indicated with colors and lift-off phases are the white spaces. T_i is the duration of each phase.

Another major difference compared to driving motions is that the set of constraints imposed on the wheels' motion depends on the wheels' contact state. When in contact with the ground, the wheel is allowed to drive without slipping and all the known terrain contact constraints are enforced (friction cone, positive normal, terrain height), plus the rolling constraint for consistency with wheeled locomotion. When not in contact, no force is produced on the end-effector and the only constraint is that the wheels stay within the reachable kinematic workspace. These abrupt changes in the set of constraints require a specific multi-phase formulation, discussed in details in the Section 5.2.

Similar to the driving formulation, the robot's dynamics is represented with a Single Rigid Body Dynamics model. In contrast to modeling the robot as a linear inverted pendulum (Siciliano and Khatib, 2007) and using the position of the ZMP to control de robot's motion, the use of a SRBD model includes the contact forces as explicit variables of the optimization problem. This way, they can be directly constrained during the motion planning, making it easier to enforce constraints on slippage and torque limits. In addition, there is no

¹Quadrupedal robots are able to perform statically stable walking gaits, i.e., at least three points of ground contact. This is not the case for robots with fewer legs, which rely exclusively on dynamically stable gaits.

restriction on the base height or orientation, increasing the range of achievable motions.

Figure 5.2 summarizes the TO formulation for hybrid walking-driving motions, based on the formulation proposed by (Winkler et al., 2018) for legged robots with point feet. All constraints are discussed in details in Section 5.3.

$$\begin{aligned}
& \text{find } {}^I\mathbf{r}(t) \in \mathbb{R}^3 && \text{(CoM position)} \\
& {}^I\boldsymbol{\theta}(t) \in \mathbb{R}^3 && \text{(CoM Euler angles)} \\
& \text{for every wheel } i : \\
& \quad {}^I\mathbf{p}_i(t) \in \mathbb{R}^3 && \text{(wheels' motion)} \\
& \quad {}^I\mathbf{f}_i(t) \in \mathbb{R}^3 && \text{(wheels' forces)} \\
& \text{s.t. } [{}^I\mathbf{r}, {}^I\boldsymbol{\theta}](0) = [{}^I\mathbf{r}_0, {}^I\boldsymbol{\theta}_0] && \text{(initial state)} \\
& \quad [{}^I\mathbf{r}, {}^I\boldsymbol{\theta}](T) = [{}^I\mathbf{r}_g, {}^I\boldsymbol{\theta}_g] && \text{(goal state)} \\
& \quad \mathbf{F}_d({}^I\mathbf{r}, {}^I\boldsymbol{\theta}, {}^I\mathbf{p}_i, {}^I\mathbf{f}_i) = \mathbf{0} && \text{(dynamic model)} \\
& \text{for every wheel } i : \\
& \quad {}^I\mathbf{p}_i(t) \in \mathcal{R}_i({}^I\mathbf{r}(t), {}^I\boldsymbol{\theta}(t)) && \text{(kinematic constraint)} \\
& \text{if wheel } i \text{ is in contact:} \\
& \quad {}^I p_i^z(t) = h_{\text{terrain}}({}^I\mathbf{p}_i^{x,y}(t)) && \text{(terrain height)} \\
& \quad C_i \mathbf{f}_i^z(t) > 0 && \text{(normal force)} \\
& \quad \| {}^C_i \mathbf{f}_i^x(t) \| \leq f_{\text{max}} && \text{(maximum torque)} \\
& \quad {}^I\mathbf{f}_i(t) \in \mathcal{F}(\mu, \mathbf{n}, {}^I\mathbf{p}_i^{x,y}(t)) && \text{(friction cone)} \\
& \quad C_i \dot{p}_i^y(t) = 0 && \text{(rolling constraint)} \\
& \text{if wheel } i \text{ is not in contact:} \\
& \quad {}^I\mathbf{f}_i(t) = \mathbf{0} && \text{(no force)} \\
& \quad {}^I p_i^z(t) > h_{\text{terrain}}({}^I\mathbf{p}_i^{x,y}(t)) && \text{(no contact)}
\end{aligned}$$

Figure 5.2: Decision variables and constraints of the TO formulation for hybrid motions. The set of constraints on the wheels' variables depends on the contact state of the wheels.

5.2

Optimization Variables Parametrization

Similarly to the transcription method used for generating purely driving motions, each decision variable is optimized over in discrete times sampled along the trajectory. The continuous motion is then obtained by sequences of third-order polynomials using the Hermite parametrization, which ensures continuous derivatives at the polynomial junctions. For the base 6-DoF motion, the formulation remains the same, with the nodes sampled at a fixed time interval ΔT . The difficulty resides in the formulation for the wheels variables

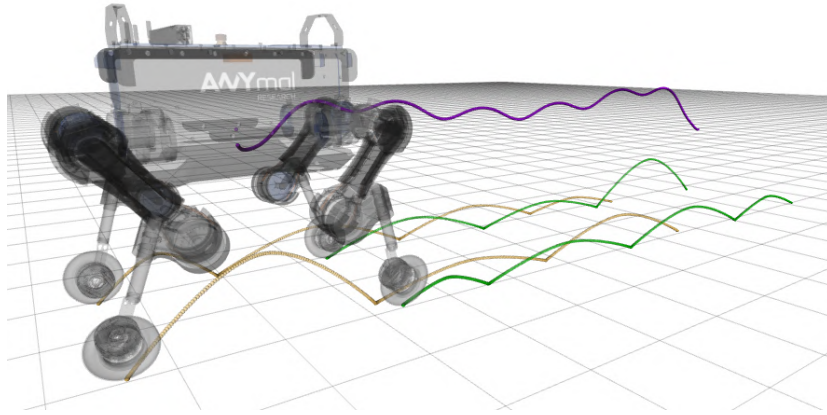
(force and motion), since the set of constraints changes depending on the contact state of the wheels and NLP formulations do not allow constraints to be activated or deactivated arbitrarily during the iterations. To overcome this problem, a phase-based parameterization based on (Winkler et al., 2018) is employed.

Similarly to legged systems with point feet, the wheels can either be in contact with the ground (contact phase) or not (flight phase). The main difference is that the velocity of the wheel's contact point is no longer forced to zero. Figure 5.3 illustrates the difference between purely walking and a hybrid motion when the wheels are in contact with the terrain. In purely walking (Figure 5.3(a)), the wheels remain stationary while in contact with the ground. In hybrid locomotion (Figure 5.3(b)), the wheels are allowed to drive when in contact.

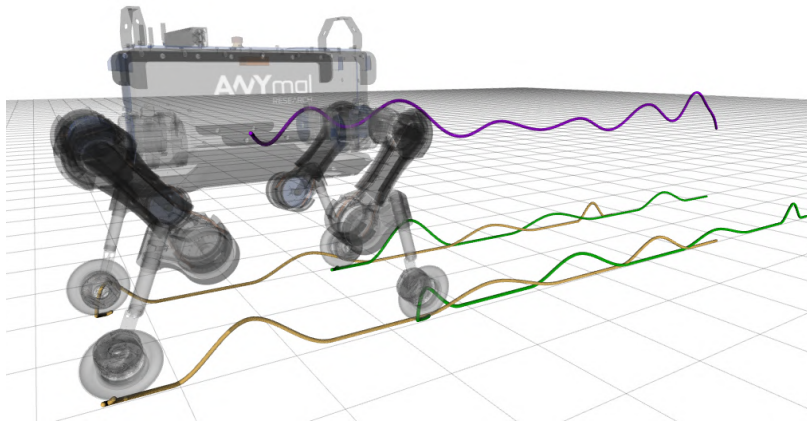
In phase-based parametrization, each wheel motion is treated individually, alternating between contact and swing phases. The sequence and durations of each phase are specified in advance by the user and phase-specific constraints can be enforced directly at the polynomial junctions, the same as in a continuous problem. Figure 5.4 illustrates the wheels parametrization for performing a hybrid driving-walking motion on flat terrain. Each phase is represented by a sequence of Hermite polynomials and the number of polynomials changes depending on the contact state. Each phase has a fixed duration ΔT_j , for $j = 1 \dots n_p$, where n_p is the total number of phases.

The contact phases are treated as regular driving phases with nodes sampled at a fixed interval Δt_d , generating $n_j = \text{floor}(\Delta T_j / \Delta t_d)$ polynomials for each contact phase j . Even though this is a parameter that can be easily changed, it is important that Δt_d is not very large, so that it is possible to generate feasible trajectories even for gaits with long contact phases. A clear example are driving motions, which are defined by a single contact phase with a long duration. In contact phases, both the wheels' positions and forces are parametrized in the same way, with the same number of nodes.

On the other hand, the swing phase is mainly characterized by no contact between the wheel and the ground, which implies zero contact forces. This is ensured in the parametrization by fixing a constant zero value for the forces in all swing phases, exactly as implemented in (Winkler et al., 2018). This restriction is not defined as a constraint of the NLP, but the wheel's contact forces are simply initialized as zero and not optimized over in the nodes adjacent to swing phases. For the wheels' motion, at least two polynomials are necessary, so the wheel can be lifted off and then lowered back down, as depicted in the white areas of the p_z graph in Figure 5.4. Since the terrain



(a) Purely walking trajectory: the wheel has zero velocity while in contact with the ground.



(b) Hybrid driving-walking trajectory: the wheel is allowed to move in contact phase.

Figure 5.3: Gazebo simulation of the ANYmal robot equipped with torque controllable wheels executing different trajectories generated by the motion planner. The purple line is the base trajectory, the green lines and the yellow lines are respectively the front wheels and hind wheels' trajectories.

constraints are enforced on the polynomial junctions, more polynomials might be necessary depending on the complexity of the terrain to ensure no collision with the obstacles during the lift-off or touch down movements of the wheels.

5.2.1 Contact Schedule

With the present formulation, there is no restriction on the gait patterns that can be used for defining the sequences and durations of the contact phases. Figure 5.5 shows examples of gaits used for the trajectories generation in this work. Tests were performed with both statically and dynamically stable gaits, including gaits with full flight phases, in which neither of the wheels is in contact with the ground for a short period of time.

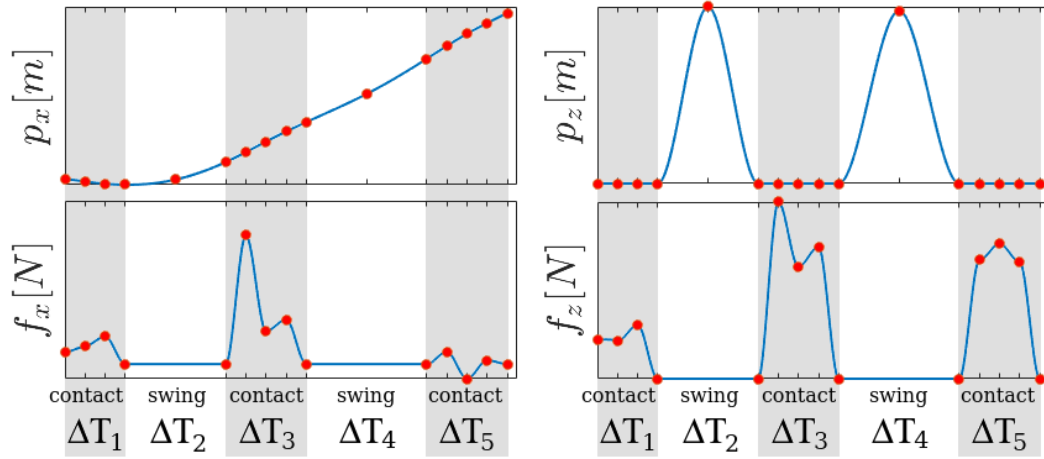


Figure 5.4: Phase-based parametrization for the wheels' contact points and forces. Nodes (red dots) in the gray area are swing nodes while nodes in the white area are contact nodes. Graphs p_x and p_z are the x and z dimension of the wheel contact point trajectory, while graphs f_x and f_z represent the x and z dimension of the wheel contact force.

For unstructured terrain containing abrupt changes in height or gaps, gaits such as a hybrid gallop or a hybrid bounding are usually appropriate choices, while for terrains fairly regular and without large obstacles, a faster gait is more efficient, such as a hybrid running trot or driving.

5.3 NLP Constraints

Most of the constraints for the hybrid motion planning task are similar to those previously defined for the driving formulation, aside from the fact that they are applied for specific times depending on the gait pattern for the motion. This section briefly reviews the constraints of the NLP considering the phase-based parametrization of the wheel's variables.

Consider the Figure 5.6, that depicts the decision variables of the optimization for a wheeled quadrupedal robot during a dynamic hybrid running trot motion. In this case, the LF and RH wheels are in swing phase, have non-zero velocity ($\dot{\mathbf{p}}_1$ and $\dot{\mathbf{p}}_4$) and zero contact force. In contrast, the wheels RF and LF are in contact phase, they also have non-zero velocity ($\dot{\mathbf{p}}_2$ and $\dot{\mathbf{p}}_3$), but constrained to the rolling direction of the wheel, and their respective contact forces (\mathbf{f}_2 and \mathbf{f}_3) balance the inertial forces on the CoM for the duration of the current phase.

For generating hybrid motions, the dynamic and kinematic constraints are the same as defined in (4-2) and (4-4), respectively. Different from the phase-depending constraints, they both are enforced at a fixed time interval along the entire trajectory.

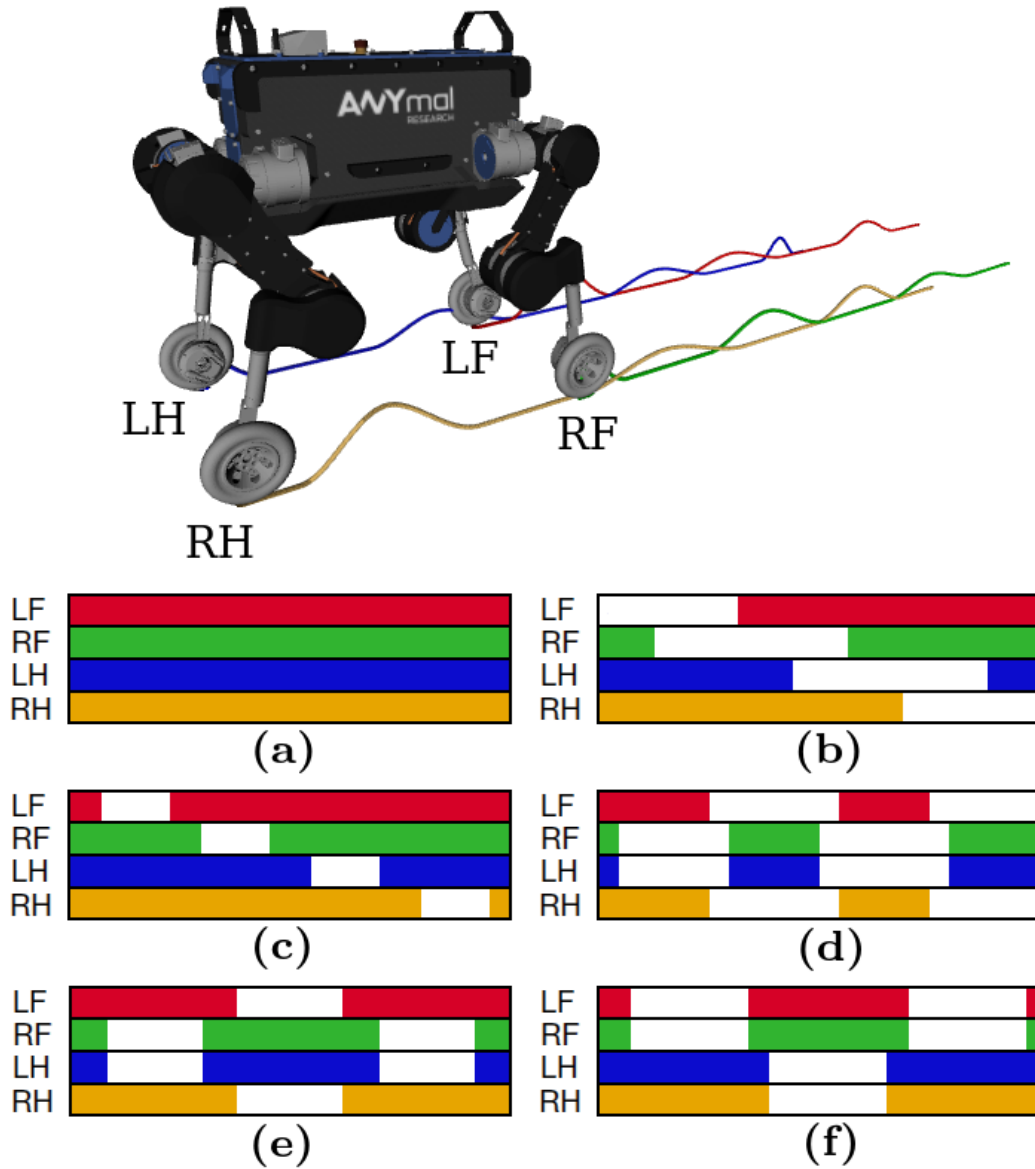


Figure 5.5: Some examples of gaits used for hybrid trajectory generation: (a) driving, (b) hybrid gallop, (c) hybrid walk, (d) hybrid running trot, (e) hybrid trot, (f) hybrid bounding. Contact phases are indicated with colors and swing phases with white spaces. An example of the wheels' trajectories for the hybrid running trot are indicated in the top figure, each with a different color, matching the colors in the gait pattern graphs.

5.3.1

Force Constraints

During contact phases, the following constraints are enforced on the wheels' force profiles: unilateral constraint, friction cone and maximum wheel torque. They are all defined in the following equations, for which \mathcal{S}_i indicates the set of time intervals in which the i^{th} wheel is in contact (stance) phase.

The unilateral constraint ensures the contact force always pushes into the terrain, which is equivalent to constrain the component of the contact force

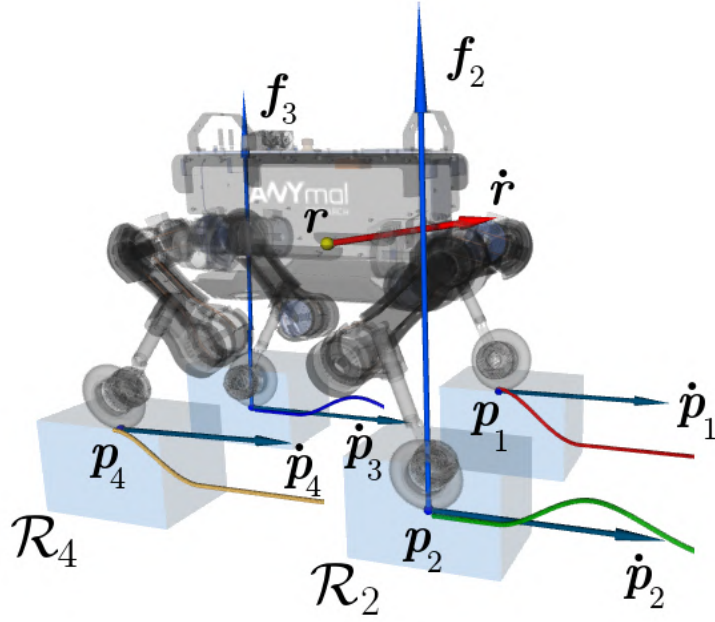


Figure 5.6: Variables of the NLP for a quadrupedal robot performing a hybrid running trot. The parallelepiped \mathcal{R}_i indicates the feasible workspace for i^{th} the wheel.

orthogonal to the terrain to be positive. This translates as follows:

$$C_i f_i^z(t \in \mathcal{S}_i) > 0 \quad (5-1)$$

where $C_i f_i^z(t)$ is the z component of the contact force on the i^{th} wheel expressed in the contact frame. The contact frame C_i is defined in the same way as it was for the purely driving formulation, exemplified in Figure 5.7 for a flat terrain. The axis \mathbf{c}_z is aligned with the terrain normal at that position, and the axis \mathbf{c}_x is perpendicular to \mathbf{c}_z and aligned with the rolling direction of the wheel.

For ensuring no slippage while driving, the contact forces are constrained to remain inside the friction pyramid that approximates the friction cone on each wheel, indicated in light red in Figure 5.7. The constraint is given by:

$$-\mu C_i f_i^z(t \in \mathcal{S}_i) \leq C_i f_i^x(t \in \mathcal{S}_i) \leq \mu C_i f_i^z(t \in \mathcal{S}_i) \quad (5-2)$$

$$-\mu C_i f_i^z(t \in \mathcal{S}_i) \leq C_i f_i^y(t \in \mathcal{S}_i) \leq \mu C_i f_i^z(t \in \mathcal{S}_i) \quad (5-3)$$

where μ is the friction coefficient of the terrain.

Lastly, the torque limits from the wheel's actuators are imposed by limiting the traction force $C_i f_i^x$ to a maximum value, as such:

$$-\tau_{max}/r_w \leq C_i f_i^x(t \in \mathcal{S}_i) \leq \tau_{max}/r_w, \quad (5-4)$$

where τ_{max} is the maximum allowed torque by the wheels' motors and r_w is

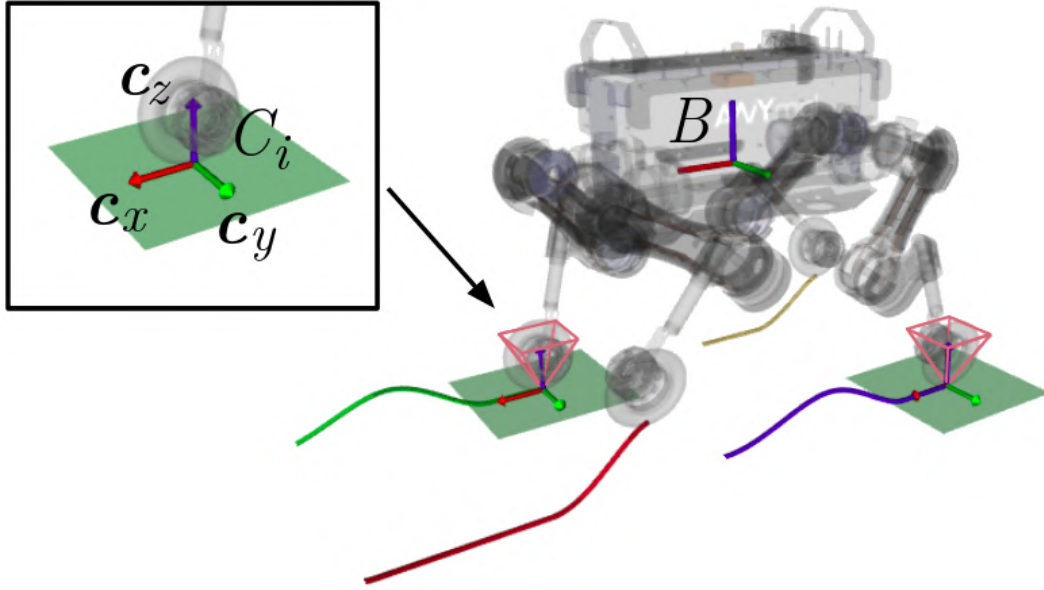


Figure 5.7: Coordinate frames used for the hybrid trajectory optimization. The frame B is attached to the CoM of the robot and each wheel in contact phase has a frame C_i attached to its contact point with the ground.

the wheel's radius.

All of these constraints are enforced on the polynomial junctions belonging to contact phases. For swing phases, the wheels' forces are set as zero and are not optimized over.

5.3.2 Motion Constraints

The motion constraints also differ depending on the contact state of the wheel. When in contact, the wheel can have a non-zero speed or acceleration, but limited to the rolling direction of the wheels for consistency with driving locomotion. This is enforced by constraining the y -velocity of the wheels' contact point to be zero:

$${}^{C_i} \dot{p}_i^y(t \in \mathcal{S}_i) = 0 \quad (5-5)$$

where ${}^{C_i} p_i^y(t)$ is the y component of the i^{th} wheel contact position expressed in the contact frame. It is important that this constraint is only enforced during contact phases, since the wheel is allowed to move in all directions when in swing phase.

In addition, terrain contact is enforced using a 2.5D height map of the terrain, either analytically defined or computed from vision data, as follows:

$${}^I p_i^z(t \in \mathcal{S}_i) = h_{terrain}({}^I \mathbf{p}_i^{x,y}(t)) \quad (5-6)$$

On the other hand, the following constraint ensures no contact with the ground during swing phases:

$${}^I p_i^z(t \notin \mathcal{S}_i) > h_{\text{terrain}}({}^I \mathbf{p}_i^{x,y}(t)) \quad (5-7)$$

It is worth noting that this constraint is only enforced on the polynomial junctions inside the swing phases. Depending on the terrain complexity and on the obstacle size, more polynomials might be necessary to ensure no collisions with the terrain. A ground clearance threshold can also be included in this constraint for further safety, if necessary.

An important issue that must be addressed specifically for planning hybrid motions is ensuring a continuous acceleration profile during the driving phases. As discussed in Section 4.1.3, the Hermite parametrization allows for discontinuities on the acceleration profile of the wheels, which causes *jittering*² on the wheel's motions and abrupt variations in the contact forces.

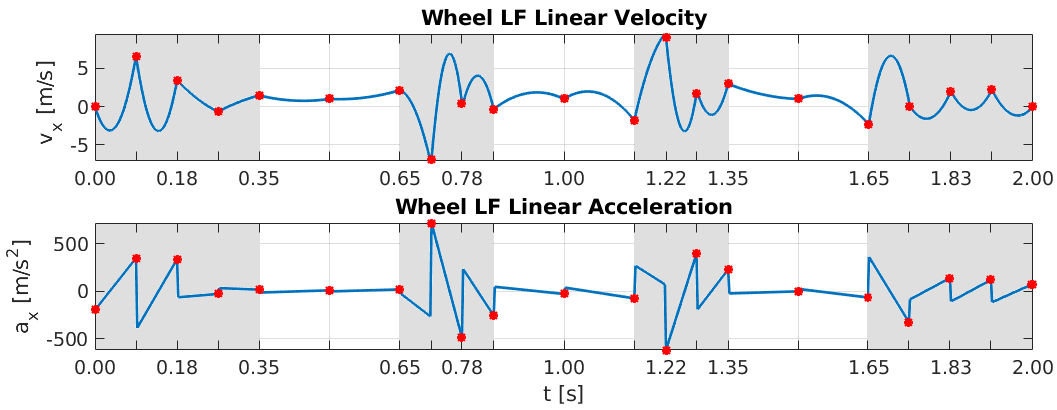
When generating purely driving motions, the trajectories are always continuous and the constraint on the acceleration continuity is simply imposed in all nodes. This is not the case for hybrid motions. The switching from contact to swing phase implies a rapid transition from zero velocity to lift-off speed. For that reason, the accelerations are only constrained to be continuous within the contact phase, but free to vary on the swing nodes. However, the accelerations in all directions are constrained to a maximum value to prevent unreasonable contact forces.

The effect of not including the acceleration constraints can be seen in Figure 5.8, that shows the LF wheel's velocity and acceleration profile in the x -direction for the robot performing a hybrid running trot motion at an average speed of 1.0 m/s in a straight direction. For the same motion, Figure 5.9 shows the position, velocity and acceleration for the same wheel in the z -direction. The transition between the contact and swing phase generates a fast change in the acceleration in that direction, which is expected.

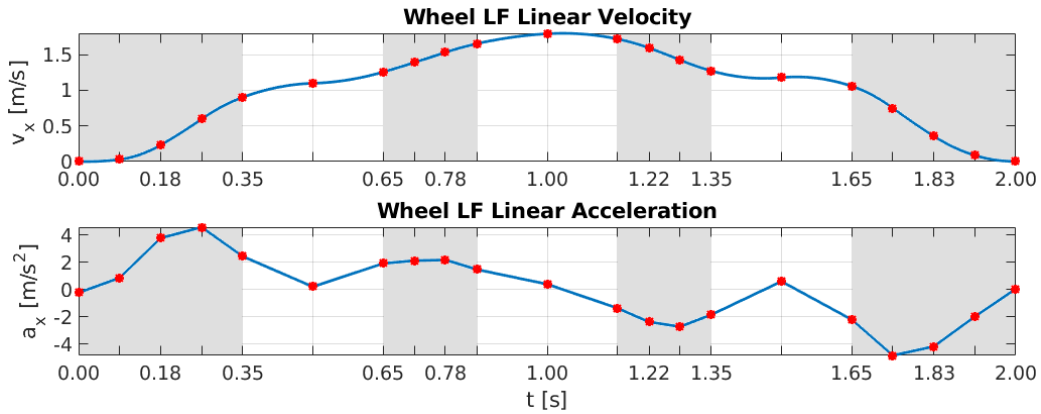
5.4 Results

This section presents the implementation and testing of several motions generated by the hybrid TO framework. The motion plans are validated in physical simulations with the ANYmal robot equipped with non-steerable wheels in different terrains, including floating steps and gaps. These scenarios

²The term *jittering* in this context refers to abrupt oscillations back and forth on the wheel's motions in its rolling direction.



(a) Wheel's motion generated **without** the acceleration continuity constraint.



(b) Wheel's motion generated **with** the acceleration continuity constraint.

Figure 5.8: Influence of the acceleration continuity constraint on the generated motions. The red dots are the polynomial junctions and the gray areas are the contact phases.

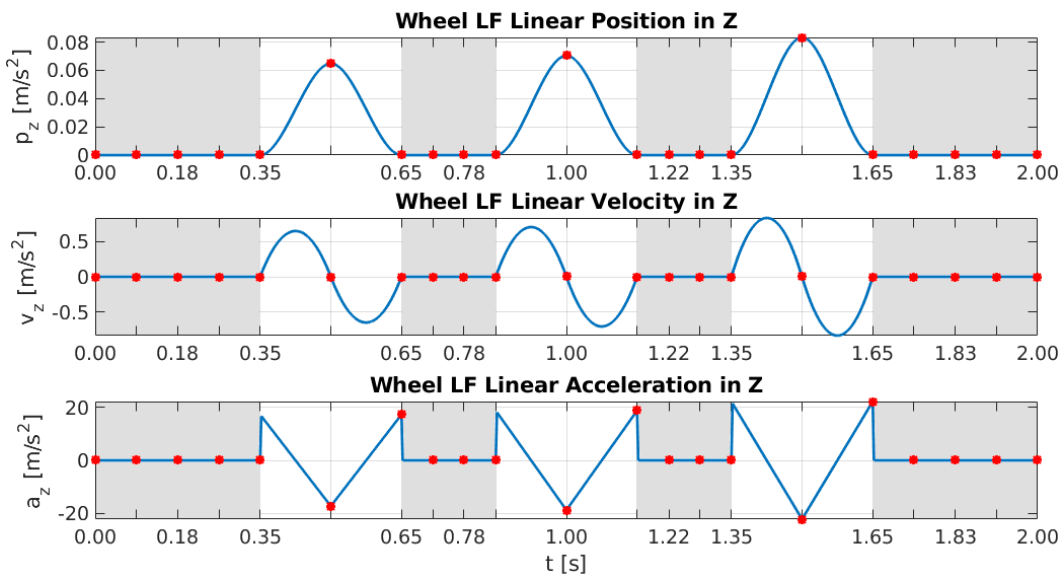


Figure 5.9: Position, velocity and acceleration profiles of the robot's LF wheel in the z -direction, with the acceleration constraint.

are examples of situations where stepping is required and for which hybrid motions are particularly well suited.

5.4.1 Implementation

The TO framework for hybrid motions is implemented in C++ using the Ifopt (Winkler, 2018) interface for the interior-point method solver Ipopt (Wächter et al., 2006). All derivatives are provided to the solver analytically, which significantly improves its performance. The framework is tested in simulations with the real quadrupedal robot ANYmal equipped with actuated non-steerable wheels. The simulations are carried out in the robot simulation environment Gazebo (Koenig and Howard, 2004) with ODE (Russell, 2008) as the physics engine. To ensure realistic results, we use the full rigid body dynamics of the robot, accounting for the torque limits of the actuators.

Given the contact schedule, the height map and the goal state, the TO framework computes the reference trajectories for the base and the wheels, as well as the wheels' contact forces. The planned trajectories are provided as input to the WBC, which computes the actuation torques for the joints and the wheels at a 400 Hz frequency while accounting for several constraints, such as actuator limitations, friction cone and the nonholonomic rolling constraints. Figure 5.10 gives an overview of the motion planning framework for hybrid motions.

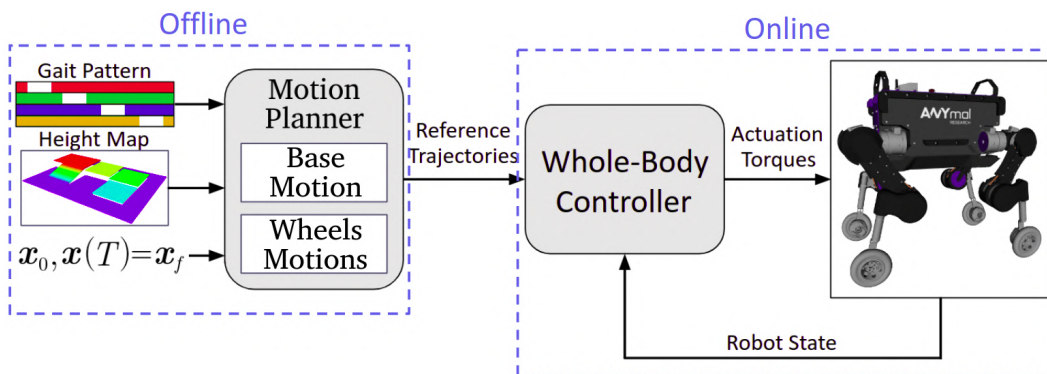


Figure 5.10: Overview of the motion planning framework for hybrid motions. The motion planner computes the reference trajectories for the base and the wheels that are given as input to the WBC, that computes the actuation torques for the robot.

For the trajectory optimization, the kinematic constraints are enforced at a time interval of 0.08 s and the dynamic constraints are enforced every 0.1 s, which is enough to ensure the feasibility of the motion plans. On driving phases, the force and motion constraints are enforced every 0.1 s. For swing

phases, two polynomials were used for describing the motion in most cases, but three polynomials were also used to improve the convergence for more difficult cases, such as going up a step. Overall, the hybrid TO problem consists of a non-linear non-convex optimization, which is a challenging task to solve. The computational cost for different scenarios is presented in the following sections along with the test results.

For the optimization, the base and the wheel's trajectories are initialized with a linear interpolation between the initial and final position of the robot, assuming the average speed during the entire motion. As discussed in detail in Chapter 3, we avoid using a cost function in the formulation to keep a low computational cost, so the objective is simply to find a feasible solution. The continuous 2.5D height map of the terrain required for the motion planning was analytically defined for each of the terrains.

5.4.2 Simulations

The simulations are carried out in the robot simulation environment Gazebo with different terrain types. Firstly, several dynamic hybrid motions are demonstrated in flat terrain with different gaits and different final positions for the base, requiring lateral displacement of the wheels during the swing phase. Then, further testing shows the robot performing dynamic hybrid motions to overcome a 0.25 m gap and a floating step with a 0.2 m height in different positions. Scenarios with discontinuous obstacles require a motion planner that takes into account the terrain information, which is the case for our framework.

5.4.2.1 Flat Terrain

We demonstrate the robot performing dynamic hybrid motions in flat terrain with several different gaits. Figure 5.11 shows some of the motions plans generated by the hybrid TO framework. The wheels' trajectories are indicated with different colors (LF – red, RF – green, LH – blue, RH – yellow) and the base trajectory is indicated in purple. The first motion, in Figure 5.11(a), is a simple driving maneuver at an average speed of 1.0 m/s, where the wheels must stay in contact with the terrain during the entire trajectory. The second (Figure 5.11(b)) is a hybrid gallop motion at an average speed of 0.67 m/s, and the third (Figure 5.11(c)) is a hybrid running trot, with full flight phases, at an average speed of 1.0 m/s.

Figure 5.12 shows the desired motion computed by the TO along with the measured positions obtained from the simulation of the hybrid running trot motion. For all the hybrid motions tested in flat terrain, the WBC was able to track the motion plans successfully, with an average RMSE of 9.5 mm for the base and 13.1 mm for the wheels. Note that the errors are slightly larger during the swing motions, which are harder to track when compared to rolling motions in terms of required actuation effort. In an attempt to improve the tracking of the swing motions, the task “swing leg motion tracking” was given a higher priority in the hierarchical WBC (Bjelonic et al., 2019), but that change did not translated into significant improvement in the simulation results.

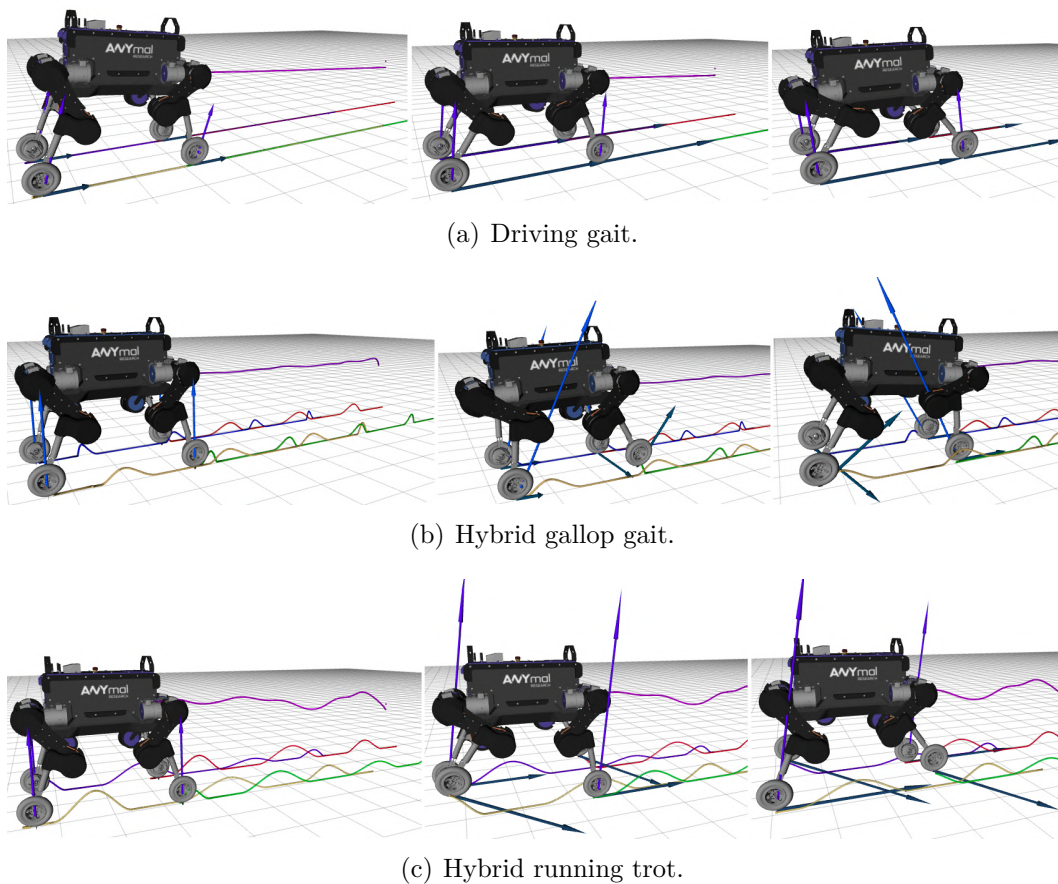


Figure 5.11: The motion plans for the base and the wheels generated for different gaits in flat terrain.

The main advantage of hybrid motions for flat terrains is the ability to perform trajectories with lateral displacement or changes in the base’s orientation. The interspersed swing phases in the wheels’ trajectories allow for the lateral displacement of the wheels while in movement, which is not possible with a purely driving motion. For the motion presented in Figure 5.13, the final position of the base was diagonally shifted of 1.0 m from the initial position

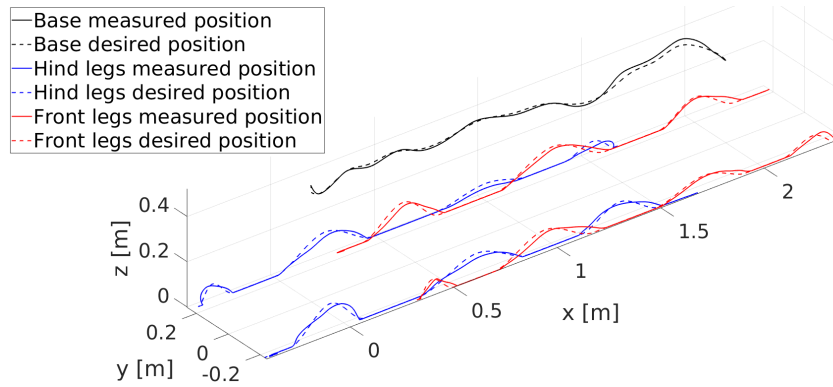


Figure 5.12: The desired positions provided as input to the WBC and the measured positions from the simulation of the robot performing a hybrid running trot motion at an average speed of 1.0 m/s.

in the y -direction and 2.0 m in the x -direction. The robot reaches the goal by performing a lateral hybrid trot motion. When in contact, the wheels only move in their rolling direction, as constrained in the formulation.

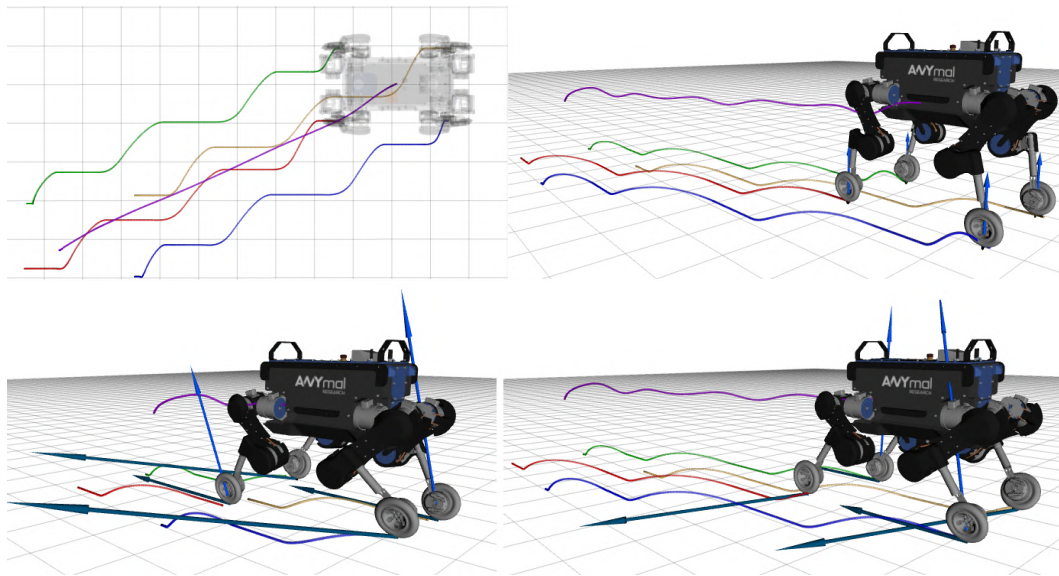


Figure 5.13: On the top left, a top view of the lateral hybrid trot motion, where the robot has to go from position $(x, y) = (0.0, 0.0)$ to $(x, y) = (2.0, 1.0)$. On the top right, an overview of the motion, in which it is possible to see the swing phases. At the bottom, two snapshots of the motion's simulation showing the lateral displacement of the wheels.

Figure 5.14 shows the behavior obtained for the same goal position and same gait pattern, but the final orientation of the base is rotated 90° with respect to the z -axis. In this case, the robot starts with a hybrid lateral trot and when is near the target position, it rotates the base using steps. The total time duration for the motion is set for 4.0 s. Figure 5.15 depicts the motion plans provided as input to the WBC and the measured data obtained from

the simulation. The tracking RMSE for this case was 6.8 mm for the base and 15.7 mm for the wheels. During the motion, the base moves at a higher speed during the lateral locomotion, reaching almost 1.5 m/s in the longitudinal direction, and changes to a nearly stationary rotation in the second half of the motion.

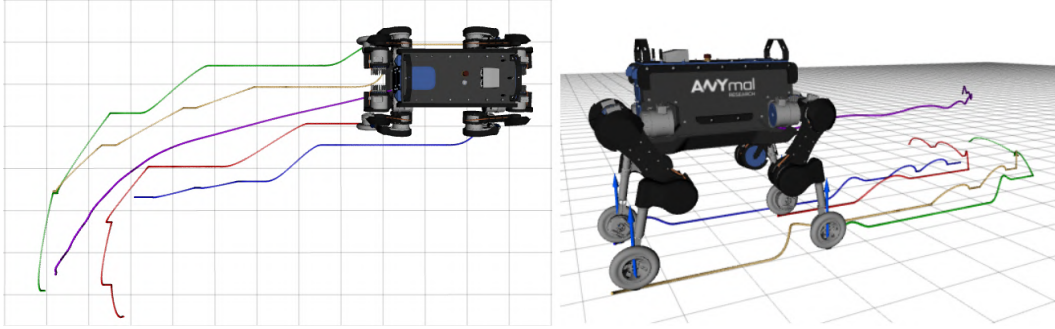


Figure 5.14: The robot moving laterally and then turning 90° with a dynamic hybrid trot gait.

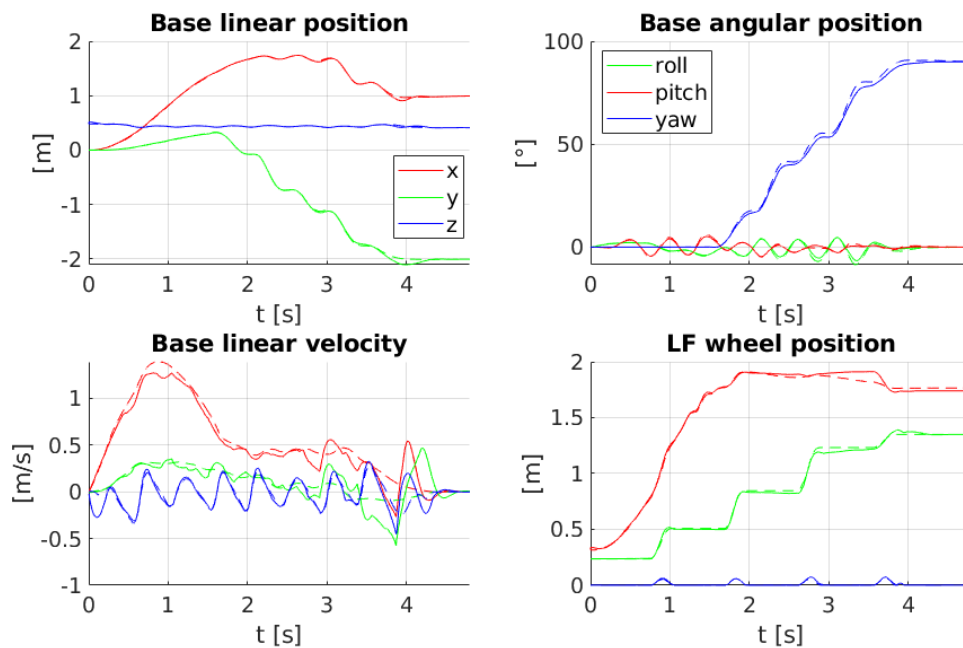


Figure 5.15: Plot results for the combined lateral motion and base turning. The dashed lines are de desired motions, with the full lines are the data obtained from the whole-body simulation.

The solver's computational times for the presented maneuvers depend on the optimization parameters and the motion complexity, but remained on average 0.51 s per second of the trajectory. Table 5.1 shows an overview of the computation cost for several hybrid motions tested in flat terrain. Gaits with fewer instances of lift-off motions are computed at least two times

faster, and motions with large changes in the base’s orientation require more computational effort.

Table 5.1: Computation times for some of the maneuvers used to test the hybrid TO framework in flat terrain.

Maneuver	Time horizon	Goal Position (x,y)	Final Yaw angle	Comp. time
Driving	2.0 s	(2.0, 0.0) m	0°	0.56 s
Hybrid running trot	2.0 s	(2.0, 0.0) m	0°	1.17 s
Hybrid gallop	3.0 s	(2.0, 0.0) m	0°	1.73 s
Hybrid trot	2.0 s	(2.0, 0.0) m	0°	0.64 s
Lateral hybrid trot	2.0 s	(2.0, 1.0) m	0°	0.76 s
Lateral hybrid running trot	2.0 s	(2.0, 1.0) m	0°	1.87 s
Lateral hybrid trot	4.0 s	(2.0, 1.0) m	60°	1.23 s
Lateral hybrid trot	4.0 s	(2.0, 1.0) m	90°	2.70 s

Similar agile hybrid motions have been shown for flat terrain using linearized ZMP constraints for the ANYmal robot with non-steerable wheels in (de Viragh et al., 2019). However, one of the main advantages of hybrid motions is that it enables dynamic obstacle crossing when combined with terrain knowledge, which is verified in the following sections.

5.4.2.2

Gap

The robot’s gained ability to traverse discontinuous obstacles with dynamic hybrid motions is demonstrated in Figure 5.16 and Figure 5.18, in which the robot is required to cross a gap with a 0.25 m width at an average speed of 0.75 m/s. Two gaits were used for this maneuver: the hybrid bounding gait, shown in Figure 5.5(f), where the robot hops with both front wheels followed by a full stance phase and then a hop with the hind wheels; and a custom gait based on the hybrid gallop (Figure 5.5(b)), adjusted to include a full stance phase after the swing of the RF wheel. Such a contact sequence allows the solver to generate trajectories that have the front wheels on one side of the gap and the hind wheels on the other side, even if for a brief period of time.

The bounding gait maneuver is shown in Figure 5.16. The robot starts lifting the front legs prior to the obstacle while the hind legs continue to drive forward. When hopping with the front wheels, the robot moves the base backwards to increase the support forces on the hind wheels and maintain stability. This behavior is only possible because we take into account the

dynamics of the robot during the motion planning. Once the front wheels are on the ground, the hind wheels are moved to the other side of the gap, completing the maneuver. The solver took 0.82 s to generate this motion. The data plots for the simulation results, presented in Figure 5.17, show that the WBC was able to track the desired motions with an average RMSE of 20.1 mm. It is interesting to notice that the robot needs to reduce its speed right after the touchdown of the front wheels to adjust itself after the impact on the front wheels, which is not considered in the model. Even though the average speed for the motion is 0.75 m/s, the robot reaches a maximum speed of 1.75 m/s during the hops.

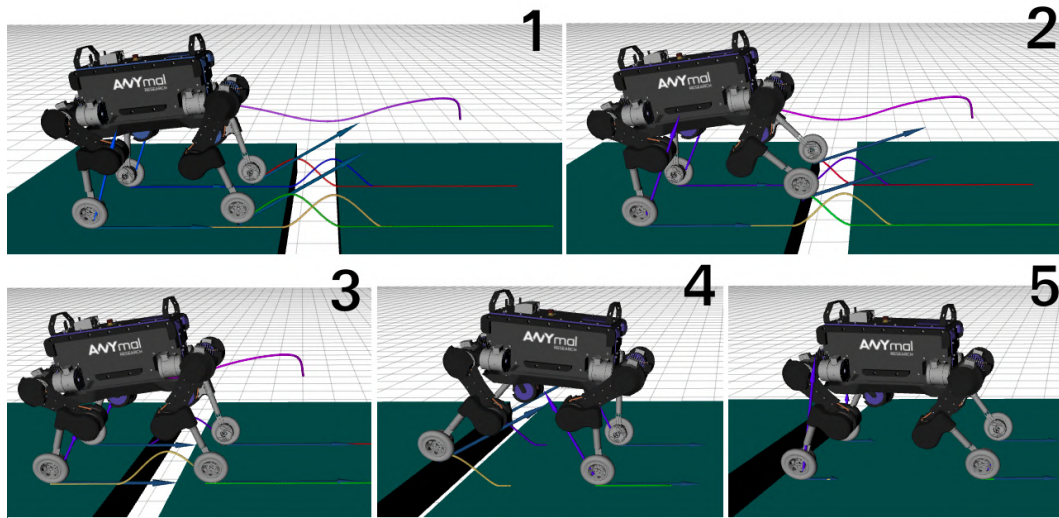


Figure 5.16: The ANYmal robot performing a dynamic hybrid bounding gait to cross a 0.25 m gap at an average speed of 0.75 m/s.

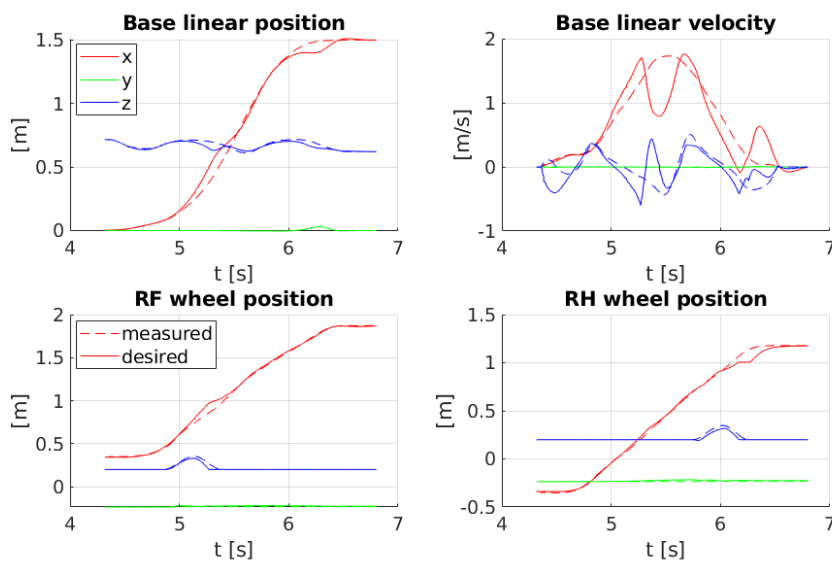


Figure 5.17: Simulation results the hybrid bounding maneuver over the gap.

Figure 5.18 shows the robot crossing the same gap using a different contact schedule, similar to a hybrid gallop gait, but with an intermediate stance phase. In this case, the LF wheel starts the movement over the gap and before it reaches the other side of the breach, the RF wheel enters the swing phase. Then, a similar sequence is executed by the hind wheels. The measured data from the motion simulation is presented in Figure 5.19. The solver took 1.58 s to optimize this trajectory. In comparison to the hybrid bounding gait, this maneuver presents a smaller RMSE tracking error, of less than 14 mm on average, but it required almost twice the computational effort. Regardless, the robot was able to overcome the gap with both strategies and they can be successfully employed for such terrains.

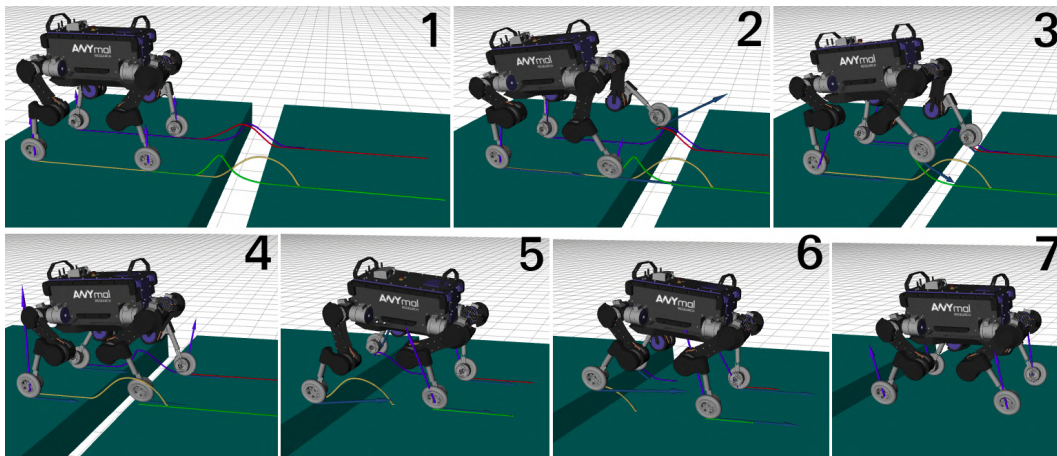


Figure 5.18: The ANYmal robot performing a dynamic hybrid gallop gait to cross the same gap.

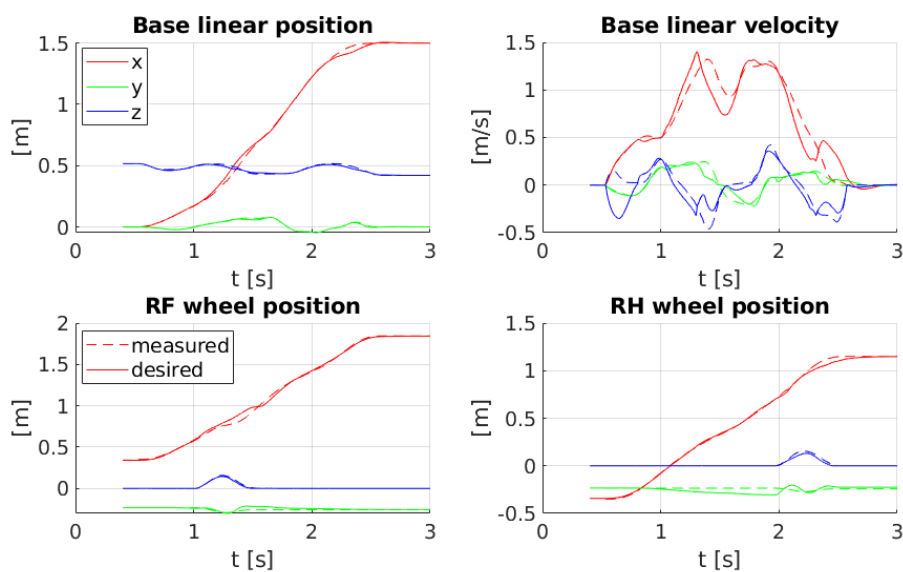


Figure 5.19: Simulation results for the hybrid gallop maneuver over the gap.

As an additional comment, the NLP formulation includes constraints that depend on the terrain height-map, which means that discontinuities in the terrain profile will affect the gradients and may hinder the convergence of the solver. To overcome this problem, the abrupt changes in the terrain are smoothed to ensure convergence. For the optimization, the gap was modeled as a 1.5 m deep parabola, as exemplified in (Jelavic and Hutter, 2019). The same procedure is done for floating steps by inserting a linear transition of a very steep slope (~ 10) between the ground and the step.

5.4.2.3 Floating Step

Another challenging obstacle that requires hybrid driving-walking motions are floating steps, that are disconnected from the ground. The same motions could be applied to regular steps as well. Figure 5.20 shows the robot traversing a floating step with a 0.2 m height (40% of the leg's length) at an average speed of 0.75 m/s, with the same hybrid gallop gait previously designed for the gap terrain. Note how the robot lifts the front left wheel prior to the obstacle and it is already in the highest position when it reaches the step, which speeds up the maneuver. Even before the LF wheel reaches the step, the RF wheel starts to lift-off the ground. The same procedure is then carried out for the hind wheels to climb up the step. This maneuver was generated in 1.13 s by the solver. Figure 5.21 confirms the successful tracking of the motions by the WBC, with an RMSE less than 12 mm in average. The largest errors happened during the RH wheel lift-off maneuver. The desired trajectory for the RH wheel has the peak of the swing phase in a higher position and the robot was not able to lift the wheel so far up. However, the climbing maneuver was completed without any collisions. A more restrictive limit on the z-acceleration of the wheels could prevent that, but it would also decrease the number of possible motions that can be discovered by the planner. Moreover, the maneuver was completed in 2.0 s. To the best of the author's knowledge, such a fast negotiation of a step with a height of four times the wheel's radius (0.05 m) with a dynamic hybrid motion has never been shown before.

The next test consists of placing the same step on the right side of the robot's path, with the target position at a straight distance from the robot. This way, it is not possible to deviate from the step, the robot has to go over it. A specific contact schedule was designed for such asymmetric obstacles, where the left wheels are allowed to remain in contact with the ground for the entire trajectory. The planned motion is presented in Figure 5.22, which shows the left wheels always in driving phase, while the right wheels go up the

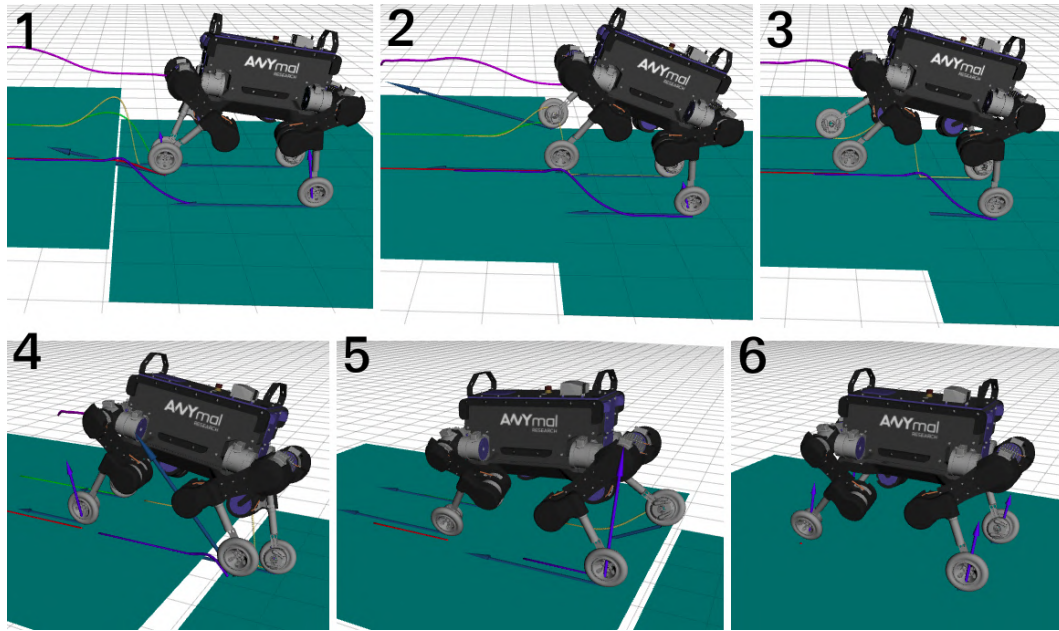


Figure 5.20: The ANYmal robot performing a dynamic hybrid gallop gait to cross a step with a 0.2 m height.

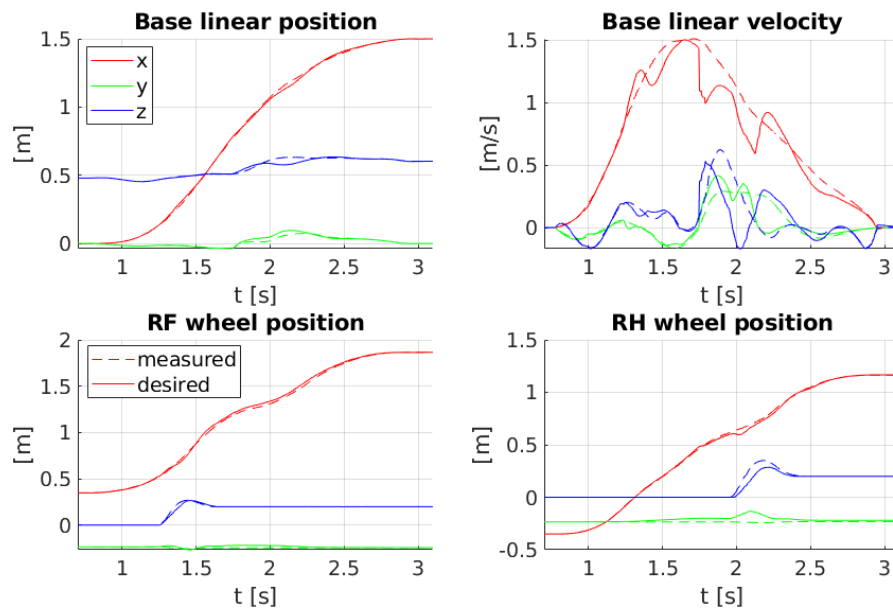


Figure 5.21: Comparison between the reference trajectories provided to the WBC and the obtained measurements from simulations.

step. Moreover, the base finishes with a 10° roll angle to stay clear from the kinematic limits of the wheels. Similar to all the other tests, the maneuver is executed at an average speed of 0.75 m/s, but the robot reaches a maximum speed of 1.2 m/s. In general, this is an easier motion to execute, since only one wheel is lifted up at a time. This is translated to a RMSE tracking error of less than 3.0 mm for both the base and the wheels, as it can be seen in Figure 5.23.

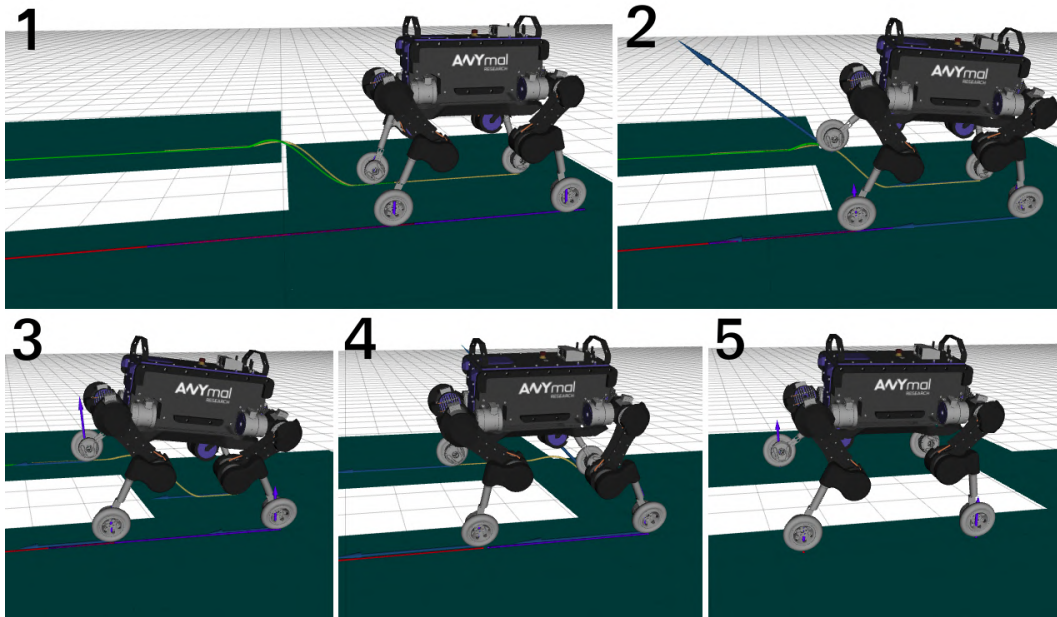


Figure 5.22: The ANYmal robot crossing a 0.2 m high step on the right side of its path.

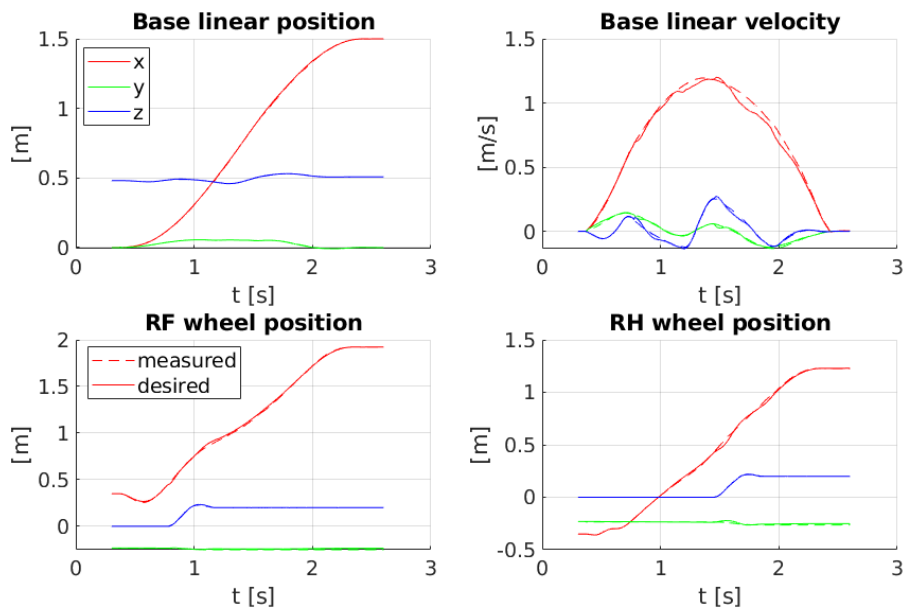


Figure 5.23: Simulation results show barely any error on the trajectory tracking for the lateral step. The executed trajectories are almost identical to the desired motions.

5.5 Conclusions

This chapter presents a TO framework for wheeled-legged robots that is able to generate purely driving motions, purely walking motions and simultaneous walking and driving motions. Such approach takes full advantage of the hybrid nature of robots with legs and wheels and their high mobil-

ity in rough terrain. The proposed formulation takes into account the terrain information and the robot dynamics, which allows wheeled-legged robots to overcome challenging and discontinuous obstacles with dynamic hybrid motions. The optimized trajectories are verified in physical simulations with the ANYmal robot equipped with actuated wheels in different scenarios at speeds equal or higher than 0.5 m/s. The robot is shown to perform dynamic turns and lateral motions in flat terrain, and crossing discontinuous obstacles in a dynamic manner. Moreover, the approach can handle both statically and dynamically stable gaits, including a hybrid running trot and a hybrid gallop gait.

For the motion planning, the gait timings given as input to the solver are considered fixed and the optimized hybrid trajectory has the same contact schedule as the pre-defined gait sequence. The work in (Winkler et al., 2018) proposes a formulation that allows the optimization of the gait sequence in which the durations of each phase are also decision variables of the NLP. Future work includes adapting this approach for wheeled-legged robots performing dynamic motions. The implementation of a gait optimization feature in our formulation would enable the automatic switch between driving and hybrid motions. The optimizer would be able to increase the duration of constant phases so the robot would drive for most of the time in flat terrain, for example. The downside would be the significant increased computational cost associated with contact schedule optimization.

Another point of improvement is to consider the distance from the obstacles in the planning problem, preventing collisions and avoiding trajectories that place the wheels dangerously close to the edges of the terrain, which would improve the reliability of the motion plans. The authors in (Bratta et al., 2020) address a similar problem for the original formulation proposed in (Winkler et al., 2018) for legged robots with point feet. They propose the inclusion of an additional constraint that forces the planner to place the foot in a position where the terrain height is constant at a certain radius before and behind the considered foothold along the robot's direction of motion. A similar constraint could be adapted for hybrid driving-walking locomotion as well.

Lastly, the simulation results indicate that the proposed TO is a promising candidate for application on the real ANYmal robot, that has been shown to perform hybrid motions successfully in (Bjelonic et al., 2020). In addition, the extended WBC discussed in Section 3.4.2 has shown success in tracking dynamic trajectories on the real robot, as presented in (Medeiros et al., 2020). Further measures to ensure real-world applicability include the implementation of a feedback policy that corrects for tracking errors in the motion controller.

This thesis has presented different TO formulations that enable wheeled-legged robots to navigate in challenging terrain with dynamic motions by taking into account the height map of the terrain. The proposed formulations generate motion plans for increasingly complex scenarios, going from a simplified 2D approach to a more general hybrid motion planning framework.

Trajectory optimization for wheeled-legged robots is a challenging task, in which the interaction between the wheels and the base is a key aspect of the motion. For this reason, we employ a unified approach that optimizes over the whole-body motion and the interaction forces in a single planning problem. A decoupled strategy with an offline motion planner and an online terrain-aware motion controller allows for planning fast complex motions with minimal pre-defined restrictions on the base and the wheels' trajectories. In all formulations, a SRBD model is used to represent the robot dynamics and no quasi-static condition is assumed. This means that inertial effects from accelerated motions are not neglected and no assumption of low speed is made. The motion plans are verified in simulations and experiments with the ANYmal robot equipped with non-steerable torque-controllable wheels. For tracking the reference trajectories, the WBC developed in (Bjelonic et al., 2019) is extended to use the knowledge of the terrain normals for improving accuracy on the estimation of the wheels' contact points.

The first formulation focuses on planning driving motions for quadrupedal wheeled robots. In this case, common scenarios often include little variation in the lateral dimension. Such scenarios can benefit from a simplification of the planning problem. To this end, a TO formulation is presented for a simplified 2D model of the wheeled-legged robot that can overcome challenging terrain with dynamic driving motions at high speeds. The main advantage of this approach is its fast computation time, less than 35 ms/s of trajectory, which makes it suitable for an online implementation in its current form. Moreover, a detailed study on possible cost functions for the trajectory optimization confirmed that the solution for the feasibility problem is just as effective for motion planning and much faster than the proposed alternatives. In addition, we use the current reactive framework of ANYmal

with wheels (Bjelonic et al., 2019) as a baseline for performance comparison in the same testing scenarios, confirming the effectiveness of a terrain-aware approach to overcome more challenging terrains.

Next, a 3D TO formulation is developed still focusing on dynamic driving motions. This is a more general formulation that includes asymmetric terrains and it is able to generate trajectories with different behaviors for the left and right wheels. An additional rolling constraint was necessary to ensure consistency with driving locomotion. The 3D approach was tested in experimental tests with the real robot, proving to be robust enough to be tracked on the real system. The ANYmal robot is shown to overcome step-like obstacles up to 65° slope with an average speed of 0.5 m/s.

Lastly, a comprehensive hybrid motion planning framework is developed to fully exploit the dynamic capabilities of a wheeled-legged robot, enabling the combination of stepping and driving motions. This formulation extends the previous ones with the ability to negotiate discontinuous terrain types using dynamic hybrid motions. In this case, the TO formulation combines the set of constraints developed for purely driving motions with a phase-based parametrization for the wheels' trajectories (Winkler et al., 2018) that allows the solver to compute continuous motions with different contact states for the wheels. The physical feasibility of the trajectories is assessed in simulations with the ANYmal robot traversing gaps and steps. By using a physically realistic simulator that runs the exact same controller embedded on the real robot, the feasibility of the motions can be well assessed through simulations. This has been verified with the experiments for the driving motions.

Overall, perceptive motion planning for wheeled-legged robots in rough terrain was achieved by exploiting data from the environment to plan the robots' motion. Moreover, we show that blind locomotion could not overcome most of the testing scenarios or would traverse them in a static step-by-step fashion at an average speed much lower than 0.5 m/s. In contrast, our framework generates faster motions by using the map of the terrain to plan the trajectories before starting the motions.

In summary, the main contribution of this work is the development of a motion planning framework for wheeled-legged robots that optimizes over the base and wheels motions and the interaction forces in a single formulation, taking into account the dynamic model of the robot and the terrain information, which allows wheeled-legged robots to overcome a variety of challenging terrains with dynamic motions.

6.1 Future Work

The results of this thesis indicate several points for future improvement of this work. Firstly, neither of the proposed approaches take into account uncertainties in the robot model or the terrain map. There is no feedback policy that corrects for these effects during execution and as such the robot can end up not accurately tracking trajectories with longer time horizons. One way to overcome this limitation is an implementation of the motion planner in a receding horizon fashion, employing an MPC formulation. In this case, the motion plans are continuously updated using online measurements of the current state of the robot and the planned trajectories are tracked for as long as it takes to compute a new one. This would allow the online adaptation of the trajectories to unpredicted disturbances and model mismatches.

The main concern for enabling a future MPC implementation is reducing the computational cost for solving the NLP. For that, more sophisticated initialization strategies can be investigated based on learning methods (Melon et al., 2020). Another possibility is to leverage information from offline planned trajectories to use as an initial guess for the online planner, which would speed up the computation. Moreover, an adaptive grid sampling for the problem discretization can be developed for improving the efficiency of the algorithm and focusing the evaluation of the constraints to more complex parts of the terrain.

In addition, the trajectory optimization problem is formulated as an NLP with discretized variables and constraints. This approach is easier to implement and there are several efficient off-the-shelf numerical solvers available for solving large-scale NLP problems. However, the NLP formulation suffer from all typical issues of a gradient-based approach. They are very sensitive to changes in the initial guess and highly prone to local minima, which can also be improved by employing smarter initialization techniques for the motions.

The hybrid motion framework can be further improved to include the optimization of the gait timings, which would allow the solver to automatically discover the ideal gait for traversing the given terrain. With the current implementation, the user must provide the gait pattern to be used depending on the scenario. However, contact schedule optimization would considerably increase the computation cost of the framework.

Lastly, the next step of this work would be to integrate an online perception system with the motion planning framework, enabling complete autonomous navigation in unknown environments.

6.2

Publications

During the development of this work, the two following papers were published in international conferences:

Medeiros, V. S., D. G. G. Rosa, and M. A. Meggiolaro (2019). **Torque Optimization for Stability Control of Wheeled Vehicles in Rough Terrain**. In: XVIII International Symposium on Dynamic Problems of Mechanics (DINAME 2019).

Medeiros, V. S., M. Bjelonic, E. Jelavic, R. Siegwart, M. A. Meggiolaro, and M. Hutter (2019). **Trajectory Optimization for Wheeled Quadrupedal Robots Driving in Challenging Terrain**. In: 9th International Symposium on Adaptive Motion of Animals and Machines (AMAM 2019).

The following journal paper was published in the journal IEEE Robotics and Automation Letters (Qualis A1), regarding the 3D driving formulation:

Medeiros, V. S., E. Jelavic, M. Bjelonic, R. Siegwart, M. A. Meggiolaro, and M. Hutter (2020). **Trajectory Optimization for Wheeled-Legged Quadrupedal Robots Driving in Challenging Terrain**. In: IEEE Robotics and Automation Letters 5 (3), pp. 4172–4179.

Moreover, the following two papers are in the final stages of production for future journal submission:

Medeiros, V. S., E. Jelavic, M. Bjelonic, R. Siegwart, M. A. Meggiolaro, and M. Hutter. **Dynamic Driving for Wheeled-Legged Robots in Rough Terrain Using 2D Trajectory Optimization**. Future submission to *Journal of Intelligent & Robotics Systems* (Qualis A2).

Medeiros, V. S., E. Jelavic, M. Bjelonic, R. Siegwart, M. A. Meggiolaro, and M. Hutter. **Motion Planning for Hybrid Dynamic Locomotion of Wheeled-Legged Robots in Rough Terrain**. Future submission to *IEEE Robotics and Automation Letters* (Qualis A1).

Bibliography

- BAE, H.; LEE, I.; JUNG, T. ; OH, J.. **Walking-wheeling dual mode strategy for humanoid robot DRC-HUBO+**. In: 2016 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 1342–1348, Oct 2016.
- BAJRACHARYA, M.; MAIMONE, M. W. ; HELMICK, D.. **Autonomy for mars rovers: Past, present, and future**. *Computer*, 41(12):44–50, 2008.
- BELLEGARDA, G.; BYL, K.. **Trajectory optimization for a wheel-legged system for dynamic maneuvers that allow for wheel slip**. In: 2019 IEEE CONFERENCE ON DECISION AND CONTROL (CDC), 2019.
- DARIO BELLICOSO, C.; JENELTEN, F.; FANKHAUSER, P.; GEHRING, C.; HWANGBO, J. ; HUTTER, M.. **Dynamic locomotion and whole-body control for quadrupedal robots**. In: 2017 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 3359–3365, 2017.
- BELLICOSO, C. D.; JENELTEN, F.; GEHRING, C. ; HUTTER, M.. **Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots**. *IEEE Robotics and Automation Letters*, 3(3):2261–2268, July 2018.
- BERTSEKAS, D. P.. **Dynamic Programming and Optimal Control**, volumen I. Athena Scientific, Belmont, MA, USA, 3rd edition, 2005.
- BETTS, J. T.. **Practical Methods for Optimal Control and Estimation Using Nonlinear Programming**. Cambridge University Press, USA, 2nd edition, 2009.
- BJELONIC, M.; BELLICOSO, C. D.; DE VIRAGH, Y.; SAKO, D.; TRESOLDI, F. D.; JENELTEN, F. ; HUTTER, M.. **Keep rollin’—whole-body motion control and planning for wheeled quadrupedal robots**. *IEEE Robotics and Automation Letters*, 4(2):2116–2123, 2019.
- BJELONIC, M.; SANKAR, P. K.; BELLICOSO, C. D.; VALLERY, H. ; HUTTER, M.. **Rolling in the deep – hybrid locomotion for wheeled-legged**

- robots using online trajectory optimization. *IEEE Robotics and Automation Letters*, 5(2):3626–3633, 2020.
- BLOESCH, M.; HUTTER, M.; HOEPFLINGER, M.; LEUTENEGGER, S.; GEHRING, C.; REMY, C. D. ; SIEGWART, R.. **State estimation for legged robots - consistent fusion of leg kinematics and IMU**. In: *PROCEEDINGS OF ROBOTICS: SCIENCE AND SYSTEMS*, Sydney, Australia, July 2012.
- BOSTON DYNAMICS. **Handle, mobile box handling robots for logistics**. <<https://www.bostondynamics.com/handle>>, 2019. Accessed: September 2019.
- BOUTON, A.; GRAND, C. ; BENAMAR, F.. **Motion control of a compliant wheel-leg robot for rough terrain crossing**. In: *2016 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA)*, p. 2846–2851, 2016.
- BOUTON, A.; GRAND, C. ; BENAMAR, F.. **Obstacle negotiation learning for a compliant wheel-on-leg robot**. In: *2017 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA)*, p. 2420–2425, May 2017.
- BOUTON, A.; GRAND, C. ; BENAMAR, F.. **Design and control of a compliant wheel-on-leg rover which conforms to uneven terrain**. *IEEE/ASME Transactions on Mechatronics*, p. 1–1, 2020.
- BRATTA, A.; ORSOLINO, R.; FOCCHI, M.; BARASUOL, V.; MUSCOLO, G. G. ; SEMINI, C.. **On the hardware feasibility of nonlinear trajectory optimization for legged locomotion based on a simplified dynamics**. In: *2020 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA)*, 2020. volume in press.
- BRUZZONE, L.; QUAGLIA, G.. **Review article: locomotion systems for ground mobile robots in unstructured environments**. *Mechanical Sciences*, 3:49–62, July 2012.
- CARON, S.; PHAM, Q. ; NAKAMURA, Y.. **Zmp support areas for multi-contact mobility under frictional constraints**. *IEEE Transactions on Robotics*, 33(1):67–80, 2017.
- CORDES, F.; KIRCHNER, F. ; BABU, A.. **Design and field testing of a rover with an actively articulated suspension system in a mars analog terrain**. *Journal of Field Robotics*, 35(7):1149–1181, 2018.

- CURRIE, J.; WILSON, D. I.. **OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User**. In: Sahinidis, N.; Pinto, J., editors, *Foundations of Computer-Aided Process Operations*, Savannah, Georgia, USA, 8–11 January 2012.
- DEFENSE ADVANCED RESEARCH PROJECTS AGENCY. **Darpa robotics challenges**. <<https://archive.darpa.mil/roboticschallenge/>>, 2015. Accessed: April 2020.
- DAI, H.; VALENZUELA, A. ; TEDRAKE, R.. **Whole-body motion planning with centroidal dynamics and full kinematics**. In: 2014 IEEE-RAS INTERNATIONAL CONFERENCE ON HUMANOID ROBOTS, p. 295–302, Nov 2014.
- DIEHL, M.; BOCK, H.; DIEDAM, H. ; WIEBER, P.-B.. **Fast Direct Multiple Shooting Algorithms for Optimal Robot Control**, p. 65–93. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
- FANKHAUSER, P.; BLOESCH, M.; GEHRING, C.; HUTTER, M. ; SIEGWART, R.. **Robot-centric elevation mapping with uncertainty estimates**. In: INTERNATIONAL CONFERENCE ON CLIMBING AND WALKING ROBOTS (CLAWAR), 2014.
- FANKHAUSER, P.; HUTTER, M.. **A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation**. In: Koubaa, A., editor, *ROBOT OPERATING SYSTEM (ROS) – THE COMPLETE REFERENCE (VOLUME 1)*, chapter 5. Springer, 2016.
- FANKHAUSER, P.; BLOESCH, M. ; HUTTER, M.. **Probabilistic terrain mapping for mobile robots with uncertain localization**. *IEEE Robotics and Automation Letters*, 3(4):3019–3026, Oct 2018.
- FEATHERSTONE, R.. **Rigid Body Dynamics Algorithms**. Springer-Verlag, Berlin, Heidelberg, 2007.
- FREITAS, G.; GLEIZER, G.; LIZARRALDE, F.; HSU, L. ; DOS REIS, N. R. S.. **Kinematic reconfigurability control for an environmental mobile robot operating in the amazon rain forest**. *Journal of Field Robotics*, 27(2):197–216, 2010.
- GEHRING, C.; BELLICOSO, C. D.; BLOESCH, M.; SOMMER, H.; FANKHAUSER, P.; HUTTER, M. ; SIEGWART, R.. **Kindr library — kinematics and dynamics for robotics**, 2016. [Online] Available: https://docs.leggedrobotics.com/kindr/cheatsheet_latest.pdf.

- GEILINGER, M.; PORANNE, R.; DESAI, R.; THOMASZEWSKI, B. ; COROS, S.. **Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels.** In: on Graphics (TOG), A. T., editor, PROCEEDINGS OF ACM SIGGRAPH, volumen 37. ACM, August 2018.
- GEILINGER, M.; WINBERG, S. ; COROS, S.. **A computational framework for designing skilled legged-wheeled robots.** IEEE Robotics and Automation Letters, 5(2):3674–3681, 2020.
- GERTZ, M.; NOCEDAL, J. ; SARTENAR, A.. **A starting point strategy for nonlinear interior methods.** Applied Mathematics Letters, 17(8):945 – 952, 2004.
- GIFTTHALER, M.; FARSHIDIAN, F.; SANDY, T.; STADELMANN, L. ; BUCHLI, J.. **Efficient kinematic planning for mobile manipulators with non-holonomic constraints using optimal control.** In: 2017 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), p. 3411–3417, 2017.
- GIORDANO, P. R.; FUCHS, M.; ALBU-SCHAFFER, A. ; HIRZINGER, G.. **On the kinematic modeling and control of a mobile platform equipped with steering wheels and movable legs.** In: 2009 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, p. 4080–4087, 2009.
- GRAND, C.; AMAR, F.; PLUMET, F. ; BIDAUD, P.. **Stability control of a wheel-legged mini-rover.** In: 5TH INT. CONF. ON CLIMBING AND WALKING ROBOTS (CLAWAR'02), 01 2002.
- GRAND, C.; BENAMAR, F.; PLUMET, F. ; BIDAUD, P.. **Decoupled control of posture and trajectory of the hybrid wheel-legged robot hylos.** In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2004. PROCEEDINGS. ICRA '04. 2004, volumen 5, p. 5111–5116, 2004.
- GRAND, C.; BENAMAR, F. ; PLUMET, F.. **Motion kinematics analysis of wheeled–legged rover over 3d surface with posture adaptation.** Mechanism and Machine Theory, 45(3):477–495, 2010.
- HARGRAVES, C.; PARIS, S.. **Direct trajectory optimization using nonlinear programming and collocation.** Journal of Guidance, Control, and Dynamics, 10(4):338–342, jul 1987.

- HEBERT, P.; BAJRACHARYA, M.; MA, J.; HUDSON, N.; AYDEMIR, A.; REID, J. I.; BERGH, C.; BORDERS, J.; FROST, M.; HAGMAN, M.; LEICHTY, J.; BACKES, P.; KENNEDY, B.; KARPLUS, P.; SATZINGER, B. W.; BYL, K.; SHANKAR, K. ; BURDICK, J. W.. **Mobile manipulation and mobility as manipulation - design and algorithms of robosimian**. *Journal of Field Robotics*, 32:255–274, 2015.
- HORNUNG, A.; WURM, K. M.; BENNEWITZ, M.; STACHNISS, C. ; BURGARD, W.. **OctoMap: An efficient probabilistic 3D mapping framework based on octrees**. *Autonomous Robots*, 2013. [Online] Available: <http://octomap.github.com>.
- HUTTER, M.; GEHRING, C.; JUD, D.; LAUBER, A.; BELLICOSO, C. D.; TSOUNIS, V.; HWANGBO, J.; BODIE, K.; FANKHAUSER, P.; BLOESCH, M.; DIETHELM, R.; BACHMANN, S.; MELZER, A. ; HOEPFLINGER, M.. **Anymal - a highly mobile and dynamic quadrupedal robot**. In: 2016 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 38–44, Oct 2016.
- HUTTER, M.; SIEGWART, R.; STASTNY, T.; RUDIN, K. ; BLÖSCH, M.. **Robot dynamics lecture notes**, 2017. Available at: https://ethz.ch/content/dam/ethz/special-interest/mavt/robotics-n-intelligent-systems/rsl-dam/documents/RobotDynamics2017/RD_HS2017script.pdf. Accessed: April 2020.
- IAGNEMMA, K.; DUBOWSKY, S.. **Traction control of wheeled robotic vehicles in rough terrain with application to planetary rovers**. *International Journal of Robotics Research*, 23:1029–1040, 10 2004.
- IAGNEMMA, K. D.; RZEPNIEWSKI, A.; DUBOWSKY, S.; PIRJANIAN, P.; HUNTSBERGER, T. L. ; SCHENKER, P. S.. **Mobile robot kinematic reconfigurability for rough terrain**. In: PROC. SENSOR FUSION AND DECENTRALIZED CONTROL IN ROBOTIC SYSTEMS III, SPIE VOL. 4196, 2000.
- INOTSUME, H.; SUTOH, M.; NAGAOKA, K.; NAGATANI, K. ; YOSHIDA, K.. **Modeling, analysis, and control of an actively reconfigurable planetary rover for traversing slopes covered with loose soil**. *Journal of Field Robotics*, 30(6):875–896, 2013.
- JARRAULT, P.; GRAND, C. ; BIDAUD, P.. **Robust obstacle crossing of a wheel-legged mobile robot using minimax force distribution and**

- self-reconfiguration.** In: 2011 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, p. 2753–2758, Sep. 2011.
- JARRAULT, P.; GRAND, C.; AMAR, F. ; BIDAUD, P.. **Experimental evaluation of obstacle clearance by a hybrid wheel-legged robot.** In: PROC. 14TH INTERNATIONAL SYMPOSIUM ON EXPERIMENTAL ROBOTICS SPRINGER, 06 2014.
- JELAVIC, E.; HUTTER, M.. **Whole-body motion planning for walking excavators.** In: 2019 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 2292–2299, 2019.
- JIA, Y.; LUO, X.; HAN, B.; LIANG, G.; ZHAO, J. ; ZHAO, Y.. **Stability criterion for dynamic gaits of quadruped robot.** Applied Sciences, 8(12), 2018.
- JIANG, H.; XU, G.; ZENG, W.; GAO, F. ; CHONG, K.. **Lateral stability of a mobile robot utilizing an active adjustable suspension.** Applied Sciences, 9(20), 2019.
- KAJITA, S.; KANEHIRO, F.; KANEKO, K.; FUJIWARA, K.; HARADA, K.; YOKOI, K. ; HIRUKAWA, H.. **Biped walking pattern generation by using preview control of zero-moment point.** In: 2003 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (CAT. NO.03CH37422), volumen 2, p. 1620–1626 vol.2, 2003.
- KALAKRISHNAN, M.; BUCHLI, J.; PASTOR, P.; MISTRY, M. ; SCHAAL, S.. **Fast, robust quadruped locomotion over challenging terrain.** In: 2010 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, p. 2665–2670, 2010.
- KELLY, M.. **An introduction to trajectory optimization: How to do your own direct collocation.** SIAM Review, 59(4):849–904, 2017.
- KLAMT, T.; BEHNKE, S.. **Anytime hybrid driving-stepping locomotion planning.** In: 2017 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 4444–4451, Sep. 2017.
- KLAMT, T.; RODRIGUEZ, D.; SCHWARZ, M.; LENZ, C.; PAVLICHENKO, D.; DROESCHEL, D. ; BEHNKE, S.. **Supervised autonomous locomotion and manipulation for disaster response with a centaur-like robot.** In: 2018 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 1–8, Oct 2018.

- KLEMM, V.; MORRA, A.; SALZMANN, C.; TSCHOPP, F.; BODIE, K.; GULICH, L.; KÜNG, N.; MANNHART, D.; PFISTER, C.; VIERNEISEL, M.; WEBER, F.; DEUBER, R. ; SIEGWART, R.. **Ascento: A two-wheeled jumping robot**. In: 2019 INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), p. 7515–7521, May 2019.
- KOENIG, N.; HOWARD, A.. **Design and use paradigms for gazebo, an open-source multi-robot simulator**. In: 2004 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS) (IEEE CAT. NO.04CH37566), volumen 3, p. 2149–2154 vol.3, Sep. 2004.
- KUDRUSS, M.; NAVEAU, M.; STASSE, O.; MANSARD, N.; KIRCHES, C.; SOUERES, P. ; MOMBAUR, K.. **Optimal control for whole-body motion generation using center-of-mass dynamics for predefined multi-contact configurations**. In: 2015 IEEE-RAS 15TH INTERNATIONAL CONFERENCE ON HUMANOID ROBOTS (HUMANOIDS), p. 684–689, Nov 2015.
- LAURENZI, A.; HOFFMAN, E. M. ; TSAGARAKIS, N. G.. **Quadrupedal walking motion and footstep placement through linear model predictive control**. In: 2018 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 2267–2273, 2018.
- LAURIA, M.; PIGUET, Y. ; SIEGWART, R.. **Octopus - an autonomous wheeled climbing robot**. In: PROCEEDINGS OF THE FIFTH INTERNATIONAL CONFERENCE ON CLIMBING AND WALKING ROBOTS (CLAWAR), p. 315–322, 2002.
- MEDEIROS, V. S.; ROSA, D. G. G. ; MEGGIOLARO, M. A.. **Torque optimization for stability control of wheeled vehicles in rough terrain**. In: XVIII INTERNATIONAL SYMPOSIUM ON DYNAMIC PROBLEMS OF MECHANICS (DINAME 2019), 2019.
- MEDEIROS, V. S.; BJELONIC, M.; JELAVIC, E.; SIEGWART, R.; MEGGIOLARO, M. A. ; HUTTER, M.. **Trajectory optimization for wheeled quadrupedal robots driving in challenging terrain**. In: 9TH INTERNATIONAL SYMPOSIUM ON ADAPTIVE MOTION OF ANIMALS AND MACHINES (AMAM 2019), August 2019.
- MEDEIROS, V. S.; JELAVIC, E.; BJELONIC, M.; SIEGWART, R.; MEGGIOLARO, M. A. ; HUTTER, M.. **Trajectory optimization for wheeled-legged**

- quadrupedal robots driving in challenging terrain. *IEEE Robotics and Automation Letters*, 5(3):4172–4179, 2020.
- MEHROTRA, S.. **On the implementation of a primal-dual interior point method.** *SIAM Journal on Optimization*, 2:575–601, 11 1992.
- MELON, O.; GEISERT, M.; SUROVIK, D.; HAVOUTIS, I. ; FALLON, M.. **Reliable trajectories for dynamic quadrupeds using analytical costs and learned initializations.** In: 2020 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), 2020. volume in press.
- MISHKIN, A. H.; MORRISON, J. C.; NGUYEN, T. T.; STONE, H. W.; COOPER, B. K. ; WILCOX, B. H.. **Experiences with operations and autonomy of the mars pathfinder microrover.** In: 1998 IEEE AEROSPACE CONFERENCE PROCEEDINGS (CAT. NO.98TH8339), volumen 2, p. 337–351 vol.2, March 1998.
- NASA. **2020 mission perseverance rover.** <<https://mars.nasa.gov/mars2020/>>, 2020. Accessed: May 2020.
- NOCEDAL, J.; WRIGHT, S. J.. **Numerical Optimization.** Springer, New York, NY, USA, second edition, 2006.
- PAPADOPOULOS, E.; REY, D.. **The force-angle measure of tipover stability margin for mobile manipulators.** *Vehicle System Dynamics - VEH SYST DYN*, 33:29–48, 01 2000.
- POSA, M.; CANTU, C. ; TEDRAKE, R.. **A direct method for trajectory optimization of rigid bodies through contact.** *The International Journal of Robotics Research*, 33(1):69–81, 2014.
- REID, W.; PÉREZ-GRAU, F. J.; GÖKTOĞAN, A. H. ; SUKKARIEH, S.. **Actively articulated suspension for a wheel-on-leg rover operating on a martian analog surface.** In: 2016 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), p. 5596–5602, May 2016.
- RIGHETTI, L.; BUCHLI, J.; MISTRY, M. ; SCHAAL, S.. **Inverse dynamics control of floating-base robots with external constraints: A unified view.** In: 2011 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, p. 1085–1090, 2011.

- SMITH, R.. **Open dynamics engine**, 2008. [Online] Available: <http://www.ode.org/>.
- SATZINGER, B. W.; LAU, C.; BYL, M. ; BYL, K.. **Experimental results for dexterous quadruped locomotion planning with robosimian**. In: 2014 INTERNATIONAL SYMPOSIUM ON EXPERIMENTAL ROBOTICS (ISER 2014), 2014.
- SATZINGER, B. W.; LAU, C.; BYL, M. ; BYL, K.. **Tractable locomotion planning for robosimian**. *The International Journal of Robotics Research*, 34(13):1541–1558, 2015.
- SCHWARZ, M.; RODEHUTSKORS, T.; DROESCHEL, D.; BEUL, M.; SCHREIBER, M.; ARASLANOV, N.; IVANOV, I.; LENZ, C.; RAZLAW, J.; SCHÜLLER, S. ; ET AL.. **Nimbro rescue: Solving disaster-response tasks with the mobile manipulation robot momaro**. *Journal of Field Robotics*, 34(2):400–425, Nov 2016.
- SICILIANO, B.; KHATIB, O.. **Springer Handbook of Robotics**. Springer-Verlag, Berlin, Heidelberg, 2007.
- SICILIANO, B.; SCIAVICCO, L.; VILLANI, L. ; ORIOLO, G.. **Robotics: Modelling, Planning and Control**. Springer Publishing Company, Incorporated, 2010.
- SIEGWART, R.; NOURBAKHSI, I. R. ; SCARAMUZZA, D.. **Introduction to Autonomous Mobile Robots**. The MIT Press, 2nd edition, 2011.
- SMITH, J. A.; SHARF, I. ; TRENTINI, M.. **Paw: a hybrid wheeled-leg robot**. In: PROCEEDINGS 2006 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2006. ICRA 2006., p. 4043–4048, 2006.
- SUN, J.; YOU, Y.; ZHAO, X.; ADIWAHONO, A. H. ; CHEW, C. M.. **Towards more possibilities: Motion planning and control for hybrid locomotion of wheeled-legged robots**. *IEEE Robotics and Automation Letters*, 5(2):3723–3730, 2020.
- SUZUMURA, A.; FUJIMOTO, Y.. **High mobility control for a wheel-legged mobile robot based on resolved momentum control**. In: 2012 12TH IEEE INTERNATIONAL WORKSHOP ON ADVANCED MOTION CONTROL (AMC), p. 1–6, March 2012.

- TOTAL. **Argos challenge**. <<https://www.total.com/dossiers/argos-challenge-building-tomorrows-oil-and-gas-robot>>, 2014. Accessed: April 2020.
- TURKER, K.; SHARF, I. ; TRENTINI, M.. **Step negotiation with wheel traction: a strategy for a wheel-legged robot**. In: 2012 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, p. 1168–1174, May 2012.
- VUKOBRATOVIĆ, M.; BOROVIAC, B.. **Zero-moment point — thirty five years of its life**. International Journal of Humanoid Robotics, 01(01):157–173, 2004.
- WÄCHTER, A.; BIEGLER, L. T.. **On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming**. Mathematical Programming, 106(1):25–57, Mar 2006.
- WERMELINGER, M.; FANKHAUSER, P.; DIETHELM, R.; KRÜSI, P.; SIEGWART, R. ; HUTTER, M.. **Navigation planning for legged robots in challenging terrain**. In: 2016 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), p. 1184–1189, Oct 2016.
- WETTERGREEN, D.; MORELAND, S.; SKONIECZNY, K.; JONAK, D.; KOHANBASH, D. ; TEZA, J.. **Design and field experimentation of a prototype lunar prospector**. The International Journal of Robotics Research, 29(12):1550–1564, 2010.
- WINKLER, A. W.. **Ifopt - A modern, light-weight, Eigen-based C++ interface to Nonlinear Programming solvers Ipopt and Snopt.**, 2018.
- WINKLER, A. W.; BELLICOSO, C. D.; HUTTER, M. ; BUCHLI, J.. **Gait and trajectory optimization for legged systems through phase-based end-effector parameterization**. IEEE Robotics and Automation Letters, 3(3):1560–1567, July 2018.
- WONG, C. Y.; TURKER, K.; SHARF, I. ; BECKMAN, B.. **Posture Reconfiguration and Navigation Maneuvers on a Wheel-Legged Hydraulic Robot**, p. 215–228. Springer International Publishing, Cham, 2015.
- ZUCKER, M.; JOO, S.; GREY, M. X.; RASMUSSEN, C.; HUANG, E.; STILMAN, M. ; BOBICK, A.. **A general-purpose system for teleoperation of**

the drc-hubo humanoid robot. *Journal of Field Robotics*, 32(3):336–351, 2015.

DE VIRAGH, Y.; BJELONIC, M.; BELLICOSO, C. D.; JENELTEN, F. ; HUTTER, M.. Trajectory optimization for wheeled-legged quadrupedal robots using linearized zmp constraints. *IEEE Robotics and Automation Letters*, 4(2):1633–1640, April 2019.

A Hermite Parametrization

The TO problem is transcribed into a NLP problem, in which the state trajectory is represented using discrete variables and the constraints are enforced at specific points along the trajectory. The continuous time motions are then obtained by a interpolation with a cubic polynomial:

$$x(t) = a_0 + a_1t + a_2t^2 + a_3t^3 \quad (\text{A-1})$$

For which the velocity profile is given by:

$$\dot{x}(t) = a_1 + 2a_2t + 3a_3t^2 \quad (\text{A-2})$$

Since four coefficients are available, the cubic polynomial can be parametrized by the initial and final position values x_0 and x_1 , the initial and final derivatives \dot{x}_0 and \dot{x}_1 , and its time duration ΔT , as illustrated in Figure A.1.

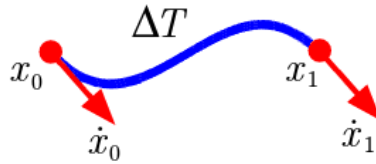


Figure A.1: Example of a cubic curve using Hermite parametrization.

Given these parameters, the determination of the trajectory is given by the solution of the following system of equations:

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

Solving the system in (A-3), the coefficients are given by:

$$\begin{aligned} a_0 &= x_0 \\ a_1 &= \dot{x}_0 \\ a_2 &= -\Delta T^{-2}[3(x_0 - x_1) + \Delta T(2\dot{x}_0 + \dot{x}_1)] \\ a_3 &= \Delta T^{-3}[2(x_0 - x_1) + \Delta T(\dot{x}_0 + \dot{x}_1)] \end{aligned} \quad (\text{A-4})$$