A whole-body predictive control approach to biped locomotion

Ewen Dantec

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Institut National des Sciences Appliquées de Toulouse

Présentée et soutenue par

Ewen DANTEC

Le 17 mai 2023

Application de méthodes de contrôle prédictif corps complet à la locomotion bipède

Ecole doctorale : **SYSTEMES**

Spécialité : **Robotique**

Unité de recherche :
LAAS - Laboratoire d'Analyse et d'Architecture des Systèmes

Thèse dirigée par
Nicolas MANSARD et Michel TAIX

Jury

M. Christian OTT, Rapporteur
M. Michele FOCCHI, Rapporteur
M. Shuuji KAJITA, Examinateur
Mme Angelika PEER, Examinatrice
M. Nicolas MANSARD, Directeur de thèse
M. Michel TAIX, Co-directeur de thèse
M. Philippe SOUERES, Président

"Who are you? Who slips into my robot body and whispers to my ghost? "

Mamoru Oshii, *Ghost in the Shell*, 1995

UNIVERSITÉ FÉDÉRALE TOULOUSE MIDI-PYRÉNÉES

Abstract

Université Fédérale Toulouse Midi-Pyrénées
Ecole Doctoral Systèmes

Doctor of Philosophy

A Whole-Body Predictive Control Approach to Biped Locomotion

by Ewen DANTEC

Humanoid robotics has been a very active field of research for the past decades, with important contributions in various scientific areas such as control engineering, biomechanics, computer science and mathematics. Nevertheless, performing reliable biped locomotion in generic environments still remains a challenge due to the real-time constraints and non-convexity of the problem. Because of previous technological limits, early works on walking robots have relied on template models and simplified dynamics. Given the steady increase in hardware computing capacities, complex control designs taking into account the whole-body dynamics of the system is becoming possible. On the other hand, predictive control algorithms based on trajectory optimization over a given preview window are proven to be a viable and robust solution for agile locomotion.

This thesis aims at implementing a whole-body predictive control framework for generic locomotion on real-world torque-controlled humanoid robots. Our controller was systematically tested on the torque-controlled robot Talos, a heavy humanoid platform with 32 actuated joints. Given the high complexity of the model, the computation frequency of our optimization solver cannot match the low-level control frequency of current robotics systems. To mitigate this issue, a first order feedback policy based on the solver sensitivities has been designed to approximate the high-level optimal command at the actuation frequency. In a second step, a 3-D walking controller for uneven terrain crossing is introduced and discussed. Two different heuristics were used to compute feet trajectories during locomotion: one based on pre-computed splines and one leveraging a height map of the environment that penalizes the flying foot velocity with respect to its height. The second heuristic allows to reduce the feet impedance and to perform push recovery in real time. Both heuristics have been combined with a high-level contact planner that generates optimal contact sequences in cluttered environments. Finally, to overcome the inherent non-convexity of planning scenarios with obstacles, a memory of motion was used to warm-start the solver and speed up its convergence.

Résumé en français

Depuis plusieurs décennies, la robotique humanoïde s'est révélée être un domaine de recherche très actif avec des contributions importantes dans divers domaines scientifiques tels que l'ingénierie de contrôle, la biomécanique, l'informatique et les mathématiques. Néanmoins, parvenir à générer une locomotion bipède fiable dans des environnements génériques reste un défi en raison des contraintes temps réel du système et de la non-convexité du problème. A cause des limites technologiques présentes il y a quarante ans, les premiers travaux sur les robots marcheurs se sont appuyés sur des modèles et des dynamiques simplifiées. Compte tenu de l'augmentation constante des capacités de calcul de nos ordinateurs, des schémas de contrôle plus complexes tenant compte de la dynamique du corps complet deviennent possibles. D'autre part, les algorithmes de contrôle prédictif basés sur l'optimisation de la trajectoire future s'imposent de plus en plus comme une option viable et robuste pour la locomotion agile.

Cette thèse vise à mettre en oeuvre une approche corps complet de la locomotion bipède à travers le prisme des méthodes de contrôle predictif. L'approche a été implémentée sur le robot Talos, un humanoïde lourd contrôlé en couple et possédant 32 joints actionnés. Compte tenu de la grande complexité du modèle, la fréquence de recalcul de notre solveur optimal est trop faible par rapport à celle du contrôle de bas niveau des plateformes robotiques actuelles. Pour atténuer ce problème, une politique de rétroaction de premier ordre basée sur les sensibilités du solveur a été conçue afin d'approximer la commande optimale à la fréquence des actionneurs. Dans un deuxième temps, un contrôleur de marche adapté à la locomotion en terrain accidenté est introduit puis discuté. Deux heuristiques différentes ont été utilisées pour calculer les trajectoires des pieds pendant la marche: la première est basée sur des splines pré-définies, la seconde utilise une carte de hauteur de l'environnement qui pénalise la vitesse du pied en vol par rapport à sa hauteur. La seconde heuristique permet de réduire l'impédance des pieds et d'effectuer des mouvements de rééquilibre après poussée en temps réel. Les deux heuristiques ont été combinées avec un planificateur de contact de haut niveau capable de définir des séquences de contact optimaux dans des environnements encombrés. Enfin, pour surmonter la non-convexité inhérente aux scénarios de planification comportant des obstacles, une mémoire du mouvement a été implémenter pour initialiser le solveur et accélérer sa convergence.

Acknowledgements

First of all, I would like to thank M. Focchi and C. Ott for reviewing this thesis, as well as the jury members P. Souères, A. Peer and S. Kajita. My deepest thanks go to N. Mansard and M. Taïx who have been my thesis directors during the past three years. I could not have gone all this way without their advices and support.

I gratefully acknowledge the researchers with whom I collaborated, and in particular T. Lembono for his work on memory of motion, S. Kleff for his insight on force feedback techniques combined with predictive control, O. Stasse for his precious advices on real-time operability and ROS architecture. The experimental work achieved in this thesis would not have been possible without the help of M. Naveau and P. Fernbach from Toward, who provided me with essential insights on torque control and locomotion theory. The software packaging of this thesis would have been an incredible mess without the assistance of G. Saurel, our software engineer. Additionally, I wish to thank A. Roig from PAL Robotics for laying the basis of my whole-body control framework and for his help on the robot.

I am indebted to all the members of the Gepetto team who have contributed to make the laboratory an interesting and enjoyable workplace. My thanks also go to all my loyal roommates who supported me during tough times, and to my cat Lectra, who helped me cope with work overload by forcing me to play with her.

Contents

List of Figures

List of Tables

List of Abbreviations

CoM	**C**enter **o**f **M**ass
CoP	**C**enter **o**f **P**ressure
CP	**C**apture **P**oint
CWC	**C**onvex **W**rench **C**one
DCM	**D**ivergent **D**omponent of **M**otion
DDP	**D**ifferential **D**ynamics **P**rogramming
DoF	**D**egree **o**f **F**reedom
GPR	**G**aussian **P**rocess **R**egression
ID	**I**nverse **D**ynamics
IK	**I**nverse **K**inematics
iLQG	**i**terative **L**inear **Q**uadratic **G**aussian
iLQR	**i**terative **L**inear **Q**uadratic **R**egulator
KKT	**K**arush-**K**uhn-**T**ucker
LIP	**L**inear **I**nverted **P**endulum
MIP	**M**ixed **I**nteger **P**rogram
MPC	**M**odel **P**redictive **C**ontrol
NLP	**N**on-**L**inear **P**roblem
NMPC	**N**onlinear **M**odel **P**redictive **C**ontrol
NN	**N**earest **N**eighbor
OCP	**O**ptimal **C**ontrol **P**roblem
QP	**Q**uadratic **P**rogramming
RL	**R**einforcement **L**earning
RRT	**R**apidly-exploring **R**andom **T**ree
$\mathbf{SE}(3)$	**S**pecial **E**uclidean Group
$\mathbf{se}(3)$	Lie Algebra of $\mathbf{SE}(3)$
SLQ	**S**equential **L**inear **Q**uadratic
$\mathbf{SO}(3)$	**S**pecial **O**rthogonal Group
$\mathbf{so}(3)$	Lie Algebra of $\mathbf{SO}(3)$
SQP	**S**equential **Q**uadratic **P**rogramming
SRB	**S**ingle **R**igid **B**ody
ZMP	**Z**ero **M**oment **P**oint

Glossary

$\boldsymbol{b}$	Generalized nonlinear forces
$\boldsymbol{c}$	CoM position
f	Dynamics of the system
$\boldsymbol{f}$	Force variable
$\boldsymbol{f}_t$	Gaps in the dynamics
$\boldsymbol{g}$	Gravity vector
H	Hamiltonian of the dynamic system
$\boldsymbol{J}_i$	Contact Jacobian
$\boldsymbol{k}$	Feedforward term of the Riccati policy
$\boldsymbol{K}$	Feedback term of the Riccati policy
ℓ_t	Cost function at node t
L	Half length of foot
$\boldsymbol{L}$	Angular momentum
$\mathcal{L}$	Langrangian of the dynamics
m	Mass of the system
$\boldsymbol{M}$	Joint-space inertia matrix
$\boldsymbol{p}$	Placement variable
q_t	Q-value of the policy at node t
$\boldsymbol{r}$	Translation vector
$\boldsymbol{S}$	Actuation matrix
T	Number of nodes in the discrete OCP
$\mathcal{T}$	Horizon length of the continuous OCP
$\boldsymbol{u}$	Whole-body control variable
$\underline{\boldsymbol{u}}$	Whole-body control trajectory
$\boldsymbol{\Delta u}$	Whole-body control variable perturbation
v	Value function
W	Half width of foot
$\underline{\boldsymbol{x}}$	Whole-body state trajectory
$\boldsymbol{x}$	Whole-body state variable
$\boldsymbol{\Delta x}$	Whole-body state variable perturbation
$\boldsymbol{z}$	Concatenated state and control
π	Control policy
$\boldsymbol{\lambda}$	Co-state variable or wrench variable
μ	Friction coefficient
$\boldsymbol{\tau}$	Torque variable

Chapter 1

Introduction

Robotics is the art of animating articulated machines. This art stands at the crossroad of numerous fields, from mechatronics to automation, including mathematics, neuroscience, computer science and biomechanics. At the heart of robotics is the wish to replicate human action with always more autonomy, so as to substitute human operators in hazardous or unpleasant work environments. Conversely, studying the way robots move can provide precious insights on the comprehension of the biological principles of locomotion in humans and animals. In a sense, robotics deals with the dialectic between executing an action and understanding a movement [Laumond 2013].

An action is primarily defined by its end goal (what to do), whereas a movement describes the series of motor commands and joint displacements producing the relevant action (how to do). When translating action into movement, three challenges arise. Suppose that a human subject is asked to drink a glass of water on a table. Whether the human would be using their left hand, right hand or both to drink the glass of water is of no importance from an operator viewpoint as long as the task is completed. This highlights the first challenge in robotics: the operational space of the task is generally quite big, meaning that there exist numerous solutions, or numerous sets of movements, to perform one given action.

For a healthy human subject, drinking a glass of water on a table poses no particular difficulty. However, explaining how they did it, what precise muscles they used, what nervous impulses they sent, is astonishingly hard. Humans, as all living beings, seem to perceive the world in terms of actions and results more than movements and executions [Smyth and Wing 2013]. The incredible complexity of even the simplest motion is hidden from us, and as a consequence we struggle to tell a machine how exactly it must perform a given task. Transcribing the exact trajectory of every limb of the body so as to achieve a given motion is a painful problem that has been first tackled by dancers who wished to record their performances on paper (using, for example, the Labanotation devised by Rudolf Laban [Knust 1978], see Fig. 1.1). This inherent complexity can be seen as the second challenge in robotics: since we do not fully understand the fundamental principles of biological motion, we struggle to find a right mathematical language to describe it.

Suppose now that an obstacle is blocking the way to the glass, and that the human subject is again asked to grasp it. If additionally the obstacle is moving on its own, the subject has to continuously correct their motion based on their perception of the obstacle in order to execute the task. This ability to adapt the command to

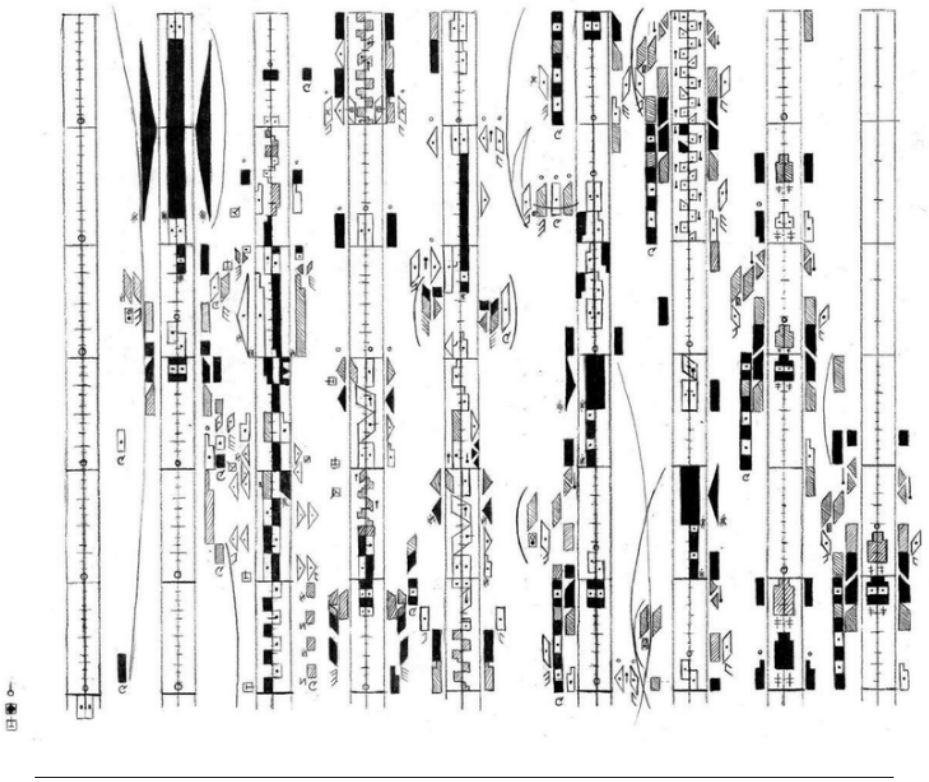

FIGURE 1.1: An example of Labanotation to transcribe dance motion [Kleida 2018].

new perceptive inputs is a key feature of autonomy in real-world robotics. Mimicking this ability requires to solve very challenging problems each time fresh data about environmental changes are processed, and to solve it fast enough so that the corresponding response is still relevant with respect to the current world configuration. Here lies the third challenge in robotics.

The experiment of picking a glass of water illustrates how compromised it seems to reconcile the viewpoint of the neurobiologist and the roboticist. Humans do not directly deal with command in motor space, nor do they produce the same exact motion when asked to perform a task two times in a row, but still appear to obey some principle of optimality [Todorov and Jordan 2002], even if it is unclear which criterion is minimized along the trajectory. On the other hand, the roboticist seeks to reproduce the efficiency, adaptability and robustness of human motion through mathematical transcription, while the scientific community is not even sure how to define these features in the first place. Moreover, physical differences between humans and robots, as well as computational capacities, also prevent complex motions from being reproduced on artificial platforms.

Robotics used to be divided between two schools of thought, one favouring a data-driven approach [Brooks 1991], the other relying on trajectory optimization [Bobrow 1982; Chernousko 1994]. Data-driven methods take their inspiration from biology and the structure of neural cortex. Through trial and error, they aim at encoding the primary components of motion into a statistical or learning process. Trajectory-based methods, on the other hand, attempt to capture the complexity of reality inside simpler mathematical structures and solve the problem via classical

FIGURE 1.2: From left to right: Opportunity (2004-2018), Curiosity (2012-present) and Perseverance (2021-present).

optimization tools. Both approaches have their disadvantages: data-driven algorithms often give imprecise and rough results at the cost of never-ending training hours, while trajectory-based algorithms feature high computational load and struggle to discover complex motions based on optimization alone.

The latest developments in robotics tend to blur the lines between these two approaches. More and more researchers acknowledge that optimization and learning are, in essence, complementary to each other. Trajectory-based methods can for example be used to guide the learning process or refine stochastic policies to make them dynamically feasible. As often in science, we pierce through by combining the best of both worlds.

1.1 Motivation

Robotics is already part of our daily lives. The first industrial robot, Unimate, was built more than 60 years ago, in the continuity of teleoperation research for nuclear application. Nowadays, most, if all, industrial processes involve the use of some kind of robots: articulated arms, conveyor belts, and so on. Since 2015, the robot Pepper has played the role of a receptionist in multiple offices and shops across Europe and Japan. The automatic vacuum cleaner Roomba has become a mainstream household item and has already reached its ninth generation. Thanks to the development of teleoperation technology, remote surgery [Wall and Marescaux 2013] is revolutionizing the field of medical care by connecting patients and surgeons across the globe. Finally, space rovers such as Curiosity or Perseverance have become a key technology for the exploration of foreign celestial bodies, demonstrating that well-engineered robotic platforms can stay operational way past their initial mission time even in harsh environments. Curiosity, for example, is still working fine more than ten years after its landing, despite being initially programmed for a two-year mission; its predecessor Opportunity had held for 14 years while relying solely on solar energy to replenish its batteries.

As for humanoid robotics, the first walking biped platform, WABOT-1, was built by Waseda University in 1972. Nearly thirty years later, the robot Asimo developed by Honda displayed very robust walking locomotion coupled with autonomous navigation and pattern recognition skills. In the 2015 DARPA competition, the robot Atlas from Boston Dynamics completed many complex tasks such as opening a door, climbing a ladder, manipulating tools or driving a vehicle. A few years later, the same robot performed dynamic multi-contact motions such as handstands, backflips, jumps over obstacles, just to name a few.

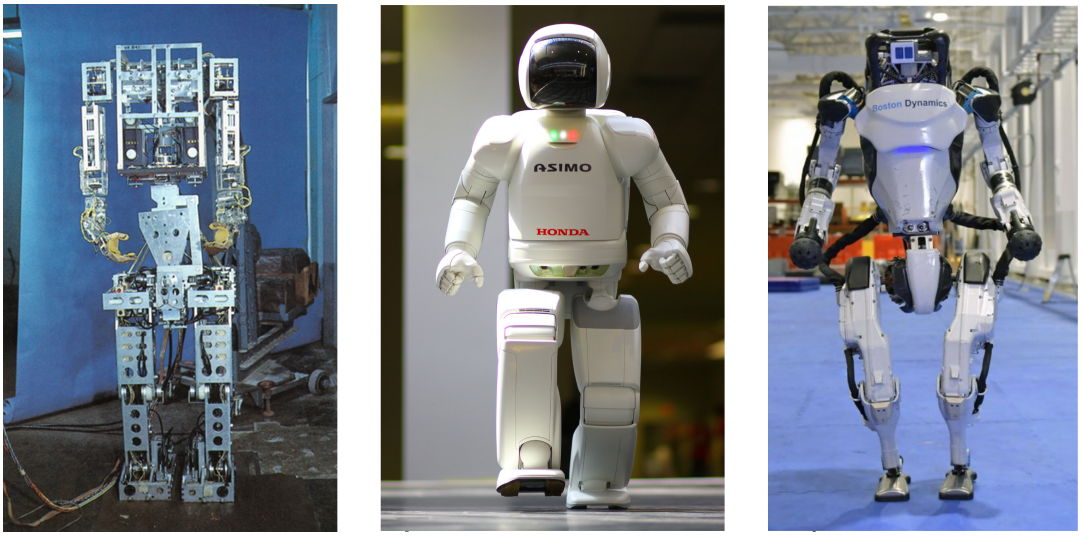

FIGURE 1.3: From left to right: WABOT-1, Asimo, Atlas. These humanoid robots have contributed to advance further the field of legged robotics.

If humanoid robots are currently so efficient that they can reproduce human gymnastics, one could wonder why they have not become a common sight in our everyday life. The answer lies in the lack of robustness, compliance, and replanning capabilities of current platforms [IEEE Spectrum 2015]. Moving in a dynamic and uncertain environment requires autonomy, compliance, robustness and adaptability, each of these being quite hard to describe mathematically. Autonomy could be seen as the ability to handle high-level orders without being given additional information. Compliance describes the tolerance to deviations from a reference equilibrium: this is an essential feature to deal smoothly with unplanned impact events or grasping tasks while ensuring stability. Robustness and adaptability both involve being able to perform well even in non-nominal conditions and adapt quickly to environmental changes.

As compared to other legged systems such as quadrupeds, bipeds are intrinsically unstable, similar to a mass attached to a vertical stick. The most impressive demonstrations on biped robots still occur inside the controlled environment of a laboratory, after countless hours of offline trajectory simulations. Additionally, human-sized robots feature incredible mechanical and electrical complexity which make them prone to hardware failures. Some of them, such as the Atlas robot, are powered through hydraulic actuation, a technology that can generate high and sharp torque commands but is quite difficult to maintain because of the significant fluid pressure in its transmission circuit [Lam 2013].

If imitating the human self remains so complicated, why do we bother to try? In a world designed for human beings, the anthropomorphic form is still the logical choice for a robot that needs to move autonomously in its environment. Climbing stairs or ladders, sitting in an armchair, driving, moving a box on a shelf, opening a trap door... All these actions require two legs and two arms to be performed nominally. Moreover, a robot similar to humans can, in theory, reproduces the entire range of actions of anthropomorphic beings, fueling countless potential applications. To cite some examples, humanoid robots have been proposed to intervene in complex industrial context such as aircraft manufacturing [Kheddar et al. 2019]. Due to the

current global population increase, some countries are studying the option of delegating elderly care to humanoid robots [Čaić et al. 2018]; if this kind of services were to be provided at home, the robot should be expected to move in the safest possible way across an environment designed for humans. Another possible application lies in the Urban Search and Rescue (USAR) program launch in 2004 by the Department of Homeland Security of the United States [Messina 2006]. This program aims at developing rescue robots able to locate, help and extricate entrapped victims during disaster events. When the environment is highly undefined, cluttered and hazardous, robots need human-like adaptability to achieve their mission. Lastly, the use of humanoid robots for space exploration has been examined. In 2011, Robonaut 2 [Diftler et al. 2012] boarded the International Space Station and performed a wide range of dexterous manipulation tasks, demonstrating that general maintenance activities in space can be covered by semi-autonomous robotics. Later in 2017, NASA organized the Space Robotics Challenge, a programming competition in which the candidates were asked to complete simple tasks in simulation with the humanoid robot Valkyrie. Inspired from the movie the Martian, the Space Robotics Challenge aims at assessing the advantages of using humanoids to establish colonies on extra-terrestrial objects like Mars or the Moon.

In my personal opinion, robots are one of the most fascinating vector of human curiosity, in the sense that they give us the capacity to reach and explore uncharted territories where we cannot yet survive. Because they stand as the embodiement of many scientific fields, they help us to push even further the boundaries of human knowledge and capabilities.

1.2 Problem Statement

Despite very promising researches and exciting applications, humanoid robots are still far from being able to wander freely into the wild. In order to truly become autonomous and reliable platforms able to move like we do, the locomotion problem should be thoroughly formulated as a synthesis between the high-level motion planning and low-level control scheme. Most frameworks in the literature of optimal control treat the system as a reduced model to alleviate computation load, thus limiting the intrinsic performances of the controller. They also tend to distribute the complexity of the optimization between a reduced model planning block and whole-body instantaneous control block, sometimes connected to a stabilizer. This multi-block scheme can be tedious to work with as all parts of the framework needs to be synchronised between them.

This thesis seeks to reformulate the locomotion problem as a whole-body model-based optimization problem where every relevant variable is refined along a predicted horizon according to some optimal criterion. The end goal is to build a generalized predictive controller for multi-contact motion generation and execution, able to leverage the whole dynamics of the system in the preview horizon. To augment the exploration capacities of this optimization scheme and overcome the non-convexity of the problem, we also investigate the option of connecting the controller to a data-based library of motion. By disserting on the state of the art, we will more accurately refine this objective, justify its importance and explain what are the missing contributions to reach it. We will then formulate in details the contributions of this thesis based on the current progresses in optimal control and robotic locomotion.

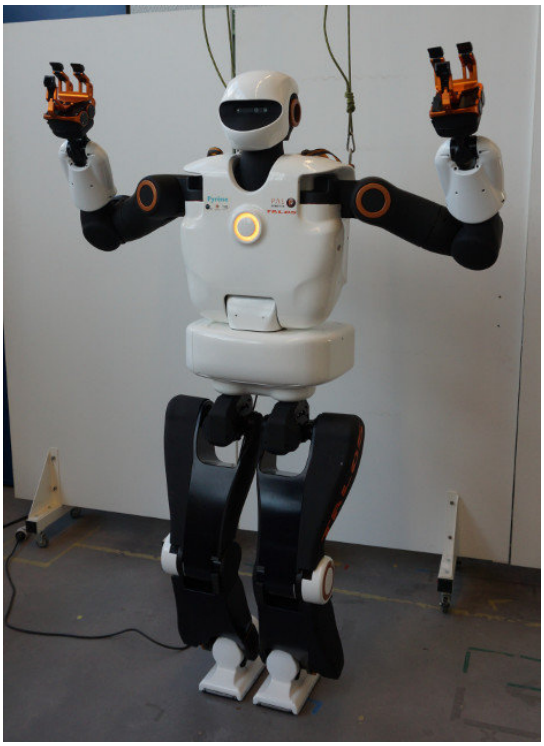

FIGURE 1.4: The humanoid robot Pyrène, first prototype of the Talos family that we used for most experiments.

The algorithms developed in this thesis have been deployed on the humanoid robot Pyrène [Stasse et al. 2017], first prototype of the Talos series built by Pal Robotics. Pyrène is a powerful 95 kg robotic platform able to perform manipulation tasks as well as locomotion on uneven terrains. It is equipped with electrical motors, an IMU and two onboard computers (dual i7 CPU at 2.8 GHz). Its kinematic tree is composed of 32 actuated joints mounted with torque sensors, except for the wrist joints and head joints. Additionally, force sensors are integrated inside the wrist and ankle frames. These sensors are a key component to perform torque control on the robot and make its motions naturally compliant, as opposed to classical position control schemes that manually encode the compliance inside the impedance gains. The low-level control framework of Pyrène works at a frequency of 2 kHz, which means that the robot is expected to receive a torque command every 0.5 ms. According to Pal Robotics benchmarks, the bandwidth of the torque tracking scheme of Pyrène is 5 Hz.

Although being a prototype, Pyrène is a high-performance scientific platform benefiting from state-of-the-art technology. The predictive algorithms that we seek to write are especially relevant to control this kind of robots with heavy limbs and high inertia.

This thesis is part of the Memmo project, a European scientific collaboration that aims at developing generic algorithms for complex motion generation in real time. In the frame of this project, the robot Pyrène has been used as an industrial demonstrator for aircraft manufacturing in an Airbus factory.

1.3 Thesis Outline

This thesis is organized as followed. Chapter II introduces the current state of the art and related works, starting from the problem of trajectory optimization and reactive control, then detailing recent progresses on online predictive control and data-driven methods.

Chapter III outlines the numerical framework used to solve our full dynamics optimal control problem. A Differential Dynamics Programming approach is presented and briefly commented.

Chapter IV addresses the issue of minimizing the impact of computation load by approximating the optimal control at a higher frequency with a local feedback policy. It also unveils the first demonstrations of our control scheme on the torque-controlled robot Talos.

Chapter V makes use of the tools previously introduced to perform whole-body locomotion in flat and non-flat environments with the Talos humanoid.

Chapter VI discusses the idea of increasing the compliance and adaptability of the locomotion scheme by removing the user-defined feet reference and letting the solver choose where to land its feet. Additionally, it provides a method to deal with uneven terrain based on a velocity height map combined with a high-level footstep planner.

Chapter VII introduces the combination of data-based methods and MPC to tackle the issue of non-convexity in the environment, through the use of a memory of motion. The demonstration provided here, although simple, is intended to be a proof of concept and can be easily generalized to more complex multi-contact motions.

1.4 Contributions

The following papers have been published during the course of this thesis:

Journals

- **E. Dantec**, M. Taïx and N. Mansard, *First Order Approximation of Model Predictive Control Solutions for High Frequency Feedback*, in IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 4448-4455, April 2022.

Conferences

- R. Budhiraja, A. Parag, **E. Dantec**, J. Carpentier, C. Mastalli and N. Mansard, *Crocoddyl: Fast computation, Efficient solvers,Receding horizon and Learning*, Journées Nationales de la Robotique Humanoïde, May 2020, Paris, France.
- **E. Dantec**, R. Budhiraja, A. Roig, T. Lembono, G. Saurel, O. Stasse, P. Fernbach, S. Tonneau, S. Vijayakumar, S. Calinon, M. Taïx and N. Mansard, *Whole*

Body Model Predictive Control with a Memory of Motion: Experiments on a Torque-Controlled Talos, IEEE International Conference on Robotics and Automation (ICRA), May 2021, Xi'an, China.

- S. Kleff, **E. Dantec**, G. Saurel, N. Mansard and L. Righetti, *Introducing Force Feedback in Model Predictive Control*, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nov 2022, Kyoto, Japan.
- **E. Dantec**, M. Naveau, P. Fernbach, N. Villa, G. Saurel, O. Stasse, M. Taïx and N. Mansard, *Whole-Body Model Predictive Control for Biped Locomotion on a Torque-Controlled Humanoid Robot*, IEEE-RAS International Conference on Humanoid Robots (Humanoids), Nov 2022, Ginowan, Japan.
- N. Villa, P. Fernbach, M. Naveau, G. Saurel, **E. Dantec**, N. Mansard and O. Stasse, *Torque Controlled Locomotion of a Biped Robot with Link Flexibility*, IEEE-RAS International Conference on Humanoid Robots (Humanoids), Nov 2022, Ginowan, Japan.

Chapter 2

State of the Art

2.1 Motion generation for walking robots

2.1.1 A brief history of planning

Let us consider an articulated robot whose task is to cross a cluttered environment in order to reach a given pose. Motion planning in robotics consists in finding a feasible path starting from its current pose and ending with the desired pose, with the constraint of avoiding collisions along the way (collisions with external obstacles and auto-collisions). Rather than considering the problem in the 6-dimensional operational space of translation and rotation, roboticists solve it in the configuration space of the robot [Lozano-Perez 1983], that is, the concatenation of all joint positions of the system. Thus, finding a feasible path for an articulated end effector in a 6-dimensional space amounts to finding a feasible path for a point in the configuration space. A path parametrized by time is called a trajectory, that is, a function that takes a time variable and returns an element of the state space. Usually this state space is composed of the configuration space extended by its tangent manifolds, representing velocity and acceleration. Additionally, a control space can be introduced to define how the user can affect the evolution of the current state. Trajectory generation aims at computing a set of state and control points parametrized by time such that the desired pose or configuration is reached at the end of the trajectory.

Let us go back to the geometric motion planning problem, also known as the piano mover problem (see Fig. 2.1). It is famous for being computationally intractable, even if the mathematics community proved that it was decidable 40 years ago [Schwartz and Sharir 1983]. In fact, it has been proven that these problems are often NP-complete or even PSPACE-complete [Hartline and Libeskind-Hadas 2003], a class of problems believed to be even harder than NP-complete. Deterministic methods [Canny 1988], which attempt to build an exact representation of the configuration space in order to extract connectivity properties, feature exponential running time at best. Given that a humanoid robot has an average of 30 articulations, such computational load makes the algorithm unusable in practice. Furthermore, only the geometry of the problem is tackled by this planning formulation, whereas kinematic and dynamic aspects also need to be taken into account to control a robot.

Local methods based on potential fields have thus been developed to solve for

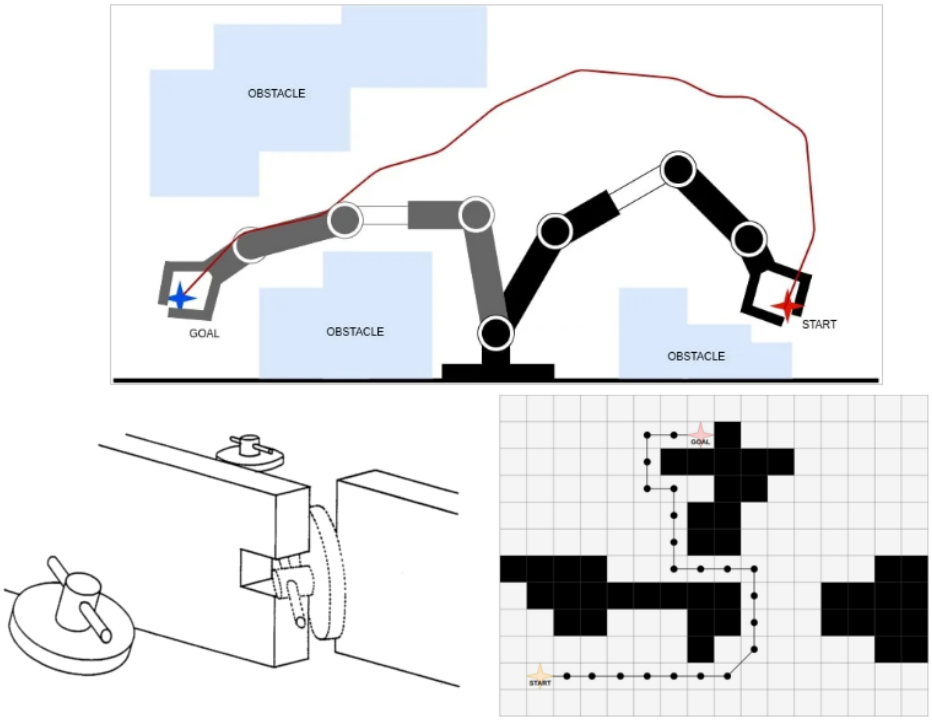

FIGURE 2.1: Illustration of several classical planning problems encountered in robotics.

practical cases [Khatib 1986]. These methods find a collision-free trajectory in real-time based on a simple gradient descent idea: the goal is an attracting point of the system while obstacles act as repulsive walls. This approach does not prevent the robot to get stuck in local minima, a situation that can be met in any maze-like environment. Stochastic methods based on a clever sampling of the configuration space quickly arose to tackle the issue of locality. Techniques such as probabilistic roadmaps [Kavraki et al. 1996; Amato and Wu 1996] and randomized potential field [Barraquand and Latombe 1991] were proposed as a way to efficiently solve the planning problem. The first approach consists in building a graph in the collision-free configuration space and connecting pairs of feasible points; the second extends [Khatib 1986] by introducing random configuration sampling to escape from local minima. However, these methods tend to struggle when the configuration space becomes large, as they heavily rely on blind exploration of the solution set. Researchers have thus proposed the Rapidly-exploring Random Tree (RRT) method [LaValle 1998]: the idea is to incrementally build a space-filling tree by sampling vertices in the search space, and bias the growth towards the largest uncharted areas, or regions of most interest. Under a set of mild conditions, it was proven that the RRT algorithm converges almost surely toward a non-optimal solution [Karaman and Frazzoli 2011]; in this work, a variant of the method, called RRT*, was proposed for optimal motion planning applications. During the last two decades, numerous extensions of RRT have been introduced, such as real-time RRT* [Naderi et al. 2015] for video game applications, closed-loop RRT [Kuwata et al. 2009] for autonomous driving, kino-dynamic RRT* [Webb and Berg 2013] to efficiently handle controllable linear dynamics, LQR-RRT* Perez et al. 2012 to find optimal paths for complex systems with underactuated dynamics, and so on. These methods are

particularly efficient for kino-dynamics trajectory planning [Donald et al. 1993], in which the aim is to generate a trajectory with bounds on velocity, acceleration and torque, in addition to kinematic constraints. This special class of planning problems lies at the heart of modern-day robotics, as bounds on joints and commands are essential to protect the expensive actuation of state-of-the-art articulated systems.

Classical planning methods tackle efficiently the geometrical aspects of the locomotion problem, but fail to account for the intrinsic physics of the movement. Besides, they cannot predict what the state of the system is going to be in the near future. In order to design even more efficient walking algorithms, physics needs to be considered, in particular the relationship between contact forces and acceleration.

2.1.2 Specificities of locomotion planning

Legged robotics is characterized by a set of several specific features: under-actuation, contact interaction, balance and optimality.

Under-actuation

A walking system is by design under-actuated, meaning that none of its actuators can directly act on its Center of Mass (CoM) position in space. If the system has enough degrees of freedom, it can control its angular momentum but is still unable to act on its linear momentum without relying on external forces like contact interaction or gravity [Wieber 2006a]. The non-holonomic aspect of angular dynamics described in [Wieber 2006a] is typically leveraged in the field of space robotics to control manipulators in weightless environment [Papadopoulos 1993; De Luca and Oriolo 1997].

Contact interaction

A consequence of under-actuation is that legged robots must apply forces to their environment in order to move. According to the third law of Newton, the environment is then exerting an opposite force on the body, resulting in a motion in the 3-dimensional operational space. This contact interaction plays a key role in the locomotion of all terrestrial beings, as it dictates how they can move and balance themselves. In most cases, the contact forces are unilateral, or in other words: the articulated system can only push against the ground, not pull it. Additionally, friction bounds limit the way the robot can exert forces on its environment without slipping.

Balance constraint

Locomotion cannot be performed without proper balance over limbs in contact. In some cases, falling over results in significant hardware damage to the robotic platform. Similarly to an inverted pendulum, biped robots are naturally unstable, which means that the walking algorithm must actively prevent any loss of balance. Balance can be ensured by commanding the CoM to remain inside the support polygon of the limbs in contact, but this leads to very slow and conservative motions, during which the system is in static equilibrium at any instant. On the other hand, dynamic tasks like jumping and running can only be executed if the robot temporarily goes out of static equilibrium. It is then essential to predict the system dynamics in order

to ensure that static equilibrium can be reached again in the near future.

Optimality criterion

An essential feature of legged locomotion, and more generally of motion generation, is that living beings perform movements in an efficient fashion, or to put it another way: they tend to optimize their actions with respect to some given performance criterion. Several optimization models have been proposed to explain the underlying principles of human motion: some postulate that we minimize the commanded torque change [Flash et al. 2013] or energy consumption [Saibene 1990]; others argue that hand movement follows the 2/3 power law [Plamondon and Guerfali 1998] which links the curvature of a trajectory to the angular velocity of the body; lately, the isochrony phenomenon, i.e. the relationship between speed of arm and trajectory distance has been considered in motion planning [Yokoyama et al. 2018].

All in all, biped locomotion represents a very difficult problem involving kinematic and dynamic considerations, non-convexity, nonlinearity, unilateral contacts and non-holonomic constraints [Wieber 2006a].

2.1.3 First approaches to the locomotion problem

The complexity of walking physics is such that it could not be tackled by premodern computers. As a workaround, researchers have first attempted to solve the locomotion problem by reducing the movement to its most simple components.

Central Pattern Generators

Preliminary results on walking motion generation for legged systems heavily rely on simple heuristics like Central Pattern Generator (CPG) [Matsuoka 1987]. The notion of CPG stems from neuro-biology and refers to a neural network found in the spinal cord of animals that produce oscillatory patterns [Cheng et al. 1998]. Those patterns can be used to set the pace of limb motor behaviors, generating periodic gaits by the simple combination of nonlinear oscillators. If the method was first designed for quadrupeds on flat floor, it has been successfully extended to snake-like robots [Wu and Ma 2010], quadrupeds on rough terrain [Ajallooeian et al. 2013], swimming fish robots [Yu et al. 2016], hexapods [Bai et al. 2019], and humanoids [Dzeladini et al. 2018]. In a sense, CPGs are an embodiement of the reflex aspect of the locomotion, without any consideration of kinematic or dynamic constraints. They do not consider the dynamic model of the system and are not able to anticipate changes in trajectory resulting from new environment updates. In other words, they cannot produce very complex motions and have thus not been examined further in the frame of this thesis. They provide, however, interesting considerations about the periodic nature of locomotion, and illustrate that walking can be performed by considering only the essential properties of the system.

Passive walkers

Another interesting approach to biped locomotion consists in leveraging the cyclic dynamics of passive walking machines [McGeer 1990], i.e. systems built to naturally display periodic walking motions powered by gravity. These machines usually come

with no actuators, or just enough to stabilize their gait around a given limit cycle. The main advantage of passive walkers lies in the efficiency of the design in terms of energy consumption: as such, they help roboticists and biomechanicians to better understand the core principles of locomotion and build even more efficient legged systems.

Many different concepts of passive walkers have been studied during the last two decades, ranging from the simplest possible model of a point mass attached to a single leg [Garcia et al. 1998] to 3D passive bipeds with knees [Collins et al. 2001] and efficient anthropomorphic robots able to walk on flat floor with minimum actuation [Wisse et al. 2007].

Passive walkers are nonetheless limited by the simplicity of their design and their lack of actuation. They are in essence dynamic machines, meaning that they cannot perform quasi-static motions nor can they get away from the cyclic trajectory imposed by their mechanical design. They are however useful to get an insight over what an efficient walking motion looks like. In some ways, building a passive walker amounts to solving an optimization problem through physics alone, rather than on a computer.

2.2 Optimization for locomotion

By drawing inspiration from passive walkers, researchers developed gait cycles for actuated bipeds which minimize energy consumption [Channon et al. 1992; Roussel et al. 1998]. The next step ahead was to rewrite the locomotion planning problem as an optimization problem, by transcribing high-level walking commands into costs and system dynamics into constraints. However, the problem of stability has quickly become prevalent in the study of anthropomorphic locomotion. Indeed, since the stable support of humanoids is limited to the convex polygon of their feet, they tend to fall over easily. The immediate solution to this issue consists in generating a Center of Mass (CoM) trajectory whose projection on the ground remains inside the support polygon of the feet in contact [Zheng and Shen 1990]. This leads to a quasi-static locomotion far from human-like gait.

2.2.1 Trajectory optimization based on template models

To produce more complex optimal trajectories while being computationally tractable, researchers turned to template models and simplified stability criteria. The aim was to extract the fundamental dynamics of walking in order to be able to predict the evolution of the system in simple experimental conditions, like flat ground locomotion.

The ZMP criterion

At the heart of biped locomotion stands the need to regulate the centroidal dynamics, an essential concept which will be described more precisely in the next chapters. In order to produce dynamic gaits, researchers have proposed walking methods based on the optimization of the Zero Moment Point (ZMP) [Vukobratovic and Juricic 1969] trajectory. The ZMP is an imaginary point on the ground at which the moment of contact forces is null in both horizontal directions. It actually describes a non-tilting condition for the feet: as long as the ZMP remains inside the convex hull of the contact points, the robot will not tip over. Likewise, it ensures the dynamic

stability of the robot, whereas trajectories based on the CoM alone could only ensure static stability. Dynamic stability guarantees that at each instant in time, the robot is able to avoid falling and to stop its motion within a few steps. This means that even if the CoM happens to lay outside of its polygon of support, the control law can bring the system back to a statically stable configuration. Methods based on ZMP were successfully deployed on the WL-10RD robot [Takanishi et al. 1989], Honda robot [Hirai et al. 1998], H5 robot [KAGAMI et al. 2000] and HRP-2 robot [Huang et al. 2001]. These early works were often limited by the fact that the computed ZMP trajectory has to be feasible by the robot, meaning that there must exist a joint trajectory that achieves the desired ZMP tracking. To overcome this issue, Huang [Huang et al. 2001] proposed heuristics based on the generation of hip and feet trajectories via spline functions, followed by the computation of the optimal ZMP and the selection of the one hip trajectory resulting in the widest stability margins.

Linear Inverted Pendulum model

One of the first template models successfully used in robotics is the Linear Inverted Pendulum (LIP) model [Sano and Furusho 1990; Kajita and Tani 1991]. The main idea behind LIP is to linearize the CoM dynamics by reducing the robot to a point mass fixed at the end of an inverted pendulum with massless shaft. This simplification however comes with very conservative assumptions like constant CoM height and zero angular momentum.

The convexity of the LIP formulation has allowed the generation of dynamic trajectories in real time [Sugihara et al. 2002], a key feature for producing any high-performance motion. Combined with the ZMP stability criterion, LIP approaches resulted in efficient real-time preview schemes [Kajita et al. 2003] which can leverage future information to compute current control outputs. This work marked the beginning of predictive control applied to biped locomotion. It is based on three main features: (1) A cart-table model with linear dynamics, (2) a predictive control framework whose objective is to track a ZMP trajectory reference over a preview horizon and (3) an inverse kinematics (IK) controller which produces joint commands so as to follow the computed ZMP. The predictive control problem is solved through Linear Quadratic Regulator (LQR) in real-time. Since the cart-table model only loosely approximates the multibody dynamics of the real robot, the preview control becomes essential to predict and compensate future errors resulting from model simplification.

In the last two decades, many variations of the LIP model were explored. Recent formulations now integrate CoM height motion [Englsberger and Ott 2012; Koolen et al. 2016a] or treat it as a disturbance [Brasseur et al. 2015]. Since angular momentum recovery balance is crucial to biped locomotion [Kuo and Zajac 1993], enhanced pendulum models using flywheel were proposed for humanoids [Komura et al. 2005; Pratt et al. 2006]. Among other LIP extensions, the Spring-Loaded Inverted Pendulum (SLIP) model was proved to encode the natural running dynamics of animals [Holmes et al. 2006] and thus produced impressive motions on humanoids [Garofalo et al. 2012; Wensing and Orin 2013]. Xiong and Ames [Xiong and Ames 2019] studied a combination between LIP dynamics and nonlinear SLIP dynamics, which are more complex to compute online, in order to decouple the periodic and transitional phases of the walking motion. Mordatch et al [Mordatch et al. 2010] deployed a linearized version of SLIP inside a physics-based locomotion controller. In [Rezazadeh and Hurst 2020], the vertical motion of the SLIP is submitted to a forced-oscillation scheme to ensure stability, and the dynamics is decoupled into

vertical and horizontal motions.

Generic stability criterion

ZMP formulations are nonetheless limited to horizontal planes with sufficient friction. A generalized stability criterion for arbitrary terrain was thus proposed in [Hirukawa et al. 2006]: the polyhedral Convex Wrench Cone (CWC) corresponding to the limits imposed on the contact and gravitational wrench applied to the CoM of the robot. Although the computation of CWC is costly, it only needs to be performed once for each pre-defined contact, allowing for offline pre-computation [Hirukawa et al. 2007]. The efficiency of CWC formulation was later improved in [Caron et al. 2015a]. The CWC approach has fueled many works on contact force optimization and robust balance, such as [Ott et al. 2011; Koolen et al. 2016b; Audren and Kheddar 2018]. In [Dai and Tedrake 2016], a convex upper bound of the real centroidal momentum is minimized in order to plan robust walking motion through convex optimization using CWC. Other works followed the idea of computing a desired force trajectory to ensure balance [Hyon et al. 2007] or computing an optimal distribution of contact forces by making use of torque redundancy [Righetti et al. 2011]. The strength of this approach is that it does not require a precise dynamic model to achieve robust and compliant balance.

While CWC and ZMP provide reliable information about how to ensure stability, they do not explain how to recover it after a large disturbance. The concept of Capture Point (CP) was introduced in [Pratt et al. 2006] as a way to define a point over which the CoM should be in the near future to prevent falling. CP was later extended to the 3-dimensional Divergent Component of Motion (DCM) [Englsberger et al. 2013] and time-varying 3D DCM [Hopkins et al. 2014]. CP was successfully implemented on torque-controlled robots [Englsberger et al. 2011; Koolen et al. 2016b] and combined with operational space schemes [Ramos et al. 2014]. Similarly, DCM trajectory generation and tracking approaches were reported in [Englsberger et al. 2014; Griffin and Leonessa 2016; Caron et al. 2019]. The key element to this success is the decomposition of the centroidal dynamics between stable and unstable parts, as feedback control is then only needed on the unstable part.

Another notable approach to dynamic walk involves the projection of the complete model on the zero dynamics submanifold, leading to exponentially stable periodic orbits for walking motions [Westervelt et al. 2003; Westervelt et al. 2004; Sreenath et al. 2011]. This idea, named Hybrid Zero Dynamics (HZD), is particularly useful for building offline gait libraries [Galliker et al. 2022] or capturing the exact underactuated dynamics of the robot in order to design stable gait feedback controllers [Grizzle and Christine 2017].

Template models like the LIP fueled the first approaches toward trajectory optimization in locomotion context. The simplicity of LIP dynamics has allowed to cut computation load and to develop receding horizon techniques that will later lead to Model Predictive Control (MPC) theory. On the other hand, dynamic stability criteria like the ZMP or CWC have allowed to consider force and actuation limits in the optimization problem.

Although elegant, the LIP formulation remains a linear approximation of real walking dynamics, with no consideration for kinematic feasibility, limbs inertia, or total angular momentum. Practical implementations of LIP schemes require fine-tuned tracking control and particular heuristics to enforce feasibility; even yet, the

original LIP model is not fitted to generate complex multi-contact whole-body motions on uneven terrain.

2.2.2 Nonlinear fullscale trajectory optimization

The locomotion problem of a full legged robot involves unilateral contact forces, friction cones, kinematic and dynamic limits, as well as nonlinear dynamics. Powerful nonlinear solvers are required to tackle the whole complexity of this problem. The algorithms used to solve it can be classified into three main branches [Diehl et al. 2007]:

- Dynamic Programming Methods, based on the solution of a Hamilton-Jacobi-Bellman equation, compute an optimal feedback policy on the entire state space. They are usually not tailored for high-dimensional problems as they tend to become overly expensive to compute.
- Indirect Methods, based either on Pontryagin's Maximum Principle or Euler-Lagrange theory, solve an ordinary differential equation before discretizing it to produce the optimal trajectory. The complexity of the dynamics equations to solve massively hinders their use in robotics.
- Direct Methods attempt to first discretize the problem, then solve the resulting finite dimension programming problem using various numerical tools. This class of methods is particularly popular among the robotics community as a finite dimension problem is easier to handle than an infinite one.

Among the Direct Methods, three main classes of transcription are commonly used in robotics:

- Single shooting methods consider only the control as a decision variable and compute the state by integrating the dynamics, starting from the beginning of the horizon. This class of algorithms is very sensitive to initial conditions and behaves poorly with long horizons, so it is not often considered in robotics.
- Multiple shooting methods divide the time horizon into a collection of smaller intervals and solve the problem on each interval using a single shooting approach, while enforcing continuity constraints at the bounds of each sub-trajectory.
- Finally, collocation methods discretize simultaneously the state and control variables and make them match over a finite set of collocation points, then interpolate between each collocation point using polynomials or splines.

The DDP algorithm

Among significant trajectory optimization techniques, Differential Dynamics Programming (DDP) [Mayne 1973] has become quite popular in robotics because of its second-order convergence rate, linear complexity in horizon length and local optimal feedback policy. The algorithm efficiently exploits the sparsity induced by the temporal structure of the optimal control problem, allowing for cheap computation efforts. The simplicity of the DDP formulation is also likely a key factor to explain its popularity, given that it can lead to efficient and robust implementations dedicated to any specific system. DDP is considered to be a shooting method based on Bellman's optimality principle [Bellman 1954]. It operates by linearizing the dynamics

at the second order around an initial trajectory solution, then performing a backward Riccati recursion in order to compute the optimal cost-to-go starting from the end of the horizon. Finally, a nonlinear forward rollout is performed to retrieve the trajectory state from the optimal feedback policy. This rollout is essential to fulfill the dynamics constraints along the trajectory.

Despite its advantages, the DDP formulation faces several dowsides: it cannot take inequalities constraints into account, unless they are written as quadratic barrier functions. Besides, the algorithm is at its core sequential, even if some interesting extensions based on decoupling the optimization from the dynamics have been proposed to make it parallelizable [Lantoine and Russell 2012]. Finally, DDP is considered less numerically robust than other NLP solvers and is sensitive to conditioning issues.

In the frame of this thesis, an improved version of the DDP algorithm has been used to generate whole-body trajectories in real-time on a fullscale humanoid robot. Additionally, the natural feedback policy induced by the Riccati recursion has been leveraged to bridge the frequency gap between low and high-level controllers. In our opinion, DDP is the best choice of solvers to implement quickly and easily a proof of concept of full dynamics predictive control.

DDP-related methods

The DDP algorithm implies the computation of the second order derivative of the dynamics, which can be difficult to obtain. Simpler versions of the algorithm omitting this term have been implemented for real-time applications, like the iterative Linear Quadratic Gaussian (iLQG) [Todorov and Li 2005], sometimes called iterative Linear Quadratic Regulator (iLQR), suited for nonlinear systems subject to control constraints. The ALTRO solver [Howell et al. 2019] notably uses iLQR combined with augmented Lagrangian to solve for constrained trajectory optimization with unfeasible initial guesses. Similar to DDP is also the Sequential Linear Quadratic (SLQ) algorithm [Sideris and Bobrow 2005], again based on the resolution of a sequence of linear quadratic sub-problems via a Riccati approach.

Another way to simplify the DDP is to perform a linear rollout of the dynamics instead of a nonlinear one. Doing so results in a Sequential Quadratic Programming (SQP) [Nocedal and Wright 2006b] algorithm which may include equality and inequality constraints, at the price of having to consider a merit function to estimate the violation of constraints induced by the linear approximation. Such a merit function is usually not easy to write as no generic method exists to define it. Nevertheless, many state-of-the-art NLP solvers are based on SQP paradigm, like Acados [Verschueren et al. 2020] or SNOPT [Gill et al. 2002].

2.2.3 Contact scheduling

Planning a sequence of feasible contacts in cluttered environments is on its own a very complex problem whose resolution depends on the model chosen to represent contact physics. In the realm of soft robotics, bodies are treated as continuously deformable objects which exhibit spring-damper behavior when coming into contact [Kim et al. 2013]. Despite being a more accurate description of contact physics, this representation is often viewed as too complex to be embedded into a full control pipeline. As a consequence, rigib body models are most often used in fullscale robotics. Under the rigid body assumption, surfaces cannot be deformed and any

collision results in an impact event in which forces and velocities become discontinuous.

Hybrid dynamics

A popular approach to locomotion with rigid contacts consists in optimizing independently the motion over each contact mode, where the dynamics remains differentiable. This amounts to decomposing the locomotion problem into its contact planning component and whole-body control component. Mixed-Integer Programming (MIP) [Deits and Tedrake 2014] or bilevel optimization [Wampler and Popovic 2009] are widespread tools to solve the contact planning problem. Other methods leverage sampling-based algorithms like RRT [Perrin et al. 2012] or probabilistic roadmaps [Tonneau et al. 2018].

Although simpler to implement, hybrid dynamics techniques add another layer of complexity to the control framework and tend to make the locomotion more conservative. Besides, the whole-body feasibility of the contact sequence given by the planner must be actively enforced.

Implicit dynamics

Solving simulaneously for the contact sequence and whole-body motion appears as a significant challenge, but allows to explore richer modes of locomotion. Some works [Posa et al. 2014] proposed to rewrite the hybrid dynamics as complementarity constraints stating that forces can be non-zero only when collision distance is null, and vice versa [Brogliato 2016]. Hybrid modes are thus implicitely described through a state-force relationship rather than integer variables. Another approach is to smooth the contact constraints [Todorov 2014] and perform continuous optimization over contact placements and motion [Mordatch et al. 2012].

Implicit dynamics is a promising angle of work to produce complex multi-contact trajectories through optimization alone, although it also suffers from poor numerical conditioning and heavy computational load.

2.2.4 Optimization for whole-body locomotion

Both multiple shooting and collocation methods lead to solving a high-dimensional NonLinear Problem (NLP) which is overly complex and may contain several local minima. As a consequence, most multi-contact trajectory optimization problems can only be solved offline, unless some simplifying heuristics is leveraged. Nevertheless, direct methods were successfully implemented on various robotics platforms for the last two decades: in [Schultz and Mombaur 2010], a direct multiple shooting algorithm generates human-like running motions based on multiphase periodic cycles and pre-specified contact sequence. In [Hereid et al. 2015], a multiple shooting formulation is used to optimize the virtual constraints in gait generation. In [Aller et al. 2022], the same transcription scheme allows to benchmark sit-to-stand motions with the humanoid REEM-C. Using a pre-defined contact sequence and a centroidal model solved through SQP, the robot HRP-2 climbed stairs with handrail in [Kudruss et al. 2015].

As for collocation methods, Posa et al. [Posa et al. 2014] used it to generate walking trajectories without the need of a pre-defined contact sequence. Two years later, the same authors introduced DIRCON [Posa et al. 2016], a trajectory optimization algorithm based on collocation and third order integration accuracy. Ma et al. [Ma

et al. 2019] applied the collocation paradigm to gait planning for quadrupeds. Recently, contact-implicit schemes combined with collocation have been attracting a lot of interest in the field [Doshi et al. 2018; Manchester et al. 2019], with some of them reaching real-time capabilities [Aydinoglu and Posa 2022].

As for the DDP algorithm, it was successfully used as a whole-body planning tool to grasp the full dynamics of the robot in [Neunert et al. 2017]. In [Budhiraja et al. 2019], the locomotion problem is separated into actuated and unactuated parts then solved conjointly using an Alternative Descent Method of Multiplier (ADMM); the whole-body part of the dynamics is obtained through a DDP scheme. In [Todorov 2014], the DDP paradigm is embedded into a full simulation pipeline with relaxed constraints and fast forward dynamics. The technique, originally not suited to handle inequality constraints, was generalized in [Tassa et al. 2014] to accommodate box inequality constraints on control. It was also extended to implicit dynamics with a closed form of the backward pass and differentiable contact model in [Chatzinikolaidis and Li 2021], allowing the solver to discover new contact sequence modes. A promising approach based on a mix of DDP and augmented Lagrangian has recently been proposed to handle inequality constraints [Jallet et al. 2022a], but has not been implemented yet on hardware. Similarly to DDP, SLQ was also integrated into a multi-level optimization framework in [Farshidian et al. 2016], where the algorithm outputs continuous control command and optimal contact switching times. Trajectory optimization based on iLQG techniques was reported in [Tassa et al. 2012] where the solver optimizes for state, control and contact forces simultaneously, without the need for prior contact sequence specification. In [Suh et al. 2022], iLQG is used along a contact-implicit scheme and a stochastic formulation of the dynamics.

Despite impressive results in terms of gait discovery and multicontact motion generation, trajectory optimization techniques considering the robot full dynamics were, until recently, restricted to open-loop applications due to their high computational load. They typically cannot cope with brutal problem changes or accumulating errors caused by model discrepancy, and are still overall limited to provide reference trajectories to a low-level tracking control scheme.

2.2.5 Gait libraries

To mimic online re-planning capacities, the idea of gait library [Liu et al. 2013] was proposed in the literature: it consists in computing offline a comprehensive dataset of relevant walking trajectories, then selecting and refining online the most adapted reference motion. Gong et al. [Gong et al. 2019] implemented a gait library enforced through inverse kinematics feedback on the biped Cassie to produce fast and robust motions; Nguyen et al. [Nguyen et al. 2020] used a gait library and control barrier functions to cross stepping stones; Guo et al. [Guo et al. 2021] constructed a whole-body trajectory library that synthesises an optimal reference based on the online centroidal prediction. The downside of this class of methods lies in the difficulty to interpolate online the stored trajectories in order to produce a good reference starting from the current state of the system.

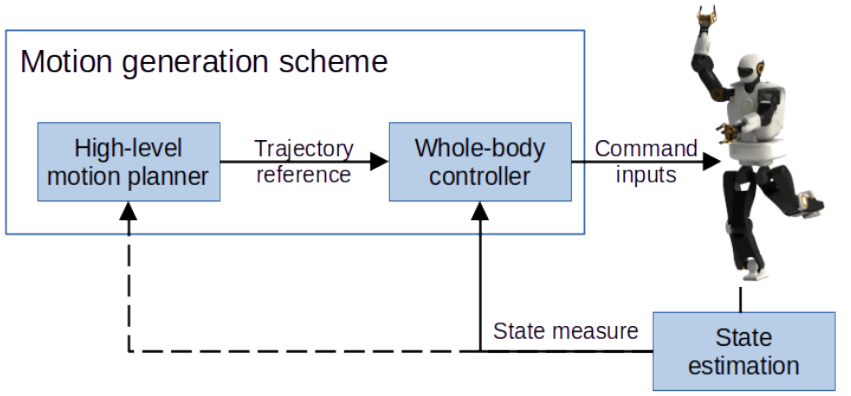

FIGURE 2.2: Generic structure of mixed control formulations. The feedback loop on the planner may not exist, depending on the nature of the planning problem. In the general case, the whole-body control frequency is one or two orders of magnitude higher than the planner frequency.

2.3 Mixed control approaches

Mixed control formulations split the locomotion problem between the high-level planning part (generating a trajectory reference) and low-level whole-body control part (tracking the reference), as illustrated in Fig. 2.2. Due to their computational complexity, most optimization solvers cannot run online in closed-loop, or not fast enough to prevent the system from drifting. A low-level controller is then needed to follow the desired trajectory and adapt it to environmental variations, model uncertainties, kinematic and dynamic limits, and so on. From a practical point of view, a robust, compliant and reactive low-level control is the key to any successful implementation on robotic platforms.

2.3.1 Problem formulation

The goal of the low-level controller is to produce torque commands that guarantee the execution of the reference plan on hardware. First motion control schemes made use of very simple control frameworks such as Proportional Integral Derivative (PID) feedback loop to perform trajectory tracking. This framework is especially dominant in industrial robotics [Arimoto 1984; Islam and Liu 2011] as it is suited for precise position control. However, PID schemes operate in the joint space of the robot and are not practical to use from a task-oriented point of view. Besides, they do not take into account the dynamics of the robot and can only follow reference motions that are geometrically and dynamically feasible.

The operational space framework [Khatib 1987] was developed in order to focus on task execution and make the control more intuitive. For each task to be considered, the relative error is expressed in the relevant task space, which is usually of smaller dimension, then transcribed into a whole-body control through projection on the nullspace of higher priority tasks. The formalism of operational space approach allows to decouple the different tasks and to define a hierarchy among them by exploiting the system redundancy.

A straightforward application of operational space framework is inverse kinematics (IK) control, which consists in finding the joint variation that leads to the completion of a given task objective, for example a desired end effector position in

Cartesian space. Kinematic constraints can be enforced at small cost thanks to the hierarchical aspect of the formulation, leading to fast implementation on real humanoid robots [Gienger et al. 2005; Mansard and Chaumette 2007].

For walking and balancing tasks, IK techniques quickly show their limits since they do not consider contact interactions and balance constraints. On the other hand, Inverse Dynamics (ID) makes the connection between the generalized control torques and joint accelerations, while including contact constraints and underactuation. Extensions of the operational space to ID tasks were quickly proposed to take into account dynamic feasibility while performing posture objectives [Khatib et al. 2004]. In [Saab et al. 2013], the formalism is generalized to the full dynamics of a humanoid robot, with reduced formulation of multiple rigid planar contacts.

2.3.2 Low-level control solvers

Operational-space formulations were first solved by computing an analytical solution through nullspace projections based on the dynamically consistent Jacobian of the task [Khatib et al. 2004; Sentis and Khatib 2005]. Later, this approach was used to perform compliant balance on the torque-controlled robot Toro [Henze et al. 2016]. In [Lee et al. 2016], a robust and dynamic model-free controller implemented a force-level operational-space framework with no need for costly model identification, thanks to an online dynamics estimation scheme and adaptive control modes.

Limitations of nullspace projection methods include the fact that inequality constraints cannot be taken into account in the task formulation. To overcome this issue, Quadratic Programming (QP) approaches were quickly proposed and implemented on real platforms. QP stands as a popular local optimization framework that handles well inequality constraints such as friction cone or joint limits. Besides, QP benefits from efficient off-the-shell solvers and a significant community of users. First QP implementations [Park et al. 2007] reformulated the balance problem as a second-order cone programming problem easily solvable with convex solvers. To arbitrate between conflicting tasks and ensure robust balance, a weighting QP scheme is used in [Collette et al. 2008]. Leveraging smooth weight variations, a whole-body QP control handled transitional tasks in [Salini et al. 2011]. In order to enforce a strict hierarchy among tasks, the work of [Kanoun et al. 2009] proposed to solve a sequence of linear QP problems ordered by priority. Following this idea, hierarchical QP [Mansard 2012] was developed to blend the efficiency and generality of QP with the simplicity of task-function methods: by decoupling motion and force equations, the control formulation cancels the ill-conditioned parts of the problem and decreases the computational load, allowing for real-time applications. In [Herzog et al. 2014], a hierarchical QP framework is used to perform momentum control on torque-controlled humanoids. Combined with a walking pattern generator that outputs optimal centroidal trajectory and contact forces, the same method produced locomotion on non-planar surfaces [Ramos et al. 2012].

In [Herzog et al. 2016a], the QP formalism was completed with a LQR scheme that performed receding horizon tracking control. Using a multi-contact high-level planner, the robot HRP-2 climbed a ladder thanks to a multi-objective model based QP [Vaillant et al. 2014]. Its successor HRP-4 performed complex and dexterous manipulation tasks by combining force-task formalism with a multi-robot QP controller [Bouyarmane et al. 2019]. Using admittance control and QP-based wrench redistribution, HRP-4 also climbed 18.5 cm-height stairs with a LIP model [Caron et al. 2019]. In a similar vein, the work of [Focchi et al. 2017] demonstrated that directly optimizing the ground reaction forces through QP greatly improved legged

locomotion of torque-controlled robots on steep slopes, by contributing to prevent contact slippage.

The flexibility and efficiency of QP formulation may explain its popularity among the robotics community. With nowadays off-the-shelf solvers, this class of programs can easily be solved in a few milliseconds or less, matching the low-level frequency of current torque-controlled robots (1 kHz or higher). Real-time QP has been combined with operational-space approaches, optimal control techniques, soft contacts, passivity-based control, wrench distribution schemes, and so on. Its massive use as an instantaneous control tool is widely acknowledged as a natural evolution of task-function paradigm and least-square solvers [Wensing et al. 2022].

Mixed control formulations can be considered as state-of-the-art solutions to the practical problem of locomotion. However, they tend to lead to multi-layered control schemes with complex interconnections between control blocks. Mixed formulations imply at least two different optimization processes running in parallel, with no guarantee on the global optimality of the final motion. Additionally, when the high-level planner generates centroidal trajectories, the whole-body feasibility of the motion is often difficult to ensure.

2.4 Predictive control

Model Predictive Control (MPC) [Rawlings et al. 2017] provides a general and efficient toolbox for online re-planning while considering system dynamics and path constraints. By repeatedly solving a finite-horizon trajectory optimization problem in close-loop, MPC can deal with the issues of state drift, model uncertainties, unpredicted environment changes and external disturbances. Its performances are deeply linked to the length of the preview horizon and the computation frequency. MPC is viewed as a naturally compliant approach thanks to its intrinsic feedback properties.

The choice of the dynamic model to be used inside MPC appears to be of crucial importance, since a very complex model would drastically affect the re-planning frequency. As a consequence, first applications of MPC in robotics used template models and linearized dynamics to speed up trajectory computation. Recently, the rise in computer power has allowed even more complex MPC schemes to become viable.

2.4.1 Linear Predictive control

The main idea behind linear Model Predictive Control is to simplify the problem until standard convex optimization tools can be used to solve it. It often relies on template models that approximate the real dynamics and make it computationally tractable. In most cases, the angular part of the centroidal dynamics, which is infamously nonlinear, is neglected as well.

Following the work of Kajita et al. [Kajita et al. 2003], numerous mixed control formulations based on MPC were proposed to generate robust walking patterns. The core idea of these approaches consists in computing the centroidal part of the dynamics along a preview horizon and projecting the solution of the first timestep to the configuration space via an instantaneous whole-body control, usually a QP scheme. Thanks to the iterative recomputation of the optimal centroidal trajectory, the walking motion is continuously adapted to the current state of the system; additionally, the instantaneous controller plays the role of a trajectory stabilizer and guarantees the robustness and the compliance of the general control scheme.

Other works based on LQR formulation proved the reliability of such a framework [Nagasaka et al. 2004; Nishiwaki and Kagami 2006], but they still remain limited by the complexity of the contact constraints; besides, they need to rely on high-level adaptation schemes in order to track the reference trajectory [Park and Cho 2000; Huang et al. 2000; Wieber and Chevallereau 2006]. To overcome this issue and increase robustness, Wieber generalized the LQR of Kajita in the form of a linear MPC [Wieber 2006b]. Diedam et al. [Diedam et al. 2008] then extended this work by introducing footstep adaptation in the MPC and using QP rather than LQR to solve the OCP, allowing inequality constraints to be taken into account. Thanks to this footstep adaptation, the robot is provided with strong recovery capabilities. Later, the work of [Herdt et al. 2010] enhanced the flexibility and independancy of the linear MPC scheme by replacing the ZMP trajectory reference with a user-defined translation speed. Another notable work on linear MPC considers centroidal dynamics with negligible angular momentum variations and establishes a dynamic balance criterion that is more general than the ZMP [Perrin et al. 2018]. This scheme allows to generate 3D locomotion with multiple non-coplanar contacts, based on a succession of convex quadratically constrained QP.

Although simple to implement, linear MPC is based on restrictive hypotheses that limit its range of applications. It usually neglects the angular momentum of the trunk, which can be significant during jumping or running. Besides, ensuring the compatibility of whole-body motions with desired reference trajectory has remained a challenging task. As the computation power of modern computers has kept on increasing, the robotics community has turned toward more complex models with nonlinear dynamics.

2.4.2 Nonlinear Predictive Control with template models

The LIP approach, although computationally tractable, remains a gross approximation of the robot complete dynamics, and is not fitted for complex scenarios involving non-coplanar contacts. Single Rigid Body (SRB) model, on the other hand, reduces the robot to a unique body with constant inertia while keeping the nonlinear form of the dynamics. This formulation is particularly adapted to quadruped robots, since their legs are light with respect to the rest of their structures, but it necessitates complex nonlinear solvers to handle it. Consequently, SRB is frequently used in trajectory optimization [Winkler et al. 2018] or simplified into linear schemes in order to become real-time [Di Carlo et al. 2018; Villarreal Magaña et al. 2020; Ding et al. 2020].

In order to handle the nonlinearity of the dynamics in real-time, several Nonlinear MPC (NMPC) frameworks were proposed in the literature. One of them is the Real-Time Iteration scheme [Diehl et al. 2005], a framework that continuously computes refined approximations of the optimal control through a Newton-type multiple shooting algorithm. Extensions of this scheme were introduced to automatically cope with footstep placements and local obstacle avoidance on the humanoid HRP-2 [Naveau et al. 2017]. Rathod et al. [Rathod et al. 2021] deployed the Real-Time Iteration scheme on the quadruped HYQ and performed terrain adaptation and disturbance rejections. Since Real-Time Iteration formulations do not scale well with longer horizons, a condensing NMPC approach with linear complexity over the horizon length was proposed in [Vukov et al. 2013]. Later, Quirynen et al. [Quirynen et al. 2015] presented a fast NMPC scheme allowing microsecond computation

thanks to code generation. This scheme is used in [Kamel et al. 2015] for the control of an aerial drone. In [Bledt and Kim 2019], a regularized nonlinear predictive control is implemented on the MIT Cheetah 3 to solve for footstep placements and ground reaction forces.

As for humanoid robots, first NMPC applications were concerned with the exact resolution of the 3D biped centroidal dynamics [Orin et al. 2013]. The key point of the approach is to consider the CoM not as a point mass, but as a 3D body with inertia depending on the robot configuration. Consequently, contact forces have an impact on the variation of angular momentum. Such a formulation is often called a walking pattern generator, and its function is generally to compute optimal CoM, angular momentum and ground forces trajectories respecting the contact constraints. It is then up to the whole-body low-level control to design a limb motion that will satisfy the high-level centroidal trajectory.

An efficient pattern generator based on centroidal dynamics was reported in [Hirukawa et al. 2007], where the state and control are optimized over a reference CoM trajectory and force distributions. A MPC formulation of this walking pattern generator was proposed and demonstrated in [Carpentier et al. 2016], where the control scheme is able to handle arbitrary 3D contacts and can work without reference trajectories. In [Ponton et al. 2016], the centroidal dynamics non-convexity is relaxed so as to be embedded inside a constrained QP, while feet reachability constraints are approximated as ellipse regions. This convex approximation is then combined with a foostep contact planner based on mixed integer programming, allowing push recovery and online adaptation of the contact sequence. However, the experimental results are limited to simulation, and the method is not formulated as a MPC despite a reduced computation load (100 ms for 5 iterations).

Similarly to linear MPC, nonlinear centroidal MPC faces the challenge of kinematic feasibility and low-level control adaptation. High-level walking pattern generators often have to rely on dynamic stabilizer to filter the desired CoM trajectory and produce stable locomotion. Tackling the walking problem as a whole requires to increase the complexity of the model even further.

2.4.3 Full kinematics MPC

The weakness of the centroidal approach lies in the feasibility constraints (kinematic limits, footstep length, momentum match between full kinematics and centroidal dynamics), as for each optimal CoM trajectory there is no guarantee that a feasible whole-body motion exists to achieve it [Carpentier and Mansard 2018b]. In [Dai et al. 2014], the centroidal dynamics and whole-body kinematics of the humanoid Atlas are conjointly solved in order to enforce the kinematic limits along the horizon; yet the approach is not quite real-time and cannot be formulated as a MPC. Using a SLQ scheme and a multi-processing estimation of the value function, Farshidian et al. [Farshidian et al. 2017] achieved real-time computation of the CoM dynamics and full kinematics of the HyQ quadruped. Such an approach is repeated in [Grandia et al. 2019] with a DDP-based solver that provides a Riccati feedback policy suited to bridge the gap between the high-level MPC and low-level actuation. In this work, hard constraints are approximated through relaxed barriers functions.

Another interesting approach is presented in [Meduri et al. 2022], where the motion plan is decomposed into its centroidal dynamics part and full kinematics part. The two problems are then solved alternatively until a kino-dynamic consensus is

reached [Herzog et al. 2016b]. Experiments on real hardware have demonstrated a MPC frequency of 20 Hz on the quadruped Solo12.

Compared to MPC with generic template models, MPC schemes based on kinematics preview partly solve the feasibility issue. Nevertheless, such an approach does not consider the complete dynamics of the system nor the effects of contact interaction, which are key points in multi-contact scenarios with heavy robots. Moreover, MPC with full kinematics preview still relies on fine-tuned whole-body controllers to produce optimal torque commands and robustify the motion.

2.4.4 Full dynamics MPC

Full dynamics MPC, although overly complex, can leverage the full potential of the robot dynamics, ensuring long-term balance and recovery from strong perturbations [Corbères et al. 2021]. For robots with heavy limbs, it is best to take into account the inertia of the end-effectors and their effects on angular momentum inside the trajectory preview. Besides, full dynamics MPC can directly provide the actuation with an optimal torque command, in case the robot is torque-controlled. Reduced MPC schemes that rely on template models have to use instantaneous tracking controllers (for example, a QP framework) to cast the centroidal solution over the whole-body configuration, usually at the price of approximating kinematics and dynamics feasibility. On the other hand, full dynamics MPC approaches have to face the curse of dimensionality; as a consequence, they usually work with shorter time horizons, thus hindering their predictive properties. As re-planning frequency is critical to ensure reactivity and robustness, a very efficient trajectory optimization technique must be chosen to solve the optimal control problem. Most recent works on full dynamics MPC implement DDP-based algorithms like iLQR or SLQ, or Gauss-Newton approximations to speed up the computation of dynamics.

In [Erez et al. 2013], a whole-body MPC framework generates biped locomotion in simulation, with a working frequency of 30 Hz for 500 ms of horizon. Such performances are obtained thanks to an iLQG scheme combined with MuJoCo, a very powerful physics engine that enables contact smoothing methods. A more simple approach was taken in [Mason et al. 2016], where a full dynamics LQR is set to track a reference walking motion computed offline by inverse dynamics, using the ZMP balance criterion. The control scheme is based on a linearization of the problem around key positions, and is thus very fast to compute online. Experimental results include balancing and biped walking on the hydraulic torque-controlled humanoid Sarcos. A SLQ formulation is leveraged in [Neunert et al. 2016] to solve a full-dynamics unconstrained MPC problem in real-time; the resulting feedforward and feedback gains are directly applied inside the low-level actuation, without relying on additional tracking controllers. Hardware demonstrations involve a hexacopter and a ballbot, both systems being highly dynamic and unstable.

One of the first demonstrations of full dynamics MPC on high-dimensional hardware reached a frequency of 190 Hz for a predicted horizon of half a second [Neunert et al. 2018]. Using explicit contact dynamics, contact locations and timings are optimized along the state and control trajectories, which are then tracked by a customized PD feedback loop. The optimization scheme, solved through a Gauss-Newton multiple shooting approach, was successfully deployed on the quadrupeds HyQ and ANYmal. In [Mastalli et al. 2022a], the low-level feedback policy is a direct output of the full-body MPC executed at 40 Hz on the torque-controlled ANYmal. This work notably paves the way for the use of local Riccati policies in low-level

torque control for locomotion tasks. Mastalli et al. [Mastalli et al. 2022b] also presented a novel DDP scheme based on inverse dynamics and embedded it inside a full locomotion-perception framework for quadrupeds. This approach benefits from lower computation costs on derivatives and a higher convergence rate at the price of having to solve dynamics equality constraints. Using a primal-dual interior point method with Gauss-Newton approximation, a full dynamics MPC with timing optimization is solved in [Katayama and Ohtsuka 2022] at a frequency of 400 Hz and produces jumping motions on the quadruped Unitree. Regarding humanoid robots, the first demonstration of whole-body MPC took place on the HRP-2 platform, using a DDP scheme without closing the loop on the state measurement [Koenemann et al. 2015]. Later, biped locomotion leveraging the full dynamics of the robot was performed in [Galliker et al. 2022] with an offline gait computed through hybrid zero dynamics. Yet, the walking motion was limited to 2 dimensions and relied on high-level reference trajectories for state and control, either defined heuristically or extracted from an external gait library.

In order to find a compromise between computation costs and prediction capacities, some works resorted to a mix of template models and whole-body models. In [Li et al. 2021], a hierarchy of models is used inside a single optimization framework and the resulting MPC is benchmarked on simulation with quadruped, biped and quadrotor. For legged locomotion, the current stance and next flying phase are implemented through a whole-body model, while subsequent phases are modeled as a reduced template. In [Yeganegi et al. 2022], a reduced MPC with 2 s horizon is implemented along a full-body MPC with 0.3 s horizon, solved through iLQG. The resulting scheme was tested in simulation with humanoid robots subject to unpredicted pushes and delays.

Although very recent, full dynamics MPC is now mature enough to produce robust and adaptable walking motions on heavy quadrupeds. Still, experimental demonstrations involving humanoids are lacking, as biped locomotion is much more difficult to perform due to its intrinsic instability and reduced support area.

2.5 Learning motion: a policy-based approach to robotics

Most recent breakthroughs in legged robotics were driven by convex optimization and ever-increasing onboard computational power. Trajectory-based approaches have become more and more complex, taking into account the whole dynamics of the system, sometimes also discovering new modes of locomotion on the fly. However, they are still limited by real-time constraints and tend to exhibit local optimal solutions when the problem becomes too complex. Overall, they struggle to perform an exhaustive search of the solution space.

On the other hand, policy-based methods are becoming an interesting alternative to optimization in robotics. The aim of such approaches is to build an accurate representation of the mapping between current state and desired control through statistical or learning processes over large chunks of data. Despite impressive results in terms of policy inference and adaptation to unmodeled dynamics, they remain limited by their sensitivity to hyper-parameters, their reliance on extensive training datasets and their costly repetitions of trials and errors. Moreover, they do not scale up well and are difficult to implement on real hardware, particularly on humanoids, as they are not suited to handle constraints.

In order to tackle these issues, the robotics community has started to explore hybrid solutions, either by attempting to identify the model parameters of the optimization scheme through learning or by guiding the optimization with relevant learned cues. Some works have also proposed to reverse roles and use model optimization as a search guide for learning. Others have explored the option of leveraging Reinforcement Learning (RL) to compute end-to-end control policies based on extensive offline training in simulation.

2.5.1 Learning end-to-end

Biological systems exhibit great compliance properties stemming from the adaptive impedance of their neuromuscular system [Selen et al. 2009]. To reproduce such compliance in robotics amounts to the use of time-varying PID gains, an approach often labeled gain scheduling. However, finding the optimal gains with respect to the completion of some given task remains a difficult issue. Some authors suggested to synthesize a variable impedance control scheme with RL, making it tractable for a various range of applications on real hardware [Buchli et al. 2011].

Relying only on data exploitation to design an end-to-end controller remains nonetheless a challenge in robotics. In order to accumulate relevant locomotion data, one must choose between learning through trial-and-error on very expensive hardware or using approximate simulations that are difficult to transfer into the real world. In this context, differentiable simulators are a key element of successful RL implementation because they speed up the computation of gradient-based learning while ensuring numerical stability [Degrave et al. 2019]. It was also proposed to embed the entire physics simulation inside a neural network layer [Avila Belbute-Peres et al. 2018]. However, the sheer dimension of the state space is still problematic. Relying too much on gradients to guide the search tends to overstrain the learning strategy and to produce conservative motions. Conversely, reference-free methods lead to tedious explorations based on trial-and-error heuristics.

Despite those difficulties, impressive demonstrations of sim-to-real learning of legged locomotion are on the rise. In [Siekmann et al. 2021], the biped Cassie executed a set of different gaits learned by a policy network enforcing periodic costs. In [Hwangbo et al. 2019], the quadruped ANYmal tracked high-level velocity commands and performed push recovery thanks to a neural network strategy entirely trained in simulation. Using nothing but a deep RL policy trained on simulation, the biped robot HRP5 followed a footstep sequence and achieved walking, turning in place and stairs climbing [Singh et al. 2022].

End-to-end RL is nonetheless at a very early stage of maturity, and cannot yet exhibit reliable performances on biped platforms. Its convergence time, lack of generalization and inability to handle constraints still hinder its development at a broader scale.

2.5.2 Speeding up model-based approaches through learning

Since model-based optimization struggles to face the non-convexity of legged dynamics, it has been proposed to alleviate the computation load by learning offline the most complex parts of the problem, then using the learned function online inside an optimal solver. Some works choose to learn the value function of Riccati-based optimal control problem [Tamar et al. 2016; Lowrey et al. 2018; Deits et al. 2019], other focus on learning the algorithm initialization [Mansard et al. 2018; Melon et al.

2021], or cost functions [Tamar et al. 2017]. A different angle of work consists in using learning to overcome the nonlinearity of the dynamics: such an approach is leveraged in [Lenz et al. 2015] where a real-time MPC with learned dynamics is designed and tested in simulation over a wide range of tasks. In a similar vein, discontinuous contact dynamics can be learned through implicit representations [Pfrommer et al. 2020].

One of the most pending issues in reduced model approaches is the gap between computing an optimal centroidal reference and finding a whole-body motion to follow this reference. As for kinematics feasibility, some works rely on proxy constraints derived from rough approximations for footstep feasibility [Perrin et al. 2012; Naveau et al. 2017] or pre-computation of contact transitions [Orthey and Stasse 2013]. In [Carpentier et al. 2017], the authors proposed to learn whole-body feasibility constraints for locomotion with the aim of implementing them as a cost inside the centroidal optimization. Using deep RL techniques, the authors of [Pandala et al. 2022] learned the unmodeled whole-body dynamics constraints of their quadruped system and have integrated it inside a centroidal MPC. Another approach to the problem consists in replacing the whole-body low-level control with a trajectory adaptation scheme generated via deep RL [Gangapurwala et al. 2021]: the proposed solution is robust to unmodeled dynamics and embeds an exteroceptive feedback of the terrain map, so as to enforce adaptation to various environments.

Another popular use of data-driven methods in model-based robotics is to learn footstep planning in cluttered environments, through supervised learning [Kalakrishnan et al. 2011], neural network classifier [Cauligi et al. 2021] or RL approaches [Tsounis et al. 2020; Gangapurwala et al. 2021]. Since state-of-the-art algorithms like mixed-integer programming are still unable to perform online re-planning of contact sequence, such learning schemes are essential to achieve adaptable and reactive walk on real robots. Using attention-based encoder trained end-to-end, the quadruped ANYmal recently completed an hour-long hike through the Alps [Miki et al. 2022].

2.5.3 Leveraging model-based optimization to guide data exploration

Robotics is characterized by the huge dimension of its state-space. As a consequence, learning-based strategies can spend a major amount of time exploring unfeasible or uninteresting chunks of the solution space, or even get stranded in local minima. Guided policy search methods attempt to draw the policy learning process toward regions of greatest interest, usually regions which minimize a given cost. In [Levine and Koltun 2013], a DDP algorithm generates guiding samples for a neural network that performs various robotics motions. in [Tsounis et al. 2020], gradient methods are used to help the training of the footstep planning policy. Similarly, in the work of [Mordatch and Todorov 2014], trajectory optimization and function approximation are blended together using an alternating direction method of multipliers.

Other approaches use reduced models to fit data in simulation and perform an exhaustive search of the state space in order to extract relevant motion heuristics [Bledt and Kim 2020]. The adaptation laws are then applied back online on the quadruped mini Cheetah, exhibiting increased performances.

Policy-based methods, blended or not with optimization, provide a promising angle of work to tackle the inherent complexity of locomotion by outsourcing it to a library of data built through offline training or data compilation. Nevertheless, they remain an unknown territory with very poor theoretical ground and tedious training work, with a higher environmental impact due to the heavy use of computer resources.

2.6 Thesis formulation

Recent developments in robotics have lead to outstanding real-world demonstrations that seem to pave the way for widespread applications of versatile legged platforms. While the first successful walking achievements relied on basic linear optimization and overly simplified models or instantaneous tracking of offline-computed trajectories, new approaches dare to take into account the whole complexity of the model and to solve conjointly for planning and control in a cluttered environment. The increase of computational performances due to Moore's law has played an important role in recent breakthroughs, allowing to deal with even more complex problems online. Still, state-of-the-art methods for locomotion remain based on mixed control formulations with a decomposition of the problem into more tractable blocks.

We believe that now is the time to develop full dynamics optimal control algorithms for high-performance locomotion on various terrains. In the frame of this thesis, we formulate three propositions that we wish to argue:

- High-level planning and low-level control can be synthetized into a single optimization framework that is cheap to implement on real hardware.
- Full dynamics MPC can produce robust and reliable locomotion on heavy robots.
- Full dynamics MPC can be combined with learning methods to increase its exploration capacities.

Our contributions to the field of robotics are both theoretical and experimental. First, we proved that the Riccati matrix produced by the backward pass of a DDP scheme can be interpreted as a local and optimal feedback policy with regard to the initial state of the problem. This policy, computed at no additional cost, can fill the frequency gap between the high and low-level control of our framework, allowing more dynamic motions to be performed online on the robot. Furthermore, we showed that the backward pass of the DDP can be derived with respect to any other parameters to extract a new set of sensitivities acting as a whole-body control feedback loop.

Second, we demonstrated that a MPC integrating the whole dynamics of the system into the horizon can perform locomotion on generic terrains with a fullscale humanoid robot, thanks to strategic user-defined hints that help the solver to converge in one iteration only.

In a third step, we studied how to reduce the number of pre-computed references needed to walk in order to inject more flexibility into the control. Push recovery on flat floor was successfully performed on the robot, and stairs climbing without feet references was investigated in simulation.

Finally, we managed to connect our whole-body MPC scheme to an offline library of motion so as to handle the issue of local minima arising from non-convex

scenarios. Collision avoidance in real time was successfully demonstrated on the robot using a MOCAP target tracked by an end-effector.

Chapter 3

Mathematical tools for robotics

This chapter introduces various useful mathematical tools which will be used in the rest of this manuscript. Our main focus is on system modeling and optimal control methods with a particular emphasis on DDP algorithm.

3.1 Optimal control formulation

The generalized continuous OCP in trajectory-based methods is defined as the minimization of a terminal cost ℓ_T and the integral of a cost ℓ over a given horizon of size $\mathcal{T} > 0$:

$$\begin{aligned} \min_{\underline{\boldsymbol{x}},\underline{\boldsymbol{u}}} \int_0^{\mathcal{T}} \ell(\boldsymbol{x}(t),\boldsymbol{u}(t),t)dt + \ell_{\mathcal{T}}(\boldsymbol{x}(\mathcal{T})) \\ s.t. \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \forall t \in [0,\mathcal{T}], \quad \dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t),\boldsymbol{u}(t)) \end{aligned} \tag{3.1}$$

In this equation, f describes the dynamics of the system, $\boldsymbol{x}(t)$[1] the state of the system at time t and $\boldsymbol{u}(t)$ the control variable of the system. $\underline{\boldsymbol{x}} : t \mapsto \boldsymbol{x}(t)$ and $\underline{\boldsymbol{u}} : t \mapsto \boldsymbol{u}(t)$ represent the state and control trajectories over the entire horizon $[0,\mathcal{T}]$. The integral and terminal costs ℓ and $\ell_{\mathcal{T}}$ are encoding the tasks the robot is bound to perform (reference tracking, regularization, etc.).

An OCP for locomotion classically involves feasibility constraints on state and control, but such constraints are difficult to deal with in real-time context. Several recent works have proposed methods to directly handle hard constraints in high-dimensional OCP [Howell et al. 2019; Kazdadi et al. 2021], but we nevertheless choose to formulate our constraints inside the cost function through penalty barriers, for the sake of computation load.

As already discussed, three main approaches exist to solve problem (3.1): Dynamic Programming, Indirect Methods and Direct Methods.

[1] For the sake of readability, in the following sections and chapters, we will denote $A_b = \frac{\partial A}{\partial b}$, except when not suitable (e.g. $\frac{\partial \Delta u}{\partial x}$). Bold capital letters indicate matrices and bold letters indicate vectors.

Dynamic programming

This approach relies on the formulation of a Hamilton-Jacobi-Bellman partial differential equation linking current action with future cost value. Given a policy $\pi(\boldsymbol{x}, t) = \boldsymbol{u}$, we can define the continuous Value function to be the minimal possible cost-to-go starting from $\boldsymbol{x}$ and t:

$$\begin{aligned} v(\boldsymbol{x}, t) &= \min_{\pi} \Big(\int_t^{\mathcal{T}} \ell(\boldsymbol{x}(t), \pi(\boldsymbol{x}(t), t), t) dt + \ell_{\mathcal{T}}(\boldsymbol{x}(\mathcal{T})) \Big) \\ \boldsymbol{x}(t) &= \boldsymbol{x}. \end{aligned} \tag{3.2}$$

By splitting the integral of the running cost into two sub-intervals of size δt and $\mathcal{T} - (t + \delta t)$ and by taking the first-order approximation of the first integral, we end up with:

$$\begin{aligned} v(\boldsymbol{x}, t) &= \min_{\pi} \Big(\int_t^{t+\delta t} \ell(\boldsymbol{x}(t), \pi(\boldsymbol{x}(t), t)) dt + \int_{t+\delta t}^{\mathcal{T}} \ell(\boldsymbol{x}(t), \pi(\boldsymbol{x}(t), t)) dt \\ &\quad + \ell_{\mathcal{T}}(\boldsymbol{x}(\mathcal{T})) \Big) \\ &\approx \min_{\pi} \big(\ell(\boldsymbol{x}(t), \pi(\boldsymbol{x}, t)) \delta t + v(\boldsymbol{x}(t + \delta t), t + \delta t) \big). \end{aligned} \tag{3.3}$$

Now, the first-order approximation of the Value function writes:

$$v(\boldsymbol{x}(t + \delta t), t + \delta t) \approx v(\boldsymbol{x}(t), t) + f(\boldsymbol{x}(t), \boldsymbol{u}(t))^\top \boldsymbol{v}_{\boldsymbol{x}}(\boldsymbol{x}(t), t) \delta t + v_t(\boldsymbol{x}(t), t) \delta t. \tag{3.4}$$

Re-injecting in (3.3) and taking $\delta t \to 0$ results in the continuous Hamilton-Jacobi-Bellman equation:

$$\begin{aligned} -v_t(\boldsymbol{x}, t) &= \min_{\boldsymbol{u}} \Big(\ell_t(\boldsymbol{x}, \boldsymbol{u}) + f(\boldsymbol{x}, \boldsymbol{u})^\top \boldsymbol{v}_{\boldsymbol{x}}(\boldsymbol{x}, t) \Big) \\ \pi^*(\boldsymbol{x}) &= \operatorname*{argmin}_{\boldsymbol{u}} \Big(\ell_t(\boldsymbol{x}, \boldsymbol{u}) + f(\boldsymbol{x}, \boldsymbol{u})^\top \boldsymbol{v}_{\boldsymbol{x}}(\boldsymbol{x}, t) \Big). \end{aligned} \tag{3.5}$$

For high-dimension states and non-smooth costs, this equation is especially challenging to solve; moreover, it provides an optimal policy over the entire state space when it would be simpler to compute only an optimal control trajectory. Being too expensive to be tractable for real-time systems, this approach is rarely taken in robotics, unless it is combined with direct or indirect methods.

Indirect methods

Indirect methods involve the formulation of the Maximum Principle of Pontryaguin and the resolution of a boundary value problem based on the calculus of variations. The objective is to describe the local solutions of the Hamilton-Jacobi-Bellman equation by a set of differential equations subjected to initial and terminal constraints. The resulting problem is then numerically solved before being discretized.

Indirect methods classically start by defining the Hamiltonian of the dynamic system as:

$$H(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t) = \ell(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + \boldsymbol{\lambda}(t)^\top f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{3.6}$$

with $\underline{\boldsymbol{\lambda}} : t \mapsto \boldsymbol{\lambda}(t)$ the co-state trajectory. Pontryaguin's Maximum Principle then asserts that the optimal state, co-state and control trajectories verify the following optimality conditions:

$$\begin{aligned} \dot{\boldsymbol{x}}^* &= \frac{\partial H}{\partial \boldsymbol{\lambda}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*) \\ -\dot{\boldsymbol{\lambda}}^* &= \frac{\partial H}{\partial \boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*) \\ 0 &= \frac{\partial H}{\partial \boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*) \\ \boldsymbol{u}^* &= \underset{\boldsymbol{u}}{\operatorname{argmin}}\, H(\boldsymbol{x}^*, \boldsymbol{u}, \boldsymbol{\lambda}^*) \end{aligned} \tag{3.7}$$

with the boundary conditions:

$$\begin{aligned} \boldsymbol{x}^*(0) &= \boldsymbol{x}_0 \\ \boldsymbol{\lambda}^*(\mathcal{T}) &= \frac{\partial \ell_{\mathcal{T}}}{\partial \boldsymbol{x}}(\boldsymbol{x}(\mathcal{T})). \end{aligned} \tag{3.8}$$

The Hamiltonian formulation is closely linked to the Hamilton-Jacobi-Bellman approach as the co-state trajectory represents the Value function gradient. The solution of the two-point boundary problem is generally computed through shooting methods or collocation. Because the system remains highly nonlinear, the differential equations involved in the formulation can be very tough to handle, and any change in the way the control arcs are distributed across the horizon requires an entirely new problem setup, hard to warm-start.

Direct methods

As discussed before, direct methods have gain popular interest among the robotics community because they turn the continuous formulation into a NLP, a class of problems for which many powerful solvers exist, including ones that efficiently treat inequality constraints. Direct methods involve a finite dimensional parameterization of the control and sometimes of the state, depending on the transcription. They are particularly suited for low-dimensional problems but tend to feature heavy computational load in high dimension. Moreover, they do not exploit the sparsity of the dynamics and the temporal causality of the state and control trajectories.

Once transcripted, the resulting NLP can be solved with fast and efficient SQP frameworks like Acados [Verschueren et al. 2020] or MUSCOD [Diehl et al. 2007], or with constrained iterative solvers like interior-point methods [Wächter and Biegler 2006] or Augmented Lagrangian [Nocedal and Wright 2006a]. The last two approaches revolve around converting the inequality constraints into penalty terms inside the objective function, then solving a sequence of unconstrained optimization problems whose minima tend toward the desired solution.

3.1.1 Transcription of the continuous problem

Problem (3.1) is of infinite dimension and needs to be transcribed in order to become tractable by classical optimization tools. In the frame of this thesis, a direct transcription approach is implemented, meaning that we first discretize the continuous OCP to turn it into a nonlinear program then solve it through a multiple shooting method.

We thus divide the interval $[0, \mathcal{T}]$ into T sub-intervals of width δt, where the state $\boldsymbol{x}_t$ at the end of each sub-interval is computed as the forward integration of the dynamics given a constant control $\boldsymbol{u}_t$ over $[t, t+\delta t]$. Here we arbitrarily choose to represent the control over each sub-interval as a constant, but it can be described by a polynom if the dimension of the problem is not a limiting factor. Similarly, the discretization time step is considered constant in the following, but can be of varying size in some works of the literature. Abusing the notations, we end up with a discretized NLP of the form:

$$\begin{aligned} \min_{\underline{\boldsymbol{x}},\underline{\boldsymbol{u}}} & \Big(\sum_{t=0}^{T-1} \ell_t(\boldsymbol{x}_t, \boldsymbol{u}_t) + \ell_T(\boldsymbol{x}_T) \Big) \\ s.t. \quad & \boldsymbol{x}_0 = \hat{\boldsymbol{x}}_0 \\ & \forall t = 0..T-1, \quad \boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t), \end{aligned} \tag{3.9}$$

with $\hat{\boldsymbol{x}}_0$ the initial state of the system, and f now representing an integrated dynamics function. The optimization now takes place over a finite set of state and control variables, which can be written as discretized trajectories: $\underline{\boldsymbol{x}} = (\boldsymbol{x}_t)_{t=0..T}$ and $\underline{\boldsymbol{u}} = (\boldsymbol{u}_t)_{t=0..T-1}$, with $\boldsymbol{x}_t \in \mathbb{R}^{n_x}$ and $\boldsymbol{u}_t \in \mathbb{R}^{n_u}$.

3.1.2 The DDP algorithm

The resulting problem is typically nonlinear and non-convex due to the complexity of the dynamics f and costs ℓ. DDP will solve this NLP recursively through a two-step process involving a backward pass to compute the optimal flow from the terminal conditions and a forward pass to roll out the state and control trajectories. In a sense, the DDP formulation is the synthesis between a direct transcription and a dynamic programming approach, leveraging the best of both worlds.

Bellman's optimality principle in discrete case

DDP is based in essence on a discrete version of the Bellman's optimality principle, which states that any suffix of an optimal policy constitutes in itself an optimal policy. Put it another way, DDP solves a set of smaller and simpler unconstrained linear QP subproblems linked through a Riccati recursion.

Following this principle, we define the value function at time t to be the minimal possible cost-to-go starting from $\boldsymbol{x}$ at t:

$$\begin{aligned} v_t(\boldsymbol{x}) = \min_{\{\boldsymbol{u}_s\}_{s=t}^{T-1}} & \Big(\sum_{s=t}^{T-1} \ell_s(\boldsymbol{x}_s, \boldsymbol{u}_s) + \ell_T(\boldsymbol{x}_T) \Big) \\ s.t. \quad & \boldsymbol{x}_t = \boldsymbol{x} \\ & \forall s = t..T-1, \quad \boldsymbol{x}_{s+1} = f(\boldsymbol{x}_s, \boldsymbol{u}_s). \end{aligned} \tag{3.10}$$

Using this notation, the Bellman equation can be written as:

$$v_t(\boldsymbol{x}) = \min_{\boldsymbol{u}} \big(\ell_t(\boldsymbol{x}, \boldsymbol{u}) + v_{t+1}(f(\boldsymbol{x}, \boldsymbol{u}))\big). \tag{3.11}$$

In other words, the recursion of the value function turns a minimization over a sequence of controls into a minimization over a single control, knowing already what form takes the optimal cost-to-go starting at the next time step. This idea is the key component of Dynamic Programming. Likewise, the optimal policy at time t is simply:

$$\boldsymbol{u}_t^*(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{u}} \big(\ell_t(\boldsymbol{x}, \boldsymbol{u}) + v_{t+1}(f(\boldsymbol{x}, \boldsymbol{u}))\big). \tag{3.12}$$

Equation (3.11) illustrates that the optimal flow of the value function can be computed recursively inside a backward pass, starting from the terminal condition $v_T(\boldsymbol{x}) = \ell_T(\boldsymbol{x})$ and going toward the initial state $\boldsymbol{x}_0$. Once an optimal policy is computed across the whole horizon, the optimal step can be obtained by line search along this direction inside a forward pass, going from 0 to T.

Computing the backward recursion

As the problem is still nonlinear and non-convex, the DDP algorithm proceeds by computing a quadratic approximation of the value function and linear approximation of the policy in the backward pass. Given trajectory candidates $(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ and perturbations $(\underline{\boldsymbol{\Delta x}}, \underline{\boldsymbol{\Delta u}})$, we define the Q-value to be the resulting variation of the cost-to-go caused by the perturbation:

$$\begin{aligned} q_t(\boldsymbol{\Delta x}, \boldsymbol{\Delta u}) = \ell_t(\boldsymbol{x} + \boldsymbol{\Delta x}, \boldsymbol{u} + \boldsymbol{\Delta u}) + v_{t+1}(f(\boldsymbol{x} + \boldsymbol{\Delta x}, \boldsymbol{u} + \boldsymbol{\Delta u})) \\ - \big(\ell_t(\boldsymbol{x}, \boldsymbol{u}) + v_{t+1}(f(\boldsymbol{x}, \boldsymbol{u}))\big). \end{aligned} \tag{3.13}$$

Minimizing the Q-value with respect to $\boldsymbol{\Delta u}$ gives the optimal policy based on the state perturbation. By approximating the Q-value with a Taylor development at the second order of the costs and dynamics, we obtain:

$$\begin{aligned} \boldsymbol{\Delta u}^*(\boldsymbol{\Delta x}) &= \operatorname*{argmin}_{\boldsymbol{\Delta u}} q_t(\boldsymbol{\Delta x}, \boldsymbol{\Delta u}) \\ &\approx \operatorname*{argmin}_{\boldsymbol{\Delta u}} \frac{1}{2} \begin{bmatrix} 1 \\ \boldsymbol{\Delta x} \\ \boldsymbol{\Delta u} \end{bmatrix}^\top \begin{bmatrix} 0 & \boldsymbol{q}_{\boldsymbol{x},t}^\top & \boldsymbol{q}_{\boldsymbol{u},t}^\top \\ \boldsymbol{q}_{\boldsymbol{x},t} & \boldsymbol{Q}_{\boldsymbol{xx},t} & \boldsymbol{Q}_{\boldsymbol{xu},t} \\ \boldsymbol{q}_{\boldsymbol{x},t} & \boldsymbol{Q}_{\boldsymbol{ux},t} & \boldsymbol{Q}_{\boldsymbol{uu},t} \end{bmatrix} \begin{bmatrix} 1 \\ \boldsymbol{\Delta x} \\ \boldsymbol{\Delta u} \end{bmatrix} \\ &= -\boldsymbol{Q}_{\boldsymbol{uu},t}^{-1}(\boldsymbol{q}_{\boldsymbol{u},t} + \boldsymbol{Q}_{\boldsymbol{xu},t}\boldsymbol{\Delta x}), \end{aligned} \tag{3.14}$$

with the following derivatives terms, $\forall t = 0..T-1$:

$$\begin{aligned}
\boldsymbol{q}_{\boldsymbol{x},t} &= \boldsymbol{\ell}_{\boldsymbol{x},t} + \boldsymbol{F}_{\boldsymbol{x},t}^{\top} \boldsymbol{v}_{\boldsymbol{x},t+1} \\
\boldsymbol{q}_{\boldsymbol{u},t} &= \boldsymbol{\ell}_{\boldsymbol{u},t} + \boldsymbol{F}_{\boldsymbol{u},t}^{\top} \boldsymbol{v}_{\boldsymbol{x},t+1} \\
\boldsymbol{Q}_{\boldsymbol{xx},t} &= \boldsymbol{L}_{\boldsymbol{xx},t} + \boldsymbol{F}_{\boldsymbol{x},t}^{\top} \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{x},t} + \boldsymbol{v}_{\boldsymbol{x},t+1} \boldsymbol{F}_{\boldsymbol{xx},t} \\
\boldsymbol{Q}_{\boldsymbol{xu},t} &= \boldsymbol{L}_{\boldsymbol{xu}} + \boldsymbol{F}_{\boldsymbol{x},t}^{\top} \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{u},t} + \boldsymbol{v}_{\boldsymbol{x},t+1} \boldsymbol{F}_{\boldsymbol{xu},t} \\
\boldsymbol{Q}_{\boldsymbol{uu},t} &= \boldsymbol{L}_{\boldsymbol{uu}} + \boldsymbol{F}_{\boldsymbol{u},t}^{T} \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{u},t} + \boldsymbol{v}_{\boldsymbol{x},t+1} \boldsymbol{F}_{\boldsymbol{uu},t} \\
\boldsymbol{V}_{\boldsymbol{xx},t} &= \boldsymbol{Q}_{\boldsymbol{xx},t} + \boldsymbol{Q}_{\boldsymbol{xu},t} \boldsymbol{Q}_{\boldsymbol{uu},t}^{-1} \boldsymbol{Q}_{\boldsymbol{xu},t} \\
\boldsymbol{v}_{\boldsymbol{x},t} &= \boldsymbol{q}_{\boldsymbol{x},t} + \boldsymbol{Q}_{\boldsymbol{xu},t} \boldsymbol{Q}_{\boldsymbol{uu},t}^{-1} \boldsymbol{q}_{\boldsymbol{u},t},
\end{aligned} \tag{3.15}$$

and the following terminal conditions:

$$\begin{aligned}
\boldsymbol{V}_{\boldsymbol{xx},T} &= \boldsymbol{L}_{\boldsymbol{xx},T} \\
\boldsymbol{v}_{\boldsymbol{x},T} &= \boldsymbol{\ell}_{\boldsymbol{x},T}.
\end{aligned} \tag{3.16}$$

In the previous equations, $(\boldsymbol{L}_{\boldsymbol{xx}}, \boldsymbol{L}_{\boldsymbol{ux}}, \boldsymbol{L}_{\boldsymbol{xu}}, \boldsymbol{L}_{\boldsymbol{uu}})$ stand for the Hessians of the cost function whereas $(\boldsymbol{\ell}_{\boldsymbol{x}}, \boldsymbol{\ell}_{\boldsymbol{u}})$ represent its gradients. In the same way, $(\boldsymbol{F}_{\boldsymbol{x}}, \boldsymbol{F}_{\boldsymbol{u}}, \boldsymbol{F}_{\boldsymbol{xx}}, \boldsymbol{F}_{\boldsymbol{uu}}, \boldsymbol{F}_{\boldsymbol{xu}})$ are the dynamics derivatives with respect to state and control variables. It is worth to note that the Gauss-Newton approximation used in (3.14) becomes exact in case the problem has linear dynamics and quadratic costs. The DDP is then equivalent to a LQR scheme. In nonlinear cases however, this quadratic approximation constitutes one of the weak points of the algorithm.

Computing the forward pass

The computation of the backward pass results in a linear feedback policy of the form:

$$\begin{aligned}
\boldsymbol{\Delta u}_t^*(\boldsymbol{\Delta x}_t) &= \boldsymbol{k}_t + \boldsymbol{K}_t \boldsymbol{\Delta x}_t \\
\boldsymbol{k}_t &= -\boldsymbol{Q}_{\boldsymbol{uu},t}^{-1} \boldsymbol{q}_{\boldsymbol{u},t} \\
\boldsymbol{K}_t &= -\boldsymbol{Q}_{\boldsymbol{uu},t}^{-1} \boldsymbol{Q}_{\boldsymbol{ux},t}
\end{aligned} \tag{3.17}$$

From there, the forward pass can be performed on the linearized dynamics or nonlinear one, depending on the structure of the problem. Using the linearized dynamics turns the DDP into a Newton descent algorithm with strong convergence guarantees on convex problems, but at the price of higher computation cost since the matrix $(\boldsymbol{F}_{\boldsymbol{x}}, \boldsymbol{F}_{\boldsymbol{u}})$ are usually very large. Moreover, a merit function on the expected cost decrease has to be implemented in order to deal with the linear approximation of the rollout, and such a function is often non trivial to write. Conversely, using a nonlinear rollout cancels the need of a merit function and ensures that the state trajectory is feasible, meaning that there is no gap in the dynamics between shooting knots.

The DDP algorithm classically implements a nonlinear rollout combined with a line search scheme, whose role is to select the longest possible step along the descent direction so as to produce a significant cost decrease. Given an optimal feedback policy $(\underline{\boldsymbol{k}}, \underline{\boldsymbol{K}})$, initial trajectory $(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$, resulting trajectory $(\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*)$ and a line search parameter $0 \leq \alpha \leq 1$ initialized at 1, the forward pass writes:

$$\begin{aligned} \boldsymbol{x}_0^* &= \boldsymbol{x}_0 \\ \boldsymbol{u}_t^* &= \boldsymbol{u}_t + \alpha \boldsymbol{k}_t + \boldsymbol{K}_t(\boldsymbol{x}_t^* - \boldsymbol{x}_t) \\ \boldsymbol{x}_{t+1}^* &= f(\boldsymbol{x}^*, \boldsymbol{u}^*). \end{aligned} \tag{3.18}$$

Feasibility-prone DDP algorithm

An improved version of the DDP algorithm is proposed in [Mastalli et al. 2020], where the gaps in the dynamics are represented by extra variables at no added cost. This new algorithm benefits from two main improvements with respect to the standard version.

- First, it can be initialized with dynamically unfeasible trajectory guesses. In practice, DDP is warm-started with only a control trajectory, and a rollout is performed to compute the initial state trajectory. With feasibility-prone DDP, it is possible to warm-start the algorithm with both control and state trajectories, even if they do not match.
- Second, the new formulation does not close the gaps in the dynamics during the early steps of forward rollout, and as a consequence features better globalization capacity similar to multiple shooting schemes.

The algorithm operates by linearizing (3.9) in order to obtain a QP formulation:

$$\begin{aligned} & \min_{\underline{\boldsymbol{\Delta x}, \boldsymbol{\Delta u}}} \sum_{t=0}^{T-1} \tilde{\ell}_t(\boldsymbol{\Delta x}_t, \boldsymbol{\Delta u}_t) + \tilde{\ell}_T(\boldsymbol{\Delta x}_T) \\ s.t. \quad & \boldsymbol{\Delta x}_0 = \boldsymbol{f}_0 \\ & \tilde{\ell}_T(\boldsymbol{\Delta x}_T) = \frac{1}{2} \boldsymbol{\Delta x}_T^\top \boldsymbol{L}_{\boldsymbol{xx},T} \boldsymbol{\Delta x}_T + \boldsymbol{\ell}_{\boldsymbol{x},T} \boldsymbol{\Delta x}_T \\ & \forall t = 0..T-1, \\ & \boldsymbol{\Delta x}_{t+1} = \boldsymbol{F}_{\boldsymbol{x},t} \boldsymbol{\Delta x}_t + \boldsymbol{F}_{\boldsymbol{u},t} \boldsymbol{\Delta u}_t + \boldsymbol{f}_{t+1} \\ & \tilde{\ell}_t(\boldsymbol{\Delta x}_t, \boldsymbol{\Delta u}_t) = \frac{1}{2} \begin{bmatrix} \boldsymbol{\Delta x}_t^\top \\ \boldsymbol{\Delta u}_t^\top \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{L}_{\boldsymbol{xx},t} & \boldsymbol{L}_{\boldsymbol{xu},t} \\ \boldsymbol{L}_{\boldsymbol{ux},t} & \boldsymbol{L}_{\boldsymbol{uu},t} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta x}_t \\ \boldsymbol{\Delta u}_t \end{bmatrix} \\ & \qquad + \begin{bmatrix} \boldsymbol{\ell}_{\boldsymbol{x},t} & \boldsymbol{\ell}_{\boldsymbol{u},t} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta x}_t \\ \boldsymbol{\Delta u}_t \end{bmatrix}. \end{aligned} \tag{3.19}$$

In this formulation, the vectors $\boldsymbol{f}_t$ represent the gaps in the dynamics, that is to say, the difference between the rollout state and the shooting state: $\boldsymbol{f}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t) - \boldsymbol{x}_{t+1}$. It is worth to notice that the second-order derivatives of the dynamics are neglected so as to alleviate computational load during online execution, but this does not affect the generality of the method.

Since the problem to solve is now quadratic, it features constant Hessians and linear Jacobians with respect to decision variables. Instead of defining the Jacobian of the value function at the current shooting knot, we compute it at the state reached at the end of the previous sub-interval, so as to propagate the deviation due to gaps in dynamics:

$$\begin{aligned}\hat{\boldsymbol{v}}_{\boldsymbol{x},t} &= \boldsymbol{v}_{\boldsymbol{x},t} + \boldsymbol{V}_{\boldsymbol{xx},t}\boldsymbol{f}_t \\ &= \boldsymbol{q}_{\boldsymbol{x},t} - \boldsymbol{Q}_{\boldsymbol{xu},t}\boldsymbol{k_t} + \boldsymbol{V}_{\boldsymbol{xx},t}\boldsymbol{f}_t.\end{aligned} \tag{3.20}$$

Taking into account the simplified dynamics, the new derivatives inside the backward pass become, $\forall t = 0..T-1$:

$$\begin{aligned}\boldsymbol{q}_{\boldsymbol{x},t} &= \boldsymbol{\ell}_{\boldsymbol{x},t} + \boldsymbol{F}_{\boldsymbol{x},t}^{\top}\hat{\boldsymbol{v}}_{\boldsymbol{x},t+1} \\ \boldsymbol{q}_{\boldsymbol{u},t} &= \boldsymbol{\ell}_{\boldsymbol{u},t} + \boldsymbol{F}_{\boldsymbol{u},t}^{\top}\hat{\boldsymbol{v}}_{\boldsymbol{x},t+1} \\ \boldsymbol{Q}_{\boldsymbol{xx},t} &= \boldsymbol{L}_{\boldsymbol{xx},t} + \boldsymbol{F}_{\boldsymbol{x},t}^{T}\boldsymbol{V}_{\boldsymbol{xx},t+1}\boldsymbol{F}_{\boldsymbol{x},t} \\ \boldsymbol{Q}_{\boldsymbol{xu},t} &= \boldsymbol{L}_{\boldsymbol{xu},t} + \boldsymbol{F}_{\boldsymbol{x},t}^{T}\boldsymbol{V}_{\boldsymbol{xx},t+1}\boldsymbol{F}_{\boldsymbol{u},t} \\ \boldsymbol{Q}_{\boldsymbol{uu},t} &= \boldsymbol{L}_{\boldsymbol{uu},t} + \boldsymbol{F}_{\boldsymbol{u},t}^{T}\boldsymbol{V}_{\boldsymbol{xx},t+1}\boldsymbol{F}_{\boldsymbol{u},t}.\end{aligned} \tag{3.21}$$

Once a descent direction is computed, a modified nonlinear rollout is performed in order to keep the gaps open during the first iterations. Given a step length $0 \leq \alpha \leq 1$, the forward pass writes:

$$\begin{aligned}\boldsymbol{x}_0^* &= \boldsymbol{x}_0 - (1-\alpha)\boldsymbol{f}_0 \\ \boldsymbol{u}_t^* &= \boldsymbol{u}_t + \alpha\boldsymbol{k}_t + \boldsymbol{K}_t(\boldsymbol{x}_t^* - \boldsymbol{x}_t) \\ \boldsymbol{x}_{t+1}^* &= f(\boldsymbol{x}^*, \boldsymbol{u}^*) - (1-\alpha)\boldsymbol{f}_t.\end{aligned} \tag{3.22}$$

Note that a step of size $\alpha = 1$ brings the solver to a feasible solution, behaving like classical DDP. It is nonetheless interesting to take a smaller step in order to mimic the behavior of a multi shooting solver.

In summary, the FDDP algorithm efficiently solves optimal control problems by taking advantage of their intrinsic sparse structure and has better globalization properties than classical DDP. Main contributions of the total computational load result from the inversion of $\boldsymbol{Q}$ and $\boldsymbol{F}$ matrices inside the backward pass. The FDDP algorithm was used in the frame of this thesis to generate whole-body optimal trajectories for locomotion tasks. Moreover, the feedback gains produced by the backward pass, sometimes called Ricatti gains, were leveraged to synthetize a low-level feedback policy at no additional cost.

3.2 System modeling

Two classes of models are primarily used in the field of optimal control: centroidal model and whole-body model. The first one projects the complexity of the dynamics unto the CoM motion, while the second takes into account the full dynamics of every joints of the robot. Centroidal approaches are favored when computation load is required to be low; on the other hand, whole-body approaches, even if expensive to compute, allow to produce more complex motions while ensuring feasibility along the preview horizon.

These models have in common that they both deal with contact dynamics and positions of end effectors in contact. Before discussing models, we first clarify how to represent rigid body motion in robotics.

3.2.1 Body pose representation

The placement of a body in Cartesian space is described by a translation and an orientation, for a total of six DoF. Given an orthogonal frame and origin point, any translation can be represented by a vector $r \in \mathbb{R}^3$. As for orientation, roboticists commonly use the rotation Lie group $\mathbf{SO(3)}$, defined by:

$$\mathbf{SO(3)} = \left\{ \boldsymbol{R} \mid \boldsymbol{R} \in \mathbb{R}^{3\times 3}, \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = I, |\boldsymbol{R}| = 1 \right\}. \tag{3.23}$$

The rotation group is a 3-dimensional manifold with differentiable structure. To any rotation is associated a unit vector $\bar{\boldsymbol{u}} = (\bar{u}_1, \bar{u}_2, \bar{u}_3) \in \mathbb{R}^3$ and an angle $\theta \in \mathbb{R}$ so that:

$$\begin{aligned} \boldsymbol{R} &= \exp(\theta \bar{\boldsymbol{u}}\times) \\ s.t. \quad \bar{\boldsymbol{u}}\times &= \begin{bmatrix} 0 & -\bar{u}_3 & \bar{u}_2 \\ \bar{u}_3 & 0 - \bar{u}_1 & \\ -\bar{u}_2 & \bar{u}_1 & 0 \end{bmatrix} \\ \exp(\boldsymbol{X}) &= \sum_{i=0}^{\infty} \frac{\boldsymbol{X}^i}{i!}. \end{aligned} \tag{3.24}$$

Equivalently, it is possible to retrieve the unit vector $\bar{\boldsymbol{u}}$ and angle θ from a rotation $\boldsymbol{R}$ thanks to the logarithmic function :

$$\begin{aligned} \theta \bar{\boldsymbol{u}}\times &= \log(\boldsymbol{R}) \\ &= \frac{\theta}{2\sin(\theta)}(\boldsymbol{R}^\top - \boldsymbol{R}) \\ s.t. \quad 1 + 2\cos(\theta) &= tr(\boldsymbol{R}), \theta \neq 0, -\pi < \theta < \pi. \end{aligned} \tag{3.25}$$

The matrix $\bar{\boldsymbol{u}}\times$ belongs to the tangent space of $\mathbf{SO(3)}$ at the identity, which forms the Lie algebra of skew-symmetric tensors, defined as:

$$\mathbf{so(3)} = \mathbf{T_I SO(3)} = \left\{ \boldsymbol{A} \mid \boldsymbol{A} \in \mathbb{R}^{3\times 3}, \boldsymbol{A}^\top = -\boldsymbol{A} \right\}. \tag{3.26}$$

Elements of the Lie algebra $\mathbf{so(3)}$ can be seen as infinitesimal rotations. Equations (3.24) and (3.25) show that $\mathbf{so(3)}$ and $\mathbf{SO(3)}$ are tightly related through exponential and logarithmic mappings.

In order to formulate geometric tasks involving a desired goal orientation, it is necessary to choose a standardized metric over $\mathbf{SO(3)}$. A widely used metric leverages the logarithm of rotation matrix. For practical computation, it is faster to define a compact logarithm function that returns an element of $\mathbb{R}^3$ rather than of $\mathbf{so(3)}$:

$$\begin{aligned} \log(\boldsymbol{R}) &= \frac{\theta}{2\sin(\theta)} \begin{bmatrix} R_{2,1} - R_{1,2} \\ R_{0,2} - R_{2,0} \\ R_{1,0} - R_{0,1} \end{bmatrix} \\ s.t. \quad 1 + 2\cos(\theta) &= tr(\boldsymbol{R}), \theta \neq 0, -\pi < \theta < \pi. \end{aligned} \tag{3.27}$$

We can now define a proper metric to measure the distance between two rotations $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$:

$$\ell^R(\boldsymbol{R}_1, \boldsymbol{R}_2) = ||\log(\boldsymbol{R}_2^\top \boldsymbol{R}_1)||. \tag{3.28}$$

Likewise, the Lie group $\mathbf{SE}(3)$ is an usual representation for rigid body motion in robotics:

$$\mathbf{SE}(3) = \left\{ \boldsymbol{P} \mid \boldsymbol{P} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{r} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, \boldsymbol{R} \in \mathbf{SO}(3), \boldsymbol{r} \in \mathbb{R}^3 \right\}. \tag{3.29}$$

The Lie algebra of $\mathbf{SE}(3)$ is denoted $\mathbf{se}(3)$ and describes velocity motion:

$$\mathbf{se}(3) = \left\{ \boldsymbol{S} \mid \boldsymbol{S} = \begin{bmatrix} \bar{\boldsymbol{u}}\times & \boldsymbol{v} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix}, \bar{\boldsymbol{u}}\times \in \mathbf{so}(3), \boldsymbol{v} \in \mathbb{R}^3 \right\}. \tag{3.30}$$

Again, for the sake of conciseness, elements of $\mathbf{se}(3)$ are practically represented as 6-dimensional vector known as spatial velocity. This vector is the concatenation of a linear velocity and an angular velocity expressed in Plücker coordinates (see [Featherstone 2008]). This notation allows to define a simple metric on the velocity of an end effector.

3.2.2 Contacts as holonomic constraints

Two main approaches exist to describe contact interaction in robotics: visco-elastic representation and rigid body representation. The first one considers that colliding surfaces are deformable and allows to define a relationship between stress and strain acting at the contact point. In practice, a non-linear string and damper model is often used to represent this relationship [Marhefka and Orin 1999]. Unfortunately, the visco-elastic approach needs high stiffness gains to accurately describe contact physics, leading to poor numerical conditioning of the OCP.

On the other hand, the rigid body representation models contacts as springs with infinite stiffness, resulting in impact events at wich forces and velocities are discontinuous. Despite this downside, rigid body representation is widely used in robotics and leads to hybrid dynamics schemes where contact modes and transitions are predefined by the user, or given by a high-level planner. In the frame of this thesis, we have chosen to represent contacts with rigid bodies rather than soft ones.

Contacts between two rigid bodies obey the principles of unilaterality and non-penetration. If the bodies do not slide with respect to each other, the exerted forces lay inside the friction constraint cone, giving rise to an inequality constraint on force. The constraint is said to be holonomic as it can be written in the form $\phi(\boldsymbol{q}) = 0$ with $\phi(\boldsymbol{q})$ signed distance function between bodies and $\boldsymbol{J} = \frac{\partial \phi}{\partial \boldsymbol{q}}$ the Jacobian of contact.

In the following section, we examine the case of an end effector coming in contact with a planar ground, in the classical Cartesian world frame. We denote $z \in \mathbb{R}$ the coordinate of the end effector along the normal to the ground, μ a friction parameter depending on materials in contact and $\boldsymbol{f}^c = (f_x^c, f_y^c, f_z^c) \in \mathbb{R}^3$ the force that the end effector exerts on the ground. Then, the non-penetration constraint can be written as a complementary condition of the form:

$$\ddot{z} \geq 0, f_z^c = 0 \quad \textbf{or} \quad \ddot{z} = 0, f_z^c \geq 0. \tag{3.31}$$

If the vertical acceleration of the end effector is strictly positive, then the contact is about to be broken and no normal force is applied; on the contrary, a non-null force

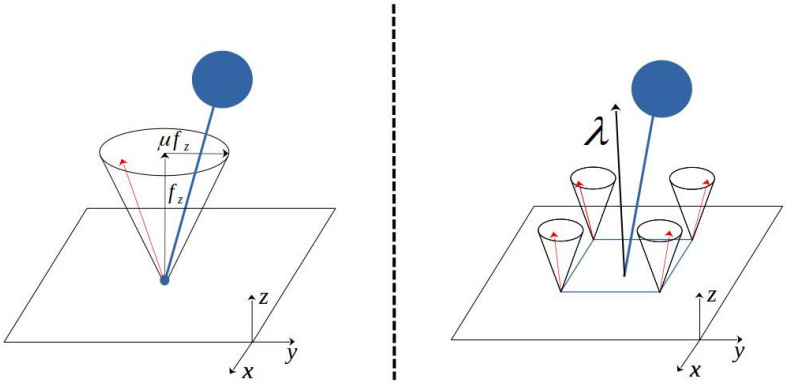

FIGURE 3.1: Left: modeling a 3D punctual contact. Right: modeling a 6D planar contact. The red arrows represent 3D forces.

can only be applied if the vertical acceleration is null. Lastly, the vertical acceleration cannot be negative as it would mean the end effector will penetrate the ground in the immediate future. In the same way, the end effector cannot pull on the ground, so the exerted force must stay positive.

3D contacts

Let us examine the case of a punctual end effector in contact with the ground (see Fig. 3.1, left). In this simple case, the system is applying a 3D force on a precise point, and in case of non-slippage this force is bound by the following constraints:

$$f_z^c \geq 0 \tag{3.32a}$$
$$||\boldsymbol{f}_{x,y}^c|| \leq \mu f_z^c. \tag{3.32b}$$

Equations (3.32a) and (3.32b) represent respectively the unilaterality condition and non-slippage condition. As the non-slippage condition is nonlinear, it is usual in robotics to consider a linear approximation based on a quadratic cone:

$$|f_x^c|, |f_y^c| \leq \mu f_z^c. \tag{3.33}$$

We thus stick to this approximation in the following sections.

6D contacts

In case the robot has rectangular feet, contacts are usually not limited to one point but rather are made across all the area under the foot. As it is not possible in practice to reconstruct the resulting forces by integrating the pressure and stress fields under the foot, rectangular contacts are classically modeled by considering that each vertice of the foot applies on the ground a 3D force laying inside a friction cone. The sum of all these forces reported to the center of the foot link results in a contact wrench $\boldsymbol{\lambda} = (\boldsymbol{f}^c, \boldsymbol{\tau}) \in \mathbb{R}^6$, concatenation of linear and angular forces (see Fig. 3.1, right). It is then possible to write the equivalent contact wrench constraints as [Caron et al. 2015b]:

$$f_z^c > 0 \tag{3.34a}$$
$$|f_x^c|, |f_y^c| \leq \mu f_z^c \tag{3.34b}$$
$$|\tau_x| \leq W f_z^c \tag{3.34c}$$
$$|\tau_y| \leq L f_z^c \tag{3.34d}$$
$$\tau_{min} \leq \tau_z \leq \tau_{max}, \tag{3.34e}$$

with W, L respectively the half width and half length of the foot, and :

$$\tau_{min} = -\mu(W+L)f_z^c + |W f_x^c - \mu\tau_x| + |L f_y^c - \mu\tau_y|$$
$$\tau_{max} = \mu(W+L)f_z^c - |W f_x^c + \mu\tau_x| - |L f_y^c + \mu\tau_y|.$$

Equations (3.34a) and (3.34b) describe the same unilaterality and non-slippage conditions as in the 3D case. Equations (3.34c) and (3.34d) represent a non-tilting condition linked to the stability bounds of the local center of pressure. Finally, (3.34e) sets bounds over the admissible yaw torque so that the feet does not rotate around its vertical axis.

3.2.3 Centroidal model

Contact forces determine the rate of change of linear centroidal momentum in legged locomotion, whereas whole-body motions affect the rate of change of angular momentum. Let us define m_r, the total mass of our multi-body system, $\boldsymbol{c} \in \mathbb{R}^3$ its center of mass and $\boldsymbol{L} \in \mathbb{R}^3$ its angular momentum. We assume that the number of end effectors in contact with the ground is n_c, with $(\boldsymbol{r}_i)_{i=1}^{n_c}$ their 3-D respective position. The under-actuated dynamics of this system is classically written as:

$$m_r \ddot{\boldsymbol{c}} = \sum_{i=1}^{n_c} \boldsymbol{f}_i - m_r \boldsymbol{g} \tag{3.35}$$
$$\dot{\boldsymbol{L}} = \sum_{i=1}^{n_c} (\boldsymbol{r}_i - \boldsymbol{c}) \times \boldsymbol{f}_i + \boldsymbol{\tau}_i, \tag{3.36}$$

with $(\boldsymbol{f}_i, \boldsymbol{\tau}_i) \in \mathbb{R}^6$ the unilateral contact wrench acting on the robot at contact i. As the name suggests, this dynamics is under-actuated because no command can act directly on $\boldsymbol{c}$, only contact forces. Expressing the resulting contact wrench at the origin of the world frame gives:

$$\begin{bmatrix} \boldsymbol{f}^c \\ \boldsymbol{\tau}^c \end{bmatrix} := \begin{bmatrix} \sum_{i=1}^{n_c} \boldsymbol{f}_i \\ \sum_{i=1}^{n_c} \boldsymbol{r}_i \times \boldsymbol{f}_i + \boldsymbol{\tau}_i \end{bmatrix}. \tag{3.37}$$

Instead of focusing on CoM dynamics, a better option to guarantee dynamic stability is to examine the ZMP dynamics, which coincides with the Center of Pressure (CoP) [Sardain and Bessonnet 2004] in flat ground conditions. The ZMP criterion typically makes the connection between the inertia of all body segments of the robot and the resulting contact forces and torques at the limb's extremities. Assuming the normal to the ground, $\boldsymbol{n}$, is aligned with the gravity vector $\boldsymbol{g}$, the ZMP is defined as the point on the ground where both horizontal components of the total contact moment are null. To put it another way, the moment resulting from the contact forces is parallel to $\boldsymbol{n}$ when expressed at the ZMP.

We then define $\boldsymbol{M}_z^c$ to be the moment of the contact forces expressed at the ZMP and $\boldsymbol{z}^p = (z^x, z^y, 0)^\top$ the ZMP coordinates in world frame. By expressing $\boldsymbol{M}_z^c$ at the origin and using the ZMP definition, we have:

$$\begin{aligned} \boldsymbol{M}_z^c \times \boldsymbol{n} &= (\boldsymbol{\tau}^c - \boldsymbol{z}^p \times \boldsymbol{f}^c) \times \boldsymbol{n} \\ &= \begin{bmatrix} \tau_x^c - z^y \cdot f_z^c \\ \tau_y^c + z^x \cdot f_z^c \\ 0 \end{bmatrix} \\ &= 0. \end{aligned} \tag{3.38}$$

Thus the ZMP coordinates writes:

$$\boldsymbol{z}^p = \begin{bmatrix} -\frac{\tau_y^c}{f_z^c} \\ \frac{\tau_x^c}{f_z^c} \\ 0 \end{bmatrix} \tag{3.39}$$

On the robot Pyrène, two 6-D force sensors located close to the ankles measure the left and right wrenches applied on the system by contact interactions. When both feet are in contact, we first compute the local center of pressure of each foot, then deduce the global center of pressure:

$$\boldsymbol{z}^p = \frac{\sum_{i=1}^2 \boldsymbol{R}_i \cdot \boldsymbol{z}_i^p + \boldsymbol{r}_i}{\sum_{i=1}^2 \boldsymbol{f}_i}, \tag{3.40}$$

where $\boldsymbol{R}_i \in SO(3)$ is the rotation of foot i and $\boldsymbol{z}_i^p$ its local center of pressure.

The ZMP is defined only inside the convex hull of the support polygon, and in general should not come close to the edges of its support in order to avoid foot tilting and ensure the robustness of the locomotion. It is usual in robotics to define tight security margins for the ZMP, so that it remains close to the center of the support polygon.

3.2.4 Whole-body model

Given a free-flyer system with n_j actuated joints and n_p contacts with the environment, we define $\boldsymbol{q} \in \mathcal{Q} := SE(3) \times \mathbb{R}^{n_j}$ to be the configuration vector of size n_q, accounting for the placement of the free-flyer joint in $SE(3)$ and the concatenation of all angular joint positions. Consequently, $\dot{\boldsymbol{q}} \in \bar{\mathcal{Q}} := se(3) \times \mathbb{R}^{n_j}$ is the velocity vector of size n_v laying in the tangent space of $\mathcal{Q}$, and $\ddot{\boldsymbol{q}}$ of size n_v is the acceleration vector. We then classically define the state of the robot to be $\boldsymbol{x} = (\boldsymbol{q}^\top, \dot{\boldsymbol{q}}^\top)^\top \in \mathcal{X}$ of size n_x and the control to be $\boldsymbol{u} = \boldsymbol{\tau} \in \mathbb{R}^{n_j}$, the vector of joint torques.

Similarly to [Featherstone 2008], we consider the dynamics of the multi-body system to be the result of the Euler-Lagrange equations:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{S}^\top \boldsymbol{\tau} + \sum_{i=1}^{n_p} \boldsymbol{J}_i(\boldsymbol{q})^\top \boldsymbol{\lambda}_i. \tag{3.41}$$

In (3.41), $\boldsymbol{M} \in \mathbb{R}^{n_v \times n_v}$ is the joint-space inertia matrix and $\boldsymbol{b} \in \mathbb{R}^{n_v}$ the generalized nonlinear forces accounting for centrifugal, Coriolis, and gravitational terms. For $i = 1..n_p$, $\boldsymbol{\lambda}_i \in \mathbb{R}^6$ and $\boldsymbol{J}_i \in \mathbb{R}^{6 \times n_v}$ stand for the contact force and contact Jacobian at contact i. The joint torques $\boldsymbol{\tau}$ are mapped to the actuated part of the dynamics thanks to the actuation matrix $\boldsymbol{S} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I}_{n_j} \end{bmatrix} \in \mathbb{R}^{n_j \times n_v}$. Similarly to the centroidal

model case, the dynamics is under-actuated because the command (here, the torque vector) cannot act directly on the free-flyer joint acceleration, due to the six first lines of $\boldsymbol{S}^\top$ being null.

3.3 Constrained multi-contact dynamics

Given an end effector i in contact with the ground, we denote $\boldsymbol{p} \in SE(3)$ its placement (position and orientation), $\boldsymbol{v} = \dot{\boldsymbol{p}} \in se(3)$ its spatial velocity and $\boldsymbol{J}_i \in \mathbb{R}^{6\times n_v}$ its contact Jacobian. Assuming non-slippage condition, the acceleration of the end effector is null:

$$\dot{\boldsymbol{v}} = \frac{\partial \boldsymbol{J}_i \dot{\boldsymbol{q}}}{\partial t} = \dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}} + \boldsymbol{J}_i \ddot{\boldsymbol{q}} = 0. \tag{3.42}$$

By combining this equality with the multi-body dynamics equation (3.41), we obtain the Karush-Kuhn-Tucker (KKT) conditions of the rigid contact dynamics:

$$\begin{bmatrix} \boldsymbol{M} & \boldsymbol{J}_c^\top \\ \boldsymbol{J}_c & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{S}^\top \boldsymbol{\tau} - \boldsymbol{b} \\ -\dot{\boldsymbol{J}}_c \dot{\boldsymbol{q}} \end{bmatrix}, \tag{3.43}$$

where $\boldsymbol{J}_c = (\boldsymbol{J}_1^\top \cdots \boldsymbol{J}_{n_p}^\top)^\top \in \mathbb{R}^{6n_p \times n_v}$ represents the concatenation of contact Jacobian matrices and $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1^\top \cdots \boldsymbol{\lambda}_{n_p}^\top)^\top \in \mathbb{R}^{6n_p}$ represents the concatenation of all contact wrenches.

Equation (3.43) can be interpreted as an optimal condition for the primal and dual trajectories $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\lambda}$ to be solution of the following convex optimization problem [Budhiraja et al. 2018]:

$$\begin{aligned} \ddot{\boldsymbol{q}}^* = \underset{\ddot{\boldsymbol{q}}}{\operatorname{argmin}} \frac{1}{2} ||\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_{free}||_{\boldsymbol{M}} \\ s.t. \quad \dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}} + \boldsymbol{J}_i \ddot{\boldsymbol{q}} = 0. \end{aligned} \tag{3.44}$$

This equation is obtained by applying Gauss's principle of least constraint to the unconstrained dynamics typically written as $\boldsymbol{M}\ddot{\boldsymbol{q}}_{free} = \boldsymbol{\tau}_g$, with $\boldsymbol{\tau}_g$ sum of the actuation and generalized forces. It has a unique solution only with $\boldsymbol{J}_c$ is full rank, which is not the case when considering more than one contact.

The formulation presented in (3.43) has two advantages: first, it integrates the contact constraint at the level of the dynamics, whereas most control schemes in the literature tackle it inside the optimization process. Second, it allows to express the concatenated contact forces as a function of state and control:

$$\begin{bmatrix} \ddot{\boldsymbol{q}} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{M} & \boldsymbol{J}_c^\top \\ \boldsymbol{J}_c & \boldsymbol{0} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{S}^\top \boldsymbol{\tau} - \boldsymbol{b} \\ -\dot{\boldsymbol{J}}_c \dot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}) \\ -\boldsymbol{g}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}) \end{bmatrix}. \tag{3.45}$$

Using clever Cholesky decomposition, the blockwise KKT matrix can be efficiently inverted during the integration of the dynamics. In case the KKT matrix is not invertible, for example when $\boldsymbol{J}_c$ is not full rank, a damped Cholesky decomposition can be used in order to approximately solve Gauss's principle of least action.

More details are available in [Mastalli et al. 2020].

All in all, (3.43) can finally be written under a discrete, integrated and force-free dynamics formulation:

$$\begin{aligned}
\boldsymbol{x}_{i+1} &= f(\boldsymbol{x}_i, \boldsymbol{u}_i) \\
\boldsymbol{\lambda}_i &= g(\boldsymbol{x}_i, \boldsymbol{u}_i) \\
\boldsymbol{F}_{\boldsymbol{x}} &= -\begin{bmatrix} \boldsymbol{M} & \boldsymbol{J}_c^\top \\ \boldsymbol{J}_c & \boldsymbol{0} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \boldsymbol{\tau}_g}{\partial \boldsymbol{x}} \\ \frac{\partial \boldsymbol{a}_0}{\partial \boldsymbol{x}} \end{bmatrix} \\
\boldsymbol{F}_{\boldsymbol{u}} &= -\begin{bmatrix} \boldsymbol{M} & \boldsymbol{J}_c^\top \\ \boldsymbol{J}_c & \boldsymbol{0} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \boldsymbol{\tau}_g}{\partial \boldsymbol{u}} \\ \frac{\partial \boldsymbol{a}_0}{\partial \boldsymbol{u}}, \end{bmatrix}
\end{aligned} \tag{3.46}$$

with $\boldsymbol{a}_0 = \dot{\boldsymbol{J}}_c \dot{\boldsymbol{q}}$ desired acceleration in the constrained space, $\frac{\partial \boldsymbol{\tau}_g}{\partial \boldsymbol{x}}$, $\frac{\partial \boldsymbol{\tau}_g}{\partial \boldsymbol{u}}$ derivatives of the classical Recursive Newton-Euler Algorithm and $\frac{\partial \boldsymbol{a}_0}{\partial \boldsymbol{x}}$, $\frac{\partial \boldsymbol{a}_0}{\partial \boldsymbol{u}}$ kinematics derivatives of the frame acceleration [Carpentier and Mansard 2018a].

3.4 FDDP formulation with contacts

Compared to the classical DDP implementation, our formulation includes contact forces inside the problem, which then writes [Budhiraja et al. 2018]:

$$\begin{aligned}
\min_{\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}}} \Big(\sum_{t=0}^{T-1} \ell_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\lambda}_t) + \ell_T(\boldsymbol{x}_T) \Big) \\
s.t. \quad \boldsymbol{x}_0 = \hat{\boldsymbol{x}}_0 \\
\forall t = 0..T-1, \quad \boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t), \\
\boldsymbol{\lambda}_t = g(\boldsymbol{x}_t, \boldsymbol{u}_t).
\end{aligned} \tag{3.47}$$

With such a formulation, the Gauss-Newton approximation of the $\boldsymbol{Q}$ coefficients become, $\forall t = 0..T-1$:

$$\begin{aligned}
\boldsymbol{q}_{\boldsymbol{x},t} &= \boldsymbol{\ell}_{\boldsymbol{x},t} + \boldsymbol{G}_{\boldsymbol{x},t}^\top \boldsymbol{\ell}_{\boldsymbol{\lambda},t} + \boldsymbol{F}_{\boldsymbol{x},t}^\top \hat{\boldsymbol{v}}_{\boldsymbol{x},t+1} \\
\boldsymbol{q}_{\boldsymbol{u},t} &= \boldsymbol{\ell}_{\boldsymbol{u},t} + \boldsymbol{G}_{\boldsymbol{u},t}^\top \boldsymbol{\ell}_{\boldsymbol{\lambda},t} + \boldsymbol{F}_{\boldsymbol{u},t}^\top \hat{\boldsymbol{v}}_{\boldsymbol{x},t+1} \\
\boldsymbol{Q}_{\boldsymbol{xx},t} &= \boldsymbol{L}_{\boldsymbol{xx},t} + \boldsymbol{G}_{\boldsymbol{x},t}^\top \boldsymbol{L}_{\boldsymbol{\lambda\lambda},t} \boldsymbol{G}_{\boldsymbol{x},t} + \boldsymbol{F}_{\boldsymbol{x},t}^T \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{x},t} \\
\boldsymbol{Q}_{\boldsymbol{xu},t} &= \boldsymbol{L}_{\boldsymbol{xu},t} + \boldsymbol{G}_{\boldsymbol{x},t}^\top \boldsymbol{L}_{\boldsymbol{\lambda\lambda},t} \boldsymbol{G}_{\boldsymbol{u},t} + \boldsymbol{F}_{\boldsymbol{x},t}^T \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{u},t} \\
\boldsymbol{Q}_{\boldsymbol{uu},t} &= \boldsymbol{L}_{\boldsymbol{uu},t} + \boldsymbol{G}_{\boldsymbol{u},t}^\top \boldsymbol{L}_{\boldsymbol{\lambda\lambda},t} \boldsymbol{G}_{\boldsymbol{u},t} + \boldsymbol{F}_{\boldsymbol{u},t}^T \boldsymbol{V}_{\boldsymbol{xx},t+1} \boldsymbol{F}_{\boldsymbol{u},t}.
\end{aligned} \tag{3.48}$$

The backward pass is then followed by a nonlinear forward pass described by (3.46). In this approach, contact forces are interpreted as hidden variables resulting from optimal state and control trajectories.

3.5 The Crocoddyl library

Our FDDP solver is implemented with the optimal control library Crocoddyl [Mastalli et al. 2020] and use the library Pinocchio [Carpentier et al. 2019] to enable fast computations of costs, dynamics and their derivatives.

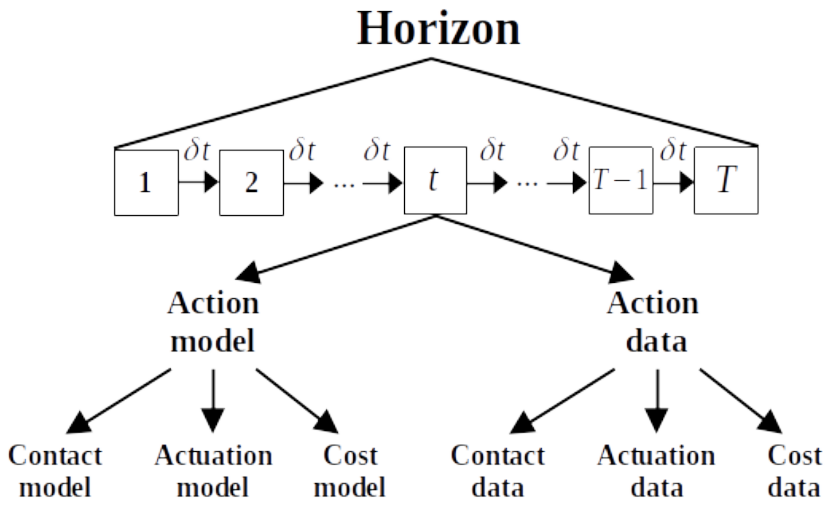

FIGURE 3.2: Structure of an OCP problem implemented in the Crocoddyl framework

Crocoddyl represents the OCP horizon by a set of T successive knots separated by a timestep δt (see Fig. 3.2). Each knot contains an invariant action model and its corresponding action data. The action model of knot t typically encompasses a contact model that defines the dynamics f at time $t \cdot \delta t$ in the future, an actuation model that describes the relationship between control and effective joint torque, and a cost model that contains the cost function ℓ_t. All corresponding numerical values and derivatives are stored inside the related action data. The last action model, also called terminal model, is not actuated and only contains the final cost model ℓ_T.

Contact and cost models can be modified on the fly to match the user's needs. In particular, it is cheap and easy to disable or enable a precise cost according to new sensor measures or high-level commands. The horizon paradigm then allows to plan in advance a change in the problem formulation, by updating the corresponding action models starting from the end of the preview window and going toward the first knot.

Crocoddyl implements analytical and sparse derivative schemes via Pinocchio. A multi threading approach is taken to compute these derivatives and further reduce the computational load. All in all, Crocoddyl is a fast, flexible and efficient optimal control framework to generate multi-contact trajectories on fullscale robots.

3.6 Conclusion

In this section were presented essential mathematical tools that will be leveraged in the following parts of this thesis to perform optimal control on a real humanoid robot. In particular, we have thoroughly described the principles of the feasibility-prone DDP algorithm and have showed how to combine it with a clever formulation of the dynamics which directly takes care of the contact constraints. In our opinion, DDP is one of the most relevant optimization algorithms to solve a whole-body predictive control problem for three main reasons:

- it exploits the sparsity of the dynamics to perform a very efficient KKT matrix inversion;

- it can be warm-started easily with infeasible trajectories, and produce feasible solutions in only one iteration;
- it computes a set of state feedback gains at no additional cost.

The main drawbacks of our DDP implementation include its inability to handle inequality and equality constraints, forcing the user to implement quadratic-barrier functions to take them into account. Moreover, since we neglect the Hessian of the dynamics, we do not benefit from a quadratic convergence of the algorithm.

Additionally, we have presented in detail how to model centroidal and whole-body dynamics, with a focus on ZMP computation. If the centroidal approach will not be used in the frame of this thesis, it remains useful to keep the formulation in mind in order to better understand how contact forces affect CoM motions. On the other hand, the ZMP is an essential criterion to ensure dynamic stability: as a consequence, this criterion would be thoroughly examined when discussing locomotion experiments.

Chapter 4

A High-Frequency Whole-Body MPC scheme

In this chapter, a predictive control scheme based on the full dynamics of the system is introduced and tested on the real platform Pyrène. The scheme leverages a first-order approximation of the optimal policy to bridge the frequency gap between the high-level MPC and the low-level control of the robot. Additionally, it provides mathematical hindsight on how to extract useful sensibilities from the DDP formulation in order to adapt the control to a fast variation of a high-level parameter. The results contained in this chapter were published in [Dantec et al. 2022a].

4.1 Motivation

The lack of computational power on mobile robots is a well-known challenge when it comes to implementing a real-time MPC scheme to perform complex motions. Currently the best solvers are barely able to reach 100 Hz for computing the control of a whole-body legged model, while modern robots are expecting new torque references in less than 1 ms. This frequency discrepancy gives rise to control latency and instability in the MPC scheme, leading to failures on real robots. Such an issue is usually tackled by using a handcrafted low-level tracking control whose inputs are the low-frequency trajectory computed by the MPC, but this approach adds another layer of complexity which should be carefully fine-tuned.

The DDP framework introduced in the previous chapter provides a good solution for whole-body control problems, although it cannot reach the low-level frequency of torque-controlled robots. When closing the loop on state measurements, the optimal control comes with a latency that can trigger instabilities if the sensitivity of the problem to initial state is significant. On the other hand, it is known that the control solution of a constraint LQR is a continuous piece-wise affine function of the initial state [Bemporad et al. 2002]. Following these observations, we propose to deduce the optimal feedback policy from the OCP formulation and we show the connection between this policy and the DDP-induced Riccati gains. The important step is to demonstrate that Riccati gains can be interpreted as the sensitivities of the optimal solution with respect to the initial state of the OCP, and as such can be used to interpolate the optimal control between two DDP computations. While we are only interested in getting the derivative of the initial control with respect to any external parameter, our proposed derivation leads to a more efficient backward pass

to compute the corresponding gain, and elegantly connects it with the Riccati matrix. This is necessary as we are proposing to use it for high-frequency control. For small state changes, this is equivalent to computing a new solution of the MPC at a much higher frequency, reaching the expected computation of the low- level control. The same approach can be generalized to any parameter of the OCP, leading to a set of Riccati-like feedback gains which can be computed at low cost by extending the classical DDP backward pass.

4.2 Sensitivity analysis of the DDP backward pass

Suppose that our OCP includes a parameter $\boldsymbol{\theta}$ that depends on time. If the variation of $\boldsymbol{\theta}$ is non-negligible over the computation duration of the OCP, then the optimal solution is no longer correct since it has been computed over an outdated value of $\boldsymbol{\theta}$. Determining the sensitivity of the OCP with respect to $\boldsymbol{\theta}$ will allow us to approximate a control correction proportional to the variation of the parameter. Such approach, where the optimization problem is differentiated, has already been studied for contact simulation [Le Lidec et al. 2021] and neural network applied to QP [Amos and Kolter 2017].

4.2.1 Sensitivity as partial derivative of the OCP

Let us view the discretized OCP described by (3.9) as a classical nonlinear problem parametrized by a given $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$. For an horizon of $T > 0$ knots, we define the optimal state and control trajectory to be $\underline{\boldsymbol{x}}^* = (\boldsymbol{x}_t^*)_{t=0..T}$ and $\underline{\boldsymbol{u}}^* = (\boldsymbol{u}_t^*)_{t=0..T-1}$. For the sake of simplicity we denote by $\boldsymbol{z}^* = (\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*) \in \mathbb{R}^{n_z}$ with $n_z = Tn_u + (T+1)n_x$ solution of the following discretized problem:

$$\begin{aligned} \mathcal{P}(\boldsymbol{\theta}) = \boldsymbol{z}^* = \underset{\boldsymbol{z}}{\text{argmin}} \;\; & \mathcal{J}(\boldsymbol{z}, \boldsymbol{\theta}) \\ s.t. \qquad & \boldsymbol{h}(\boldsymbol{z}, \boldsymbol{\theta}) = 0, \end{aligned} \tag{4.1}$$

with $\mathcal{J} : \mathbb{R}^{n_z} \times \mathbb{R}^{n_\theta} \to \mathbb{R}$ the total cost function and $\boldsymbol{h} : \mathbb{R}^{n_z} \times \mathbb{R}^{n_\theta} \to \mathbb{R}^{(T+1)\cdot n_x}$ the $T+1$ discretized dynamics constraints given by $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t)\ \forall t = 0..T-1$ and $\boldsymbol{x}_0 = \boldsymbol{f}_0$.

In order to be able to apply the KKT optimality conditions, we must assume some regularity qualifications under which a constrained minimizer also satisfies the KKT conditions. We now assume that $\mathcal{J}$ and $\boldsymbol{h}$ are continuously twice differentiable, and that the Mangasarian-Fromovitz constraint qualification holds for our nonlinear problem. This means that the gradient of the equality constraint $\boldsymbol{H_z} \in \mathbb{R}^{n_z \times (T+1)\cdot n_x}$. is of full rank at $\boldsymbol{z}^*$, or in other words, the gradients of each individual constraint in $\boldsymbol{h}$ are linearly independent at $\boldsymbol{z}^*$.

From here we write the Lagrangian of the problem as

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\nu}, \boldsymbol{\theta}) = \mathcal{J}(\boldsymbol{z}, \boldsymbol{\theta}) + \boldsymbol{\nu}^T \boldsymbol{h}(\boldsymbol{z}, \boldsymbol{\theta}), \tag{4.2}$$

with $\boldsymbol{\nu} \in \mathbb{R}^{(T+1)\cdot n_x}$ the dual variable of the Lagrangian. Then, the KKT optimality conditions state that if $\boldsymbol{z}^*$ is solution of the problem (4.1) for $\boldsymbol{\theta}$, the gradients of the Lagrangian are null at $\boldsymbol{z}^*$:

$$\begin{aligned} \mathcal{L}_z(z^*, \nu, \theta) &= \mathcal{J}_z(z^*, \theta) + \nu^T H_z(z^*, \theta) = 0 \\ h(z^*, \theta) &= 0. \end{aligned} \tag{4.3}$$

Finally we suppose that for each $\nu \in \mathbb{R}^{(T+1)\cdot n_x}$ satisfying the KKT conditions and each $\Delta z \neq 0 \in \mathbb{R}^{n_z}$ so that $\Delta z^\top H_z(z^*, \theta) = 0$, we have the following inequality:

$$\Delta z^T \mathcal{L}_{zz}(z^*, \nu, \theta) \Delta z > 0 \tag{4.4}$$

Under these assumptions, Shapiro [Shapiro 1988] showed that the function $\mathcal{P}(\cdot)$ is directionally differentiable and that for each $\theta \in \mathbb{R}^{n_\theta}$, $\Delta\theta \in \mathbb{R}^{n_\theta}$, there exists ν verifying the KKT conditions such that $\mathcal{P}'(\theta; \Delta\theta)$, derivative along the direction $\Delta\theta$, is the unique solution of the following quadratic program:

$$\begin{aligned} &\underset{\Delta z}{\operatorname{argmin}} \left(\frac{1}{2} \Delta z^T \mathcal{L}_{zz}(z, \nu, \theta) \Delta z + \Delta z^T \mathcal{L}_{z\theta}(z, \nu, \theta) \Delta\theta \right) \\ &s.t. \quad \Delta z^\top H_z(z, \theta) = 0 \end{aligned} \tag{4.5}$$

Since in our case, the Lagrangian multiplier ν is unique due to the upper diagonal form of the constraints matrix, it turns out we can easily compute the directional derivative of our problem along the direction $\Delta\theta$, and from it immediately deduce the sensitivities of problem (4.1) under a KKT-like shape:

$$\begin{bmatrix} \frac{\partial \Delta z}{\partial \theta} \\ \frac{\partial \nu}{\partial \theta} \end{bmatrix} = - \begin{bmatrix} \mathcal{L}_{zz} & H_z^T \\ H_z & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{L}_{z\theta} \\ 0 \end{bmatrix} \tag{4.6}$$

In this equation, $\frac{\partial \nu}{\partial \theta}$ is the derivative of a Lagrangian multiplier increment we are not interested in. On the other hand, $\frac{\partial \Delta z}{\partial \theta} \Delta\theta$ is the correction to apply to the state and control trajectories when the parameter error is $\Delta\theta$. As literally solving (4.6) would be inefficient to find the derivatives, we now show how to exploit the DDP structure to evaluate the sensitivities.

4.2.2 Sensitivity computation

Equation (4.6) is very similar to the descent direction given by the QP described in (3.19). To illustrate this point, let us solve (3.19) by Lagrangian method rather than dynamic programming. To simplify notations, we rewrite the equation in a more compact form:

$$\begin{aligned} &\min_{\Delta z} \mathcal{J}(\Delta z) \\ &s.t. \quad h(\Delta z) = 0, \end{aligned} \tag{4.7}$$

with $\Delta z = (\underline{\Delta x}, \underline{\Delta u})$ and $h : \mathbb{R}^{n_z} \to \mathbb{R}^{(T+1)\cdot n_x}$ the $T+1$ dynamics constraints given by $h_{t+1} = \Delta x_{t+1} - (F_{x,t} \Delta x_t + F_{u,t} \Delta u_t + f_{t+1})$ $\forall t = 0..T-1$ and $h_0 = \Delta x_0 - f_0$. Additionally, we denote by $f = (f_0^\top, .., f_T^\top)^\top$ the concatenated dynamic drift.

As usual, the optimum of (4.7) is characterized by the gradients of the associated Lagrangian vanishing. Writing the Lagrangian under the expression:

$$\mathcal{L}(\Delta z, \nu) = \mathcal{J}(\Delta z) + \nu^T h(\Delta z), \tag{4.8}$$

and assuming the KKT matrix is invertible, the solution of QP (4.7) takes the following simple form:

$$\begin{bmatrix} \boldsymbol{\Delta z} \\ \boldsymbol{\nu} \end{bmatrix} = - \begin{bmatrix} \mathcal{L}_{\boldsymbol{zz}} & \boldsymbol{H}_{\boldsymbol{z}}^T \\ \boldsymbol{H}_{\boldsymbol{z}} & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{J}_{\boldsymbol{z}} \\ \boldsymbol{f} \end{bmatrix}. \tag{4.9}$$

Equation (4.9) is nearly the same as (4.6), except that the inverted KKT matrix is multiplied by $(-\mathcal{J}_{\boldsymbol{z}}, -\boldsymbol{f})$ rather than $(-\mathcal{L}_{\boldsymbol{z\theta}}, 0)$. Since DDP can be interpreted as an efficient operator to evaluate the multiplication by the KKT inverse, finding the sensitivities of the problem amounts to computing the DDP backward pass while replacing the terms $(\mathcal{J}_{\boldsymbol{x}}, \mathcal{J}_{\boldsymbol{u}}, \boldsymbol{f})$ of (3.21) with $(\mathcal{L}_{\boldsymbol{x\theta}}, \mathcal{L}_{\boldsymbol{u\theta}}, 0)$. This immediately leads to the following modified backward pass, $\forall t = 0..T-1$:

$$\boldsymbol{Q}_{\boldsymbol{x\theta},t} = \mathcal{L}_{\boldsymbol{x\theta},t} + \boldsymbol{F}_{\boldsymbol{x},t}^\top \boldsymbol{V}_{\boldsymbol{x\theta},t+1} \tag{4.10a}$$
$$\boldsymbol{Q}_{\boldsymbol{u\theta},t} = \mathcal{L}_{\boldsymbol{u\theta},t} + \boldsymbol{F}_{\boldsymbol{u},t}^\top \boldsymbol{V}_{\boldsymbol{x\theta},t+1} \tag{4.10b}$$
$$\boldsymbol{V}_{\boldsymbol{x\theta},t} = \boldsymbol{Q}_{\boldsymbol{x\theta},t} - \boldsymbol{Q}_{\boldsymbol{xu},t} \boldsymbol{K}_{\boldsymbol{\theta},t} \tag{4.10c}$$
$$\boldsymbol{K}_{\boldsymbol{\theta},t} = -\boldsymbol{Q}_{\boldsymbol{uu},t}^{-1} \boldsymbol{Q}_{\boldsymbol{u\theta},t}, \tag{4.10d}$$

with the following terminal condition:

$$\boldsymbol{V}_{x\boldsymbol{\theta},T} = \mathcal{L}_{\boldsymbol{x\theta},T}. \tag{4.11}$$

The gain matrix $\boldsymbol{K}_{\boldsymbol{\theta}}$ can be viewed as the derivative of the feedforward gain $\boldsymbol{k}_t$ in (3.17). Similarly, the terms $\boldsymbol{v}_{\boldsymbol{x}}, \boldsymbol{q}_{\boldsymbol{x}}$ and $\boldsymbol{q}_{\boldsymbol{u}}$ of (3.15) have been replaced by $\boldsymbol{V}_{\boldsymbol{x\theta}}, \boldsymbol{Q}_{\boldsymbol{x\theta}}$ and $\boldsymbol{Q}_{\boldsymbol{u\theta}}$ in (4.10). Propagating the forward pass based on (4.6) then gives:

$$\begin{aligned} \frac{\partial \boldsymbol{\Delta u}_t}{\partial \boldsymbol{\theta}} &= \boldsymbol{K}_t \frac{\partial \boldsymbol{\Delta x}_t}{\partial \boldsymbol{\theta}} + \boldsymbol{K}_{\boldsymbol{\theta},t} \\ \frac{\partial \boldsymbol{\Delta x}_{t+1}}{\partial \boldsymbol{\theta}} &= \boldsymbol{F}_{\boldsymbol{x},t} \frac{\partial \boldsymbol{\Delta x}_t}{\partial \boldsymbol{\theta}} + \boldsymbol{F}_{\boldsymbol{u},t} \frac{\partial \boldsymbol{\Delta u}_t}{\partial \boldsymbol{\theta}}. \end{aligned} \tag{4.12}$$

Equation (4.12) provides the sensitivity of the control command with respect to the variation of $\boldsymbol{\theta}$. The only limitation to this approach is that the matrix $(\mathcal{L}_{\boldsymbol{x\theta},t}, \mathcal{L}_{\boldsymbol{u\theta},t})$ should be simple to compute; ideally, only $(\boldsymbol{l}_{\boldsymbol{x},t}, \boldsymbol{l}_{\boldsymbol{u},t})$ should depend on $\boldsymbol{\theta}$ for the method to be efficient. If $(\boldsymbol{L}_{\boldsymbol{xx},t}, \boldsymbol{L}_{\boldsymbol{ux},t}, \boldsymbol{L}_{\boldsymbol{uu},t})$ are also depending on $\boldsymbol{\theta}$, 3-dimensional matrices start to intervene in the sensitivity computation, making it difficult to handle.

In what follows, we are mostly interested by $\frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{\theta}}$, sensitivity of the first control with respect to the parameter $\boldsymbol{\theta}$. Since the OCP (3.9) has the same derivatives as the LQR (3.19) at each knot, the sensitivity of both problems are equal:

$$\frac{\partial \boldsymbol{u}_0^*}{\partial \boldsymbol{\theta}} = \frac{\partial \boldsymbol{\Delta u}_0^*}{\partial \boldsymbol{\theta}}. \tag{4.13}$$

A feedback policy can then be obtained from (4.12) when $\boldsymbol{\theta}$ is measured at a higher frequency than the DDP solves (4.7). We examine two cases, where $\boldsymbol{\theta}$ is respectively the initial state $\boldsymbol{x}_0$ and desired position of the end effector $\boldsymbol{r}^*$. The first case is motivated by the need to mitigate the latency issue which arises when recomputing the MPC over a new state measurement; the second case has been selected to highlight the genericity of the approach through the implementation of a common end-effector tracking task.

Sensitivity over the initial state

We consider first the case where $\boldsymbol{\theta} = \boldsymbol{x}_0$, initial measured state in the discrete OCP described by (3.9). By deriving the forward pass equation in (3.18), it is straightforward to deduce the derivative of the first optimal control with respect to initial state:

$$\frac{\partial \boldsymbol{u}_0^*}{\partial \boldsymbol{x}_0} = \boldsymbol{K}_0. \tag{4.14}$$

This equation shows, as it is already known, that the Riccati gains $\boldsymbol{K}_0$ can be interpreted as the sensitivity of the optimal policy to initial state. Given that the current state of the robot is provided at a sufficiently high frequency, the Riccati gains act as a feedback term which approximates the optimal control between two LQR computations.

Sensitivity over a desired end effector position

In the following paragraph, the concatenated dual variables are defined as $\boldsymbol{\nu} = (\boldsymbol{\nu}_0^\top \cdots \boldsymbol{\nu}_T^\top)^\top \in \mathbb{R}^{T \cdot n_x}$. Let us develop the Lagrangian of the LQR in (4.8) under the form:

$$\begin{aligned}
\mathcal{L}(\underline{\boldsymbol{\Delta x}}, \underline{\boldsymbol{\Delta u}}, \boldsymbol{\nu}) &= \sum_{t=0}^{T-1} \tilde{\ell}_t(\boldsymbol{\Delta x}_t, \boldsymbol{\Delta u}_t) + \tilde{\ell}_T(\boldsymbol{\Delta x}_T) \\
&\quad + \begin{bmatrix} \boldsymbol{\nu}_0 \\ \cdots \\ \boldsymbol{\nu}_T \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{\Delta x}_0 - \boldsymbol{f}_0 \\ \cdots \\ \boldsymbol{\Delta x}_{t+1} - \boldsymbol{F}_{\boldsymbol{x},t}\boldsymbol{\Delta x}_t - \boldsymbol{F}_{\boldsymbol{u},t}\boldsymbol{\Delta u}_t - \boldsymbol{f}_{t+1} \\ \cdots \end{bmatrix} \\
s.t. \quad & \tilde{\ell}_T(\boldsymbol{\Delta x}_T) = \frac{1}{2}\boldsymbol{\Delta x}_T^\top \boldsymbol{L}_{\boldsymbol{xx},T}\boldsymbol{\Delta x}_T + \boldsymbol{\ell}_{\boldsymbol{x},T}\boldsymbol{\Delta x}_T \\
& \forall t = 0..T-1, \\
& \tilde{\ell}_t(\boldsymbol{\Delta x}_t, \boldsymbol{\Delta u}_t) = \frac{1}{2}\begin{bmatrix} \boldsymbol{\Delta x}_t^\top \\ \boldsymbol{\Delta u}_t^\top \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{L}_{\boldsymbol{xx},t} & \boldsymbol{L}_{\boldsymbol{xu},t} \\ \boldsymbol{L}_{\boldsymbol{ux},t} & \boldsymbol{L}_{\boldsymbol{uu},t} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta x}_t \\ \boldsymbol{\Delta u}_t \end{bmatrix} \\
& \qquad + \begin{bmatrix} \boldsymbol{\ell}_{\boldsymbol{x},t} & \boldsymbol{\ell}_{\boldsymbol{u},t} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta x}_t \\ \boldsymbol{\Delta u}_t \end{bmatrix}.
\end{aligned} \tag{4.15}$$

The derivatives of this Lagrangian with respect to state and control write:

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{x},T} &= \boldsymbol{\ell}_{\boldsymbol{x},T} + \boldsymbol{L}_{\boldsymbol{xx},T}\boldsymbol{\Delta x}_T + \boldsymbol{\nu}_T \\
\forall t &= 0..T-1, \\
\mathcal{L}_{\boldsymbol{x},t} &= \boldsymbol{\ell}_{\boldsymbol{x},t} + \boldsymbol{L}_{\boldsymbol{xx},t}\boldsymbol{\Delta x}_t + \boldsymbol{L}_{\boldsymbol{xu},T}\boldsymbol{\Delta u}_t - \boldsymbol{F}_{\boldsymbol{x},t}^\top \boldsymbol{\nu}_{t+1} + \boldsymbol{\nu}_t \\
\mathcal{L}_{\boldsymbol{u},t} &= \boldsymbol{\ell}_{\boldsymbol{u},t} + \boldsymbol{L}_{\boldsymbol{uu},t}\boldsymbol{\Delta u}_t + \boldsymbol{L}_{\boldsymbol{ux},t}\boldsymbol{\Delta x}_t - \boldsymbol{F}_{\boldsymbol{u},t}^\top \boldsymbol{\nu}_{t+1}.
\end{aligned} \tag{4.16}$$

Let us suppose that the cost function contains a term that penalizes the Cartesian position $\boldsymbol{r} \in \mathbb{R}^3$ of an end effector with respect to a desired position $\boldsymbol{r}^*$. We write this term $\ell^r(\boldsymbol{x}) = a(\boldsymbol{r}(\boldsymbol{x}) - \boldsymbol{r}^*)$, with $a : \mathbb{R}^3 \to \mathbb{R}^+$ a differentiable activation function, for example the square norm.

With this cost formulation, the only Lagrangian derivatives depending on $\boldsymbol{r}^*$ are $\mathcal{L}_{\boldsymbol{x},t}$, $\forall t = 0..T$, because they contain the term $\boldsymbol{\ell}_{\boldsymbol{x},t}$ in which lays the derivative of ℓ^r.

We neglect here the dependency of $\boldsymbol{L_{xx}}$ over $\boldsymbol{r}^*$, as computing it would mean to manipulate heavy 3-D matrices. We note $\boldsymbol{J}_{ee} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}$ the Jacobian of the end effector and $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ the Hessian of the activation function. Then, the Lagrangian derivative with respect to $\boldsymbol{r}^*$ writes, $\forall t = 0, T$:

$$\mathcal{L}_{\boldsymbol{xr},t} = -\boldsymbol{J}_{ee}(\boldsymbol{x}_t)^\top \boldsymbol{A}. \tag{4.17}$$

Consequently, the backward pass resulting in the computation of the final sensitivity $\boldsymbol{K}_{\boldsymbol{r},0}$ is, $\forall t = 0..T-1$:

$$\boldsymbol{Q}_{\boldsymbol{xr},t} = -\boldsymbol{J}_{ee}(\boldsymbol{x}_t)^\top \boldsymbol{A} + \boldsymbol{F}_{\boldsymbol{x},t}^\top \boldsymbol{V}_{\boldsymbol{r},t+1} \tag{4.18a}$$
$$\boldsymbol{Q}_{\boldsymbol{ur},t} = \boldsymbol{F}_{\boldsymbol{u},t}^\top \boldsymbol{V}_{\boldsymbol{r},t+1} \tag{4.18b}$$
$$\boldsymbol{V}_{\boldsymbol{r},t} = \boldsymbol{Q}_{\boldsymbol{xr},t} - \boldsymbol{Q}_{\boldsymbol{xu},t}\boldsymbol{K}_{\boldsymbol{r},t} \tag{4.18c}$$
$$\boldsymbol{K}_{\boldsymbol{r},t} = -\boldsymbol{Q}_{\boldsymbol{uu},t}^{-1}\boldsymbol{Q}_{\boldsymbol{ur},t}, \tag{4.18d}$$

with the following terminal condition:

$$\boldsymbol{V}_{\boldsymbol{r},T} = -\boldsymbol{J}_{ee}(\boldsymbol{x}_T)^\top \boldsymbol{A}. \tag{4.19}$$

This backward pass is simpler to compute as compared to the classical DDP backward pass, and results in a feedback gain $\boldsymbol{K}_{\boldsymbol{r},0}$ corresponding to an error on the desired position of the end effector.

4.3 Whole-body MPC formulation

In this section, we describe the formulation of our whole-body predictive control scheme and explain how the Riccati gains are used in practice to approximate the optimal control at the low-level frequency of the robot. The control problem to be solved is defined as a simple reaching task with the left hand.

4.3.1 OCP formulation

As highlighted before, we formulate our OCP under the form of a discretized NLP, similar to (3.9), with f being the discrete dynamics defined in Sec. 3.3. The running cost function does not depend on time and writes $\ell = \sum_{i=1}^{5} w_i \ell^i$, with $(w_i)_{i=1}^5$ a set of weighting scalars. The five different terms are defined as:

- **State regularization cost**:
  $\ell^x(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_d)^\top \boldsymbol{B}_x (\boldsymbol{x} - \boldsymbol{x}_d)$ with $\boldsymbol{B}_x$ a positive definite weight matrix and $\boldsymbol{x}_d$ the default initial state. This cost prevents state drift and excessive velocity spikes during the motion.

- **Control regularization cost**:
  $\ell^u(\boldsymbol{u}) = (\boldsymbol{u} - \boldsymbol{u}_d)^\top \boldsymbol{B}_u (\boldsymbol{u} - \boldsymbol{u}_d)$ with $\boldsymbol{B}_u$ a positive definite weight matrix and $\boldsymbol{u}_d$ the gravity-compensating torque in default state. This cost prevents excessive control spikes during the motion.

- **Hand translation cost**:
  $\ell^r(\boldsymbol{x}) = a_L(\boldsymbol{r}(\boldsymbol{x}) - \boldsymbol{r}^*)$ with $\boldsymbol{r}(\boldsymbol{x}) \in \mathbb{R}^3$ current end effector position, $\boldsymbol{r}^* \in \mathbb{R}^3$ desired end-effector position and $a_L : \boldsymbol{s} \mapsto log(1 + \frac{||\boldsymbol{s}||}{\alpha})$ a logarithmic activation function. In a neighbourhood of $\boldsymbol{r}^*$, the activation function acts as the squared norm function, while toward infinity its gradient tends to zero. As

a consequence, a target too far away results in a weak attraction. The width factor α is taken equal to 0.2.

- **Kinematic limit cost**:
  $\ell^b(\boldsymbol{x}) = \frac{1}{2}||max(\boldsymbol{x} - \boldsymbol{x}_u, \boldsymbol{0})||^2 + \frac{1}{2}||min(\boldsymbol{x} - \boldsymbol{x}_l, \boldsymbol{0})||^2$ with $\boldsymbol{x}_u$ the upper bound and $\boldsymbol{x}_l$ the lower bound of the joints positions and velocities. The goal of this cost is to keep the joints away from their kinematic limits. Upper and lower bounds include safety margins that can be tuned by the user, depending on the application.

- **CoM tracking cost**:
  $\ell^{com}(\boldsymbol{x}) = ||\boldsymbol{c}(\boldsymbol{x}) - \boldsymbol{c}^*||^2$ with $\boldsymbol{c}(\boldsymbol{x})$ current CoM and $\boldsymbol{c}^*$ desired CoM. This cost is implemented to prevent the CoM from drifting away of its support polygon. The desired CoM is deduced from the initial state.

The terminal cost function ℓ_T is identical to the running cost function, except that there is no control regularization cost nor kinematic limit cost. The role of this terminal cost is to approximate an infinite horizon problem through a single knot. To do so, the weight of the reaching task in ℓ_T is usually increased to force the completion of the motion at the end of the preview horizon.

During the execution of the task, the contact phases are imposed and the CoM position reference remains fixed in the center of the support polygon, so as to prevent the CoM from drifting out of balance. The friction constraint described in (3.33) is then trivially respected. Similarly, the default state $\boldsymbol{x}_d$ and default control $\boldsymbol{u}_d$ do not change during the execution of the MPC. The only quantity likely to vary in our reaching experiments is the desired end effector position $\boldsymbol{r}^*$.

4.3.2 Cost parametrization

	ℓ^x	ℓ^u	ℓ^r	ℓ^b	ℓ^{com}
Running costs	0.02	0.001	25	1000	500
Terminal costs	0.02		250		500

TABLE 4.1: Cost weights of our reaching OCP.

The weight distribution selected to perform the reaching task is presented in Table 4.1. The weight matrix $\boldsymbol{B}_u$ is the identity matrix, while the weight matrix $\boldsymbol{B}_x$ is described in Table 4.2 is carefully tuned to penalize critical behaviors like tilting torso or swinging base. The regularization weights on the left arm are reduced in order to focus on the reaching task.

Broadly speaking, the shape of the solution produced by the MPC varies continuously with respect to the weights of the cost functions. This means that the weights selected in this chapter could be tuned to favor one behavior over another, depending on the user's needs. Our approach first aims at establishing a general proof of concept of our whole-body predictive control framework, through the completion of simple tasks in a compliant and robust way.

4.3.3 Model and timings

The whole-body model includes 22 actuated joints (12 for the legs, 8 for the arms, 2 for the torso), plus the free flyer joint state: this results in a state dimension of size

	Base pose	Base angle	Leg	Torso	Left Arm	Right Arm
Position	100	10000	500	500	50	500
Velocity	10	10	10	10	1	10

TABLE 4.2: Diagonal terms of the weight matrix $\boldsymbol{B}_x$.

$n_x = 57$ ($n_q = 29, n_v = 28$) and control dimension of size $n_u = n_j = 22$. Since the complexity of the DDP increases as the cube of the state and control dimensions, using such a complex model does not allow to choose a very long preview horizon.

As the computational cost of the DDP algorithm is linear with respect to the number of knots, a compromise must be found between the predictive capacities of the MPC and its computational load. Because the DDP is initialized with the previous computed trajectories shifted by one knot, only one iteration of the algorithm is needed to generate a satisfying solution. Still, this one iteration can be costly depending on the choice of time parameters. Intuitively, a horizon length of 1 s appears to be a relevant guess to perform simple tasks like reaching an object or stepping forward. Besides, we would like to match approximately the MPC time computation and OCP timestep, because one control cycle is theoretically equivalent to one knot in the horizon. However, one must keep in mind that the Crocoddyl library uses for now a simple explicit Euler scheme to integrate the dynamics; as a consequence, long timesteps may lead to accuracy loss and control instabilities. More advanced integration schemes for DDP are under development [Jallet et al. 2022b], and we can expect in the near future to be able to reduce the number of knots in the horizon while safely increasing the timestep length to keep a constant preview period.

For all these reasons, we chose to work with an OCP composed of 100 knots separated by a 10 ms timestep. With this specific time parametrization, the dynamics of the robot is previewed over 1 s in the future, while one iteration of DDP takes about 13 ms. Experimentally, this choice of parameters has produced satisfying results in simulation and on the real robot.

4.3.4 Riccati interpolation of the MPC

Our DDP scheme produces the optimal control along with the sensitivities associated with the initial state of the robot and desired position of the end-effector. The first sensitivity $\boldsymbol{K}_0 = \frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{x}_0}$ is directly obtained during the backward pass of the DDP, at no additional cost. The second sensitivity $\boldsymbol{K}_r = \frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{r}}$ is computed through the modified backward pass described in Sec. 4.2.2.

The resulting feedback policy is a first order Taylor development of the MPC. Suppose that the MPC has been solved at a given observed state $(\boldsymbol{x}_0, \boldsymbol{r}_0)$, with $\boldsymbol{u}_0 = mpc(\boldsymbol{x}_0, \boldsymbol{r}_0)$ the resulting optimal command. Before the next MPC computation ends, a new observation $(\hat{\boldsymbol{x}}, \boldsymbol{r}^*)$ is measured, but cannot be used directly by the OCP solver. Our linear feedback scheme will then approximate the MPC solution between two recomputations:

$$mpc(\hat{\boldsymbol{x}}, \boldsymbol{r}^*) \approx mpc(\boldsymbol{x}_0, \boldsymbol{r}_0) + \boldsymbol{K}_0(\hat{\boldsymbol{x}} - \boldsymbol{x}_0) + \boldsymbol{K}_r(\boldsymbol{r}^* - \boldsymbol{r}_0) \tag{4.20}$$

It has been estimated that the Riccati gains might be too stiff to be actually used for feedback [Mason et al. 2016]. Yet we see here that they are nothing more than a linear interpolation of the MPC feedback, and, for a sufficiently high frequency,

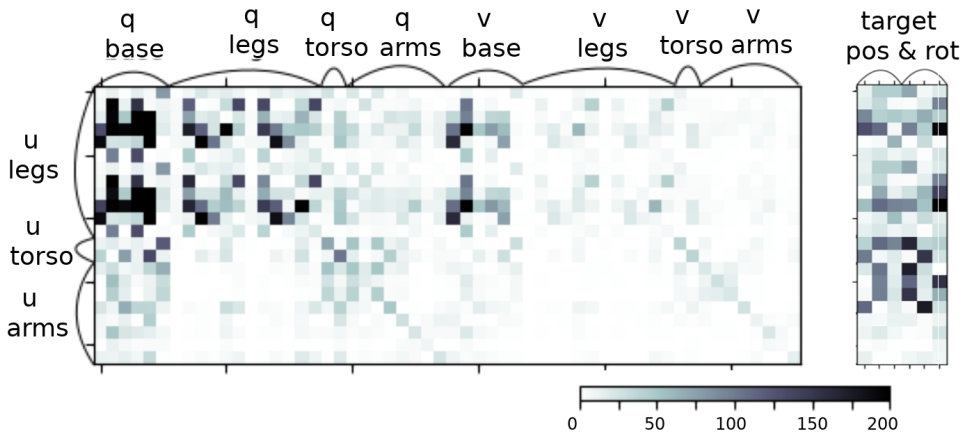

FIGURE 4.1: Left: matrix of the absolute Riccati gain $\boldsymbol{K}_0$ for a whole-body OCP involving end-effector tracking and biped stabilization. Right: matrix of the absolute placement (translation and rotation) gains for a whole-body OCP involving tracking a reference placement with the left hand while stabilizing balance. On the robot, only the first 3 columns of this matrix is used (corresponding to position feedback).

they lead to a fair numerical approximation of the MPC, hence are not stiffer than the MPC itself.

The structures of typical feedback gains for initial state and tracking target are presented in Fig. 4.1. It is interesting to note that the Riccati matrix for initial state presents a strong diagonal in position: the most contributing correction of a given joint torque strongly depends on the position error of this joint. It can also be noted that the feedback correction associated to the base position of the robot is very high, and affects mainly the leg controls. This is expected since the legs are the main drivers of the robot base position. Finally, one can notice that the velocity feedback coefficients of the Riccati matrix are small compared to the position feedback terms, which is reassuring since velocity estimates are typically more noisy, and LQR controllers tend to create brutal velocity feedbacks [Mason et al. 2016].

One could wonder how the local policy deduced from the Riccati gains would compare against a hand-tuned whole-body controller meant to track the desired reference state $\boldsymbol{x}^*$. This comparison was briefly discussed in [Mastalli et al. 2022a], where it was shown that a tracking controller with fixed gains does not perform better than our state-feedback scheme. Since our solution comes with no additional tuning and no performance loss, it is, in our opinion, the best choice to generate a low-level control policy on the robot.

4.3.5 Control scheme overview

The MPC process is described in Alg. 1. The OCP is first solved until convergence before starting the motion (line 9). The DDP algorithm is initialized with the constant state and control trajectories defined by the reference state $\hat{\boldsymbol{x}}$ and reference control $\boldsymbol{u} = RNEA(\hat{\boldsymbol{x}})$ (lines 2 and 3). These references represent the initial state of the robot before MPC launch and the corresponding gravity-compensated torque command, obtained through the Recursive Newton-Euler Algorithm [Featherstone 2008]. This warm-start is typically not feasible, but our improved version of the DDP scheme allows us to handle it.

At each control cycle, a single DDP iteration is performed, using the previous solution as an initial warm-start at lines 14 and 15, before iterating with the latest sensor measurements at line 16. The classical way to warm-start the solver is to shift the previous solution by one knot and use $\hat{\boldsymbol{x}}$ as the initial state [Diehl et al. 2005]. Aside from the measured state given by joint encoders and base IMU, the solver updates the tracking target position $\boldsymbol{r}^*$ obtained from motion capture camera. Once the OCP is solved (line 17), the feedback gains $(\boldsymbol{K}_0, \boldsymbol{K}_r)$, initial measures $(\boldsymbol{x}_0, \boldsymbol{r}_0)$ and feedforward torque $\boldsymbol{u}_0$ are sent to the low-level control block (line 18), where the linear feedback loop is computed. The resulting command $\boldsymbol{u}$ is forwarded to a torque-control scheme composed of a proportional-derivative feedback on the joint torque measurement $\hat{\boldsymbol{\tau}}$, plus a feedforward term that compensates for the intrinsic joint dynamics (motor inertia, high frictions, inner flexibility of the harmonic drive...). This dynamics is typically not considered inside the high-level whole-body model, but should nonetheless be taken into account when implementing motions on the real platform. All in all, the low-level torque feedback loop allows to consider that every joint behaves as an ideal joint from the viewpoint of the MPC.

To stop the tracking motion, the user disables the corresponding cost starting from the end of the horizon, and receding toward the first knot at each control cycle (line 22). With such an approach, the robot smoothly goes to initial configuration as the task function changes continuously from tracking to standing still. This illustrates one of the advantages of the preview horizon, which can be leveraged to inform the solver about future changes in problem formulation.

Algorithm 1 MPC algorithm for goal tracking

Require: T, T_{tot}
1: - Measure initial state $\hat{\boldsymbol{x}}$
2: - Initialize reference state: $\boldsymbol{x}_d \leftarrow \hat{\boldsymbol{x}}$
3: - Initialize reference control: $\boldsymbol{u}_d \leftarrow RNEA(\hat{\boldsymbol{x}})$
4: - Initialize target CoM: $\boldsymbol{c}^* \leftarrow CoM(\hat{\boldsymbol{x}})$
5: - Build preview horizon with T action models
6: - Set target r^* in OCP
7: - $\underline{\boldsymbol{x}} \leftarrow (\boldsymbol{x}_d)_{i=0}^{T}$
8: - $\underline{\boldsymbol{u}} \leftarrow (\boldsymbol{u}_d)_{i=0}^{T-1}$
9: - $\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*, K[0] \leftarrow ddp.solve(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ until convergence
10: - Initialize control cycle counter: $i_c \leftarrow 0$
11: **while** *True* **do**
12: - Update target r^* in OCP
13: - Measure new state $\hat{\boldsymbol{x}}$
14: - $\underline{\boldsymbol{x}}[0 : T-1] \leftarrow \underline{\boldsymbol{x}}[1 : T]$
15: - $\underline{\boldsymbol{u}}[0 : T-2] \leftarrow \underline{\boldsymbol{u}}[1 : T-1]$
16: - $\underline{\boldsymbol{x}}[0] = \hat{\boldsymbol{x}}$
17: - $\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*, \boldsymbol{K}_0, \boldsymbol{K}_r \leftarrow ddp.solve(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ with only one iteration
18: - Publish optimal policy $\boldsymbol{x}_0^*, \boldsymbol{u}_0^*, \boldsymbol{K}_0, \boldsymbol{r}^*$
19: - $\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}} \leftarrow \underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*$
20: - $i_c \leftarrow i_c + 1$
21: **if** $i_c >= T_{tot}$ **and** $i_c <= T_{tot} + T$ **then**
22: - Deactivate hand translation cost in action model number $T - (i_c - T_{tot})$
23: **end if**
24: **end while**

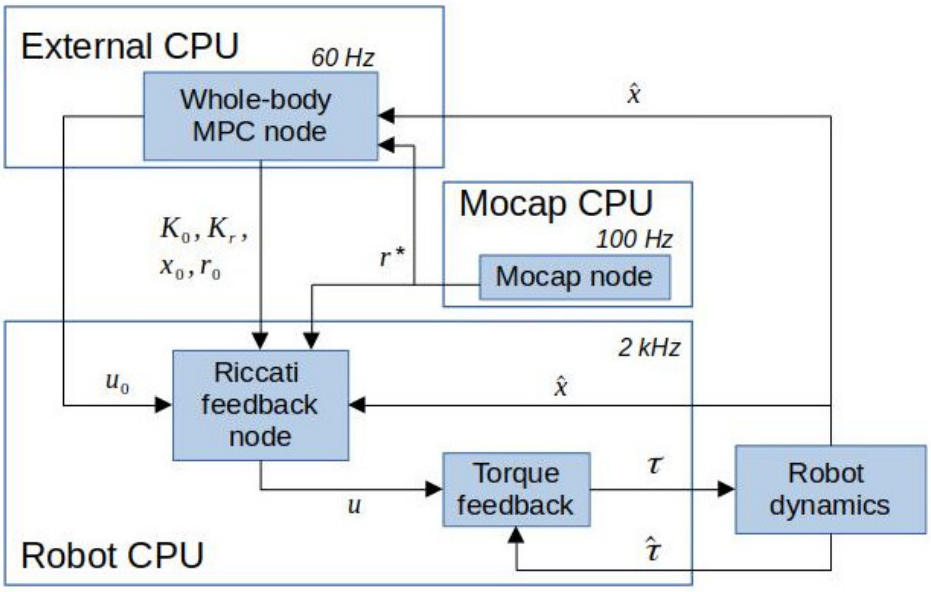

FIGURE 4.2: Diagram of the ROS implementation of our control scheme.

The architecture is illustrated in Fig. 4.2, and is composed of two parallel processes running on independent CPUs at different frequencies: one for the whole-body MPC, the other for the low-level feedback control. In addition, a motion capture system (MOCAP) measures at 100 Hz the position of the target to be reached and sends it to the other two nodes. ROS publisher-subscriber architecture is used to build the communication framework between all running processes to synchronize their different frequencies. Thus, every ROS node uses the latest available data on the topics it subscribes to, and publishes the output once computation is done.

4.4 Experimental results

The experiments evaluate the interest of the linear feedback for direct state feedback using $\frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{x}_0}$ (H1) and direct feedback to the target position $\boldsymbol{r}^*$ using $\frac{\partial \boldsymbol{u}_0}{\partial \boldsymbol{r}}$ (H2). A video summarizing the experimental results is available at `https://gepettoweb.laas.fr/articles/dantec22.html`.

In order to benchmark these feedbacks, three different experimental protocols were set up on Pyrène. For all experiments, the MPC node is running on a powerful external computer (AMD Ryzen 5950X, 16 cores and 4.9 GHz with 64 GB of RAM) whereas the low-level control node is running on the robot internal computer. Thanks to the efficiency of the Pinocchio and Crocoddyl libraries, the OCP is solved at approximately 70 Hz in each case.

Two of the experiments are based on the same whole-body tracking task, which requires the robot to reach a moving target with the end effector while balancing on both feet. The third experiment consists in testing the robustness and compliance of the control with respect to external disturbances. In all cases, the robot state is estimated by an observer provided by the manufacturer. No other additional filter were added to the state estimation block. Neck and wrist joints are kept fixed in position control for practical reasons.

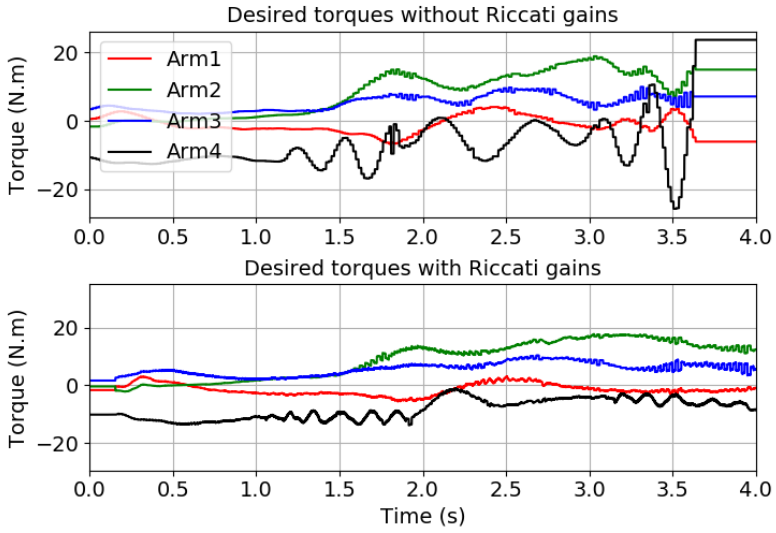

FIGURE 4.3: Experiment A: whole-body tracking experiment: without Riccati gains, the desired torque oscillations become too large after 3.6 seconds, and the securities of the robot are triggered, shutting down all motors. With Riccati gains, the torque trajectory remains stable.

4.4.1 Experiment A: Riccati feedback experiment

To test (H1), the whole-body tracking task was performed with and without the use of Riccati gains in the low-level control loop. For this experiment, the target $\boldsymbol{r}^*$ is not estimated from sensor but set to an arbitrary sinusoidal trajectory. Fig. 4.3 shows that the MPC alone produces higher reaction spikes (due to delay) which will eventually trigger the robot inner securities on torque and velocity. With the Riccati gains, the oscillatory peaks remain limited. Tracking accuracy can then be safely improved by raising the weight of the tracking cost, leading to Fig. 4.4. Note that without Riccati gains, it is not safe to use such weights on the robot.

We also experimented with a fixed gain matrix. While our robot is torque controlled (i.e. low-level motor controllers feedback on measured torques), we have copied the PD gains typically used when the robot is position controlled. Yet the behavior in simulation remains unstable and similar to no feedback, as shown in Fig. 4.5. This tends to illustrate that the off-diagonal Riccati terms in Fig. 4.1 cannot be ignored.

4.4.2 Experiment B: disturbances experiment

Our second experimental protocol aims at showing that direct state feedback (H1) alone produces a decent control policy able to stabilize the robot under external disturbances. The experiment consists in launching the MPC with no goal-tracking cost, then shutting it down and applying external disturbances to the robot. The policy sent to the low-level control then becomes $\boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{K}_0(\hat{\boldsymbol{x}} - \boldsymbol{x}_0)$, with $\hat{\boldsymbol{x}}$ being the only varying quantity. This formulation is approximating the optimal stabilizing policy around $\boldsymbol{x}_0$. Here $(\boldsymbol{u}_0, \boldsymbol{K}_0, \boldsymbol{x}_0)$ are drawn from the last DDP computation of the high-level control. As can be seen in Fig. 4.6, the Riccati gains efficiently reject the perturbations and produce a smooth stabilizing torque in response to it. While

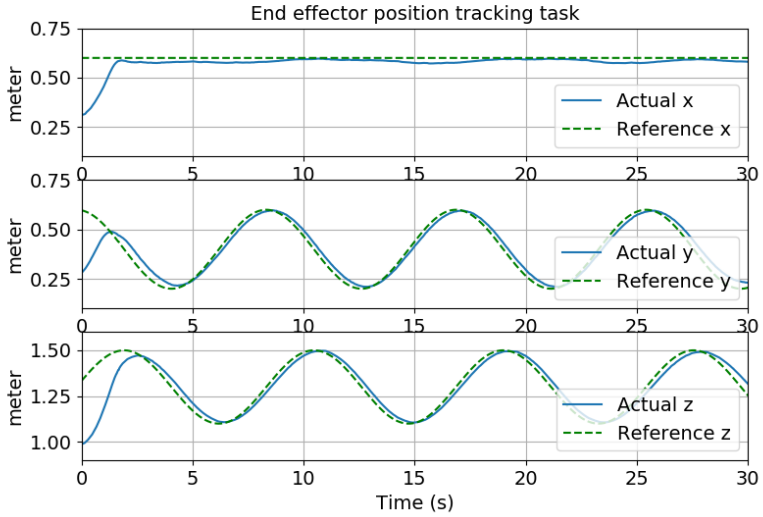

FIGURE 4.4: Experiment A: tracking plot for the MPC scheme with Riccati gains. Tracking weights are two times higher than in Fig. 4.3 (50 instead of 25).

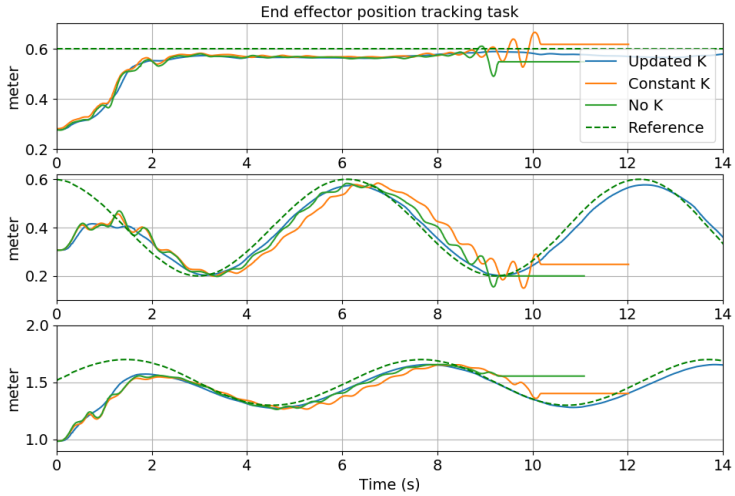

FIGURE 4.5: Experiment A (in simulation): tracking plot for the MPC scheme with Riccati gains, with constant gains and no gains.

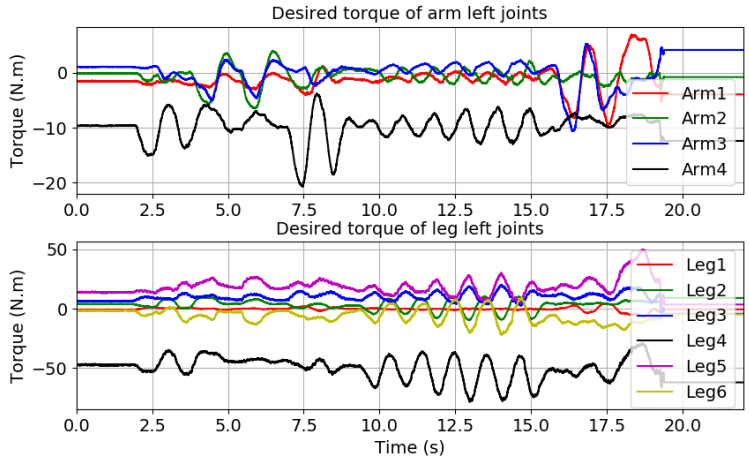

FIGURE 4.6: Experiment B: the MPC is shut down and external disturbances are manually applied on the left arm and shoulder. Disturbances start at $t = 2.4$ s and continue up until $t = 20$ s.

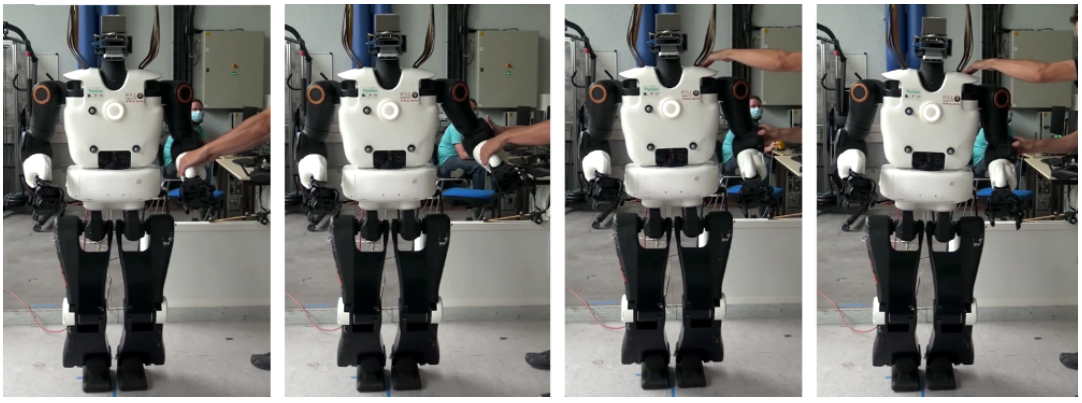

FIGURE 4.7: Experiment B: snapshot of the robot behavior subjected to external disturbances.

our goal is not to promote a purely linear feedback, this experiment illustrates how large the stable domain of our linear policy can be.

4.4.3 Experiment C: placement feedback experiment

Our third experimental protocol aims at testing (H2) by implementing a feedback policy on the target position, as described in (4.20). This time the MOCAP system is used to provide the desired placement of the end effector at a frequency of 100 Hz. Given that this frequency is similar to the MPC node frequency, the target feedback is small compared to the feedforward term $\boldsymbol{u}_0$. To better observe the feedback effect, the MPC has been shut down so that the drift in desired placement becomes significant over time.

The resulting plots of experiment C are presented in Fig 4.8 and Fig. 4.9: first, the MPC node is started and the OCP is solved at approximately 70 Hz. Then, at $t = 18$ s, we shut down the MPC but keep the target moving to illustrate the effect of the placement feedback gains. From this moment the control is only due to state and target feedbacks. Finally, starting from $t = 38$ s, external disturbances are applied to the left arm the robot.

Using the sensitivities to approximate the optimal solution is relevant as long as the current OCP, defined by the current state and target position, does not vary too much from the OCP computed just before shutting down the MPC. The architecture of this scheme is the same as in Fig. 4.2, without the MPC node to update the terms $(\boldsymbol{u}_0, \boldsymbol{r}_0, \boldsymbol{x}_0, \boldsymbol{K}_0, \boldsymbol{K}_r)$. Switching off the MPC and relying only on the Riccati gains lead the robot state to remain in the vicinity of the initial state used by the MPC node to compute the last OCP solution. As a consequence, the accuracy of the tracking decreases after the MPC shutdown. Nevertheless, placement gains provide a coarse approximation of the optimal control and are sufficient to move the end effector toward the desired target and reject external perturbations, as observed in Fig. 4.9.

Figure 4.10 provides the computation cost of the DDP and the extra loop computing $\boldsymbol{K}_r$. As expected, $\boldsymbol{K}_r$ is much cheaper to compute as it always corresponds to back-propagating only 3 columns, as opposed to the classical backward pass which works with the n_x columns of $\boldsymbol{K}_t$. As can be observed on Fig. 4.10, the computation of one DDP iteration can take up to 17 ms, and approximately 13.5 ms on average. Despite random time spikes, the control remains stable thanks to the Riccati feedback on state and target position.

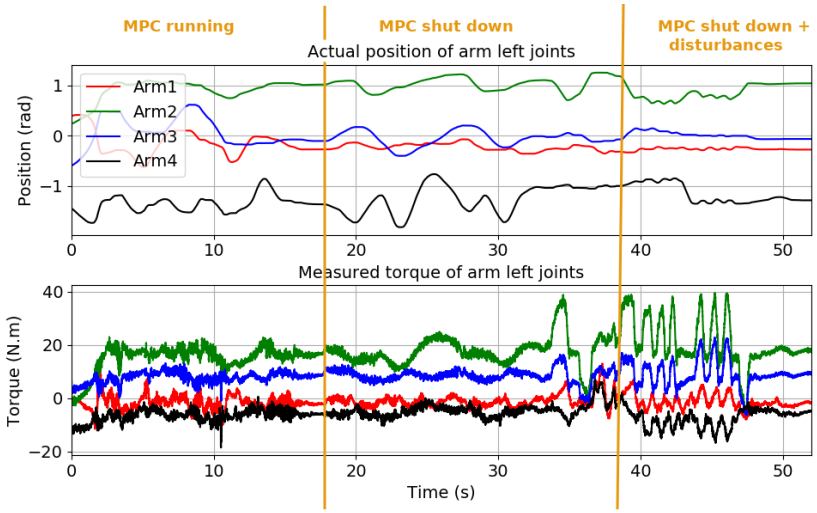

FIGURE 4.8: Experiment C: placement feedback state and control plots: MPC is shut down at $t = 18\ s$, and external disturbances start to be applied at $t = 38\ s$.

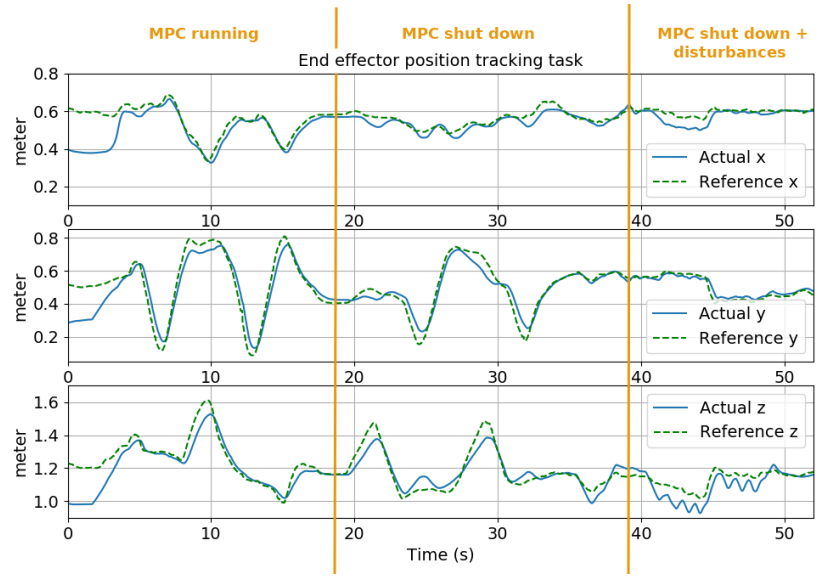

FIGURE 4.9: Experiment C: placement feedback tracking plot.

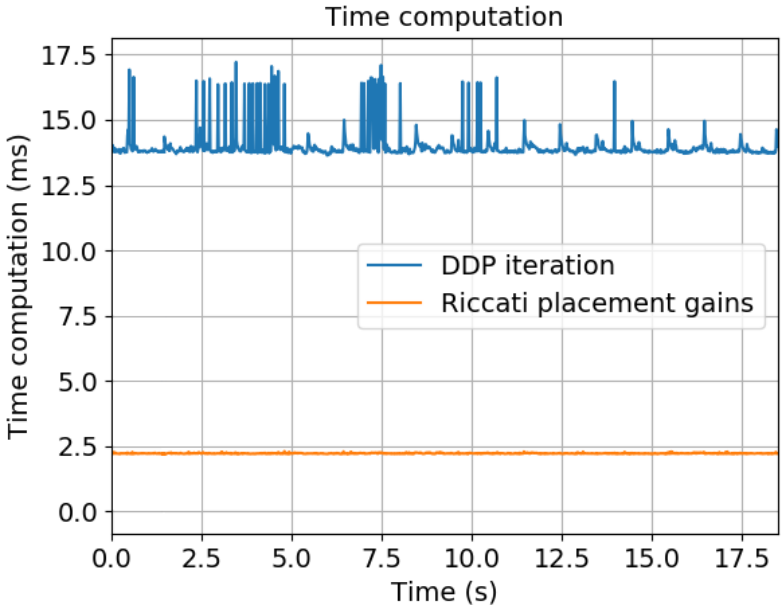

FIGURE 4.10: Experiment C: placement feedback time plot.

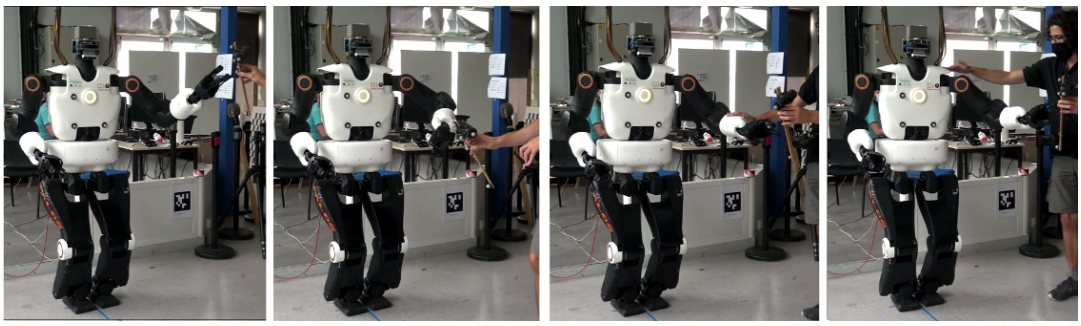

FIGURE 4.11: Experiment C: snapshot of the robot reaching a MOCAP target while being subjected to external disturbances.

4.5 Conclusion

This chapter has highlighted the idea that the classical feedback gains obtained during the Riccati recursion of the DDP can be used as sensitivities with respect to the initial state of the OCP. These gains allow to interpolate the optimal control at 2 kHz in order to produce dynamic yet stable motions that would otherwise fail because of the MPC computation latency. The importance of the Riccati feedback loop has been illustrated on a whole-body tracking movement on the full scale humanoid robot Pyrène. Additionally, other sensitivities can be computed and used at 2 kHz in order to adapt the control to a rapidly changing parameter, e.g. the desired end effector position. This is especially useful to deduce at high frequency a good approximation of the optimal policy when small disturbances are observed by fast sensors.

This work paves the way for the implementation of a real-time whole-body torque MPC with dynamic contacts, able to perform challenging tasks like walking or interacting with the environment, at arbitrary high feedback frequency. The MPC scheme was proven to be compliant against unexpected disturbances, a key feature for the implementation of a walking controller. The next step to be taken consists in building a preview horizon with varying contact constraints representing single support and double support phases during locomotion or manipulation tasks.

An interesting extension to the whole-body MPC framework is to study the relevance of force feedback applied to manipulation tasks with contact. On this matter, we have collaborated with Sebastien Kleff on a whole-body sanding experiment with Pyrène and have published the results in [Kleff et al. 2022]. In order to not overload the manuscript, we will not discuss this work in the frame of this thesis, but we mention it nonetheless as a promising research direction for future developments.

Chapter 5

Locomotion with foot references

Humanoid robots have now been walking for more than 50 years, using various heuristics ranging from simple passive walking schemes to complete locomotion frameworks which involve footstep optimization and push recovery strategies. Robot models are getting more and more sophisticated with numerous occurrences of full dynamics predictive approaches on quadruped hardwares [Neunert et al. 2018; Katayama and Ohtsuka 2022; Mastalli et al. 2022b; Mastalli et al. 2022a][1]. Yet, few demonstrations of full dynamics schemes have taken place on biped hardwares.

This chapter presents a whole-body MPC framework for locomotion on flat and uneven terrain, illustrated by real-world experiments on the torque-controlled platform Pyrène. Our experimental validation shows good and promising results for dynamic locomotion at different gaits as well as 10 cm height stairstep crossing. The method is fully generic and can be combined with a high-level footstep planner for locomotion in cluttered environments. The mathematical tools previously introduced are used to build a hybrid dynamic problem over a receding horizon; wrench cone constraints are approximated by convex regularization and foot reference trajectories are hand-tuned splines set to minimize end effector velocity at take-off and landing. The results contained in this chapter were published in [Dantec et al. 2022b].

5.1 Motivation

Locomotion of biped robots are characterized by unstable dynamics, physical limitations of contact forces and, in some cases, non-negligible inertial effects. Predictive control, in particular ZMP preview schemes, has proven to be a suited technique to tackle the locomotion problem, because of its ability to handle constrained dynamics and to replan the motion on the fly, starting from the next sampling state. A crucial question is to identify which control problem should be solved online to perform robust walking, or in other words, which level of model complexity should be considered to maximize robustness. Classical approaches based on template models quickly show their limits in terms of genericity and versatility. Because such methods rely on specific heuristics to ensure dynamic feasibility, intrinsic compliance and control stability, they tend to result in complex software architectures that are time-consuming to implement on hardware. Besides, they typically neglect the

[1] Papers from C. Mastalli are also part of the Memmo Project.

effect of limbs inertia in the preview horizon, although it is significant for heavy robots like Pyrène.

On the other hand, we argue that our whole-body predictive control framework is more compact and generic enough to handle various locomotion situations with few to no tuning efforts. In order to provide a preliminary proof of concept, we wish to design a walking formulation based on pre-computed foot trajectories and to test it on different scenarios. The locomotion problem, as compared to the tracking task presented in Chap. 4, includes contact switch events whose dynamics is discontinuous and complex to integrate, yet our control framework is robust enough to deal with such disturbances. Due to the huge computational power demanded by our solver, it is particularly essential to implement the Riccati feedback policy presented in Chap. 4 inside the low-level part of the controller.

5.2 Walking MPC formulation

This section introduces the formulation of our whole-body MPC for generalized locomotion on uneven ground. As in Chap. 4, we leverage the Riccati feedback gains inside our low-level control block to approximate the optimal torque command at 2 kHz. The key point of the locomotion problem is to ensure that the contact forces remain inside a stable contact wrench cone taking into account friction constraints and CoP constraints. Contact transitions are handled by defining in advance all the contact models along the horizon, following a hybrid dynamics approach. A receding strategy is then implemented to shift the horizon of the problem at each control cycle.

5.2.1 OCP formulation

Similar to Sec. 4.3.1, the OCP formulation is based on contact-constrained dynamics f and cost $\ell = \sum_{i=1}^{6} w_i \ell_i$. Three cost terms in ℓ have already be introduced in Sec. 4.3.1: state regularization ℓ^x, control regularization ℓ^u and kinematics limit cost ℓ^b. These particular terms remain unchanged in our walking OCP, save for their respective weights which have been tuned to improve the walk.

Foot placement cost

The fourth cost is a foot placement term whose role is to track a reference placement trajectory $\boldsymbol{p}^*(t) \in \mathbf{SE}(3), \forall t = 0..T$. For each foot, the placement cost writes:

$$\ell^p(\boldsymbol{x}, t) = a_L(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}^*(t)), \tag{5.1}$$

with $\boldsymbol{p}(\boldsymbol{x}) \in \mathbf{SE}(3)$ foot placement when the robot state is $\boldsymbol{x}$. Function a_L represents the logarithmic activation function introduced in Sec. 4.3.1, here applied to a residual of dimension 6. The rotation part of residual $\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}^*(t)$ is computed through the rotation logarithm introduced in Sec. 3.2.1.

Wrench tracking cost

In order to regulate the contact forces during support phases, a wrench tracking cost was implemented inside our control framework. Its role is to approximate the 6-D contact constraints summarized in (3.34). For each contacting foot, one wrench cost function is defined, similarly to the foot placement cost. As usual, we denote by

$\boldsymbol{\lambda} = (\boldsymbol{f}, \boldsymbol{\tau})$ the contact wrench, (W, L) half-width and half-length of the rectangular feet and μ the friction parameter. We also define f_{min}, f_{max} to be the minimum and maximum normal forces of the contact interaction. All inequality constraints defined by (3.34) can then be concatenated into a single expression [Caron et al. 2015b]:

$$\begin{aligned} \boldsymbol{b}^l &\leq \boldsymbol{A}_c \boldsymbol{\lambda} \leq \boldsymbol{b}^u \\ \boldsymbol{b}_i^l &= \begin{cases} -\infty & \text{if} \quad i \neq 5 \\ f_{min} & \text{else} \end{cases} \\ \boldsymbol{b}_i^u &= \begin{cases} 0 & \text{if} \quad i \neq 5 \\ f_{max} & \text{else} \end{cases} \end{aligned} \tag{5.2}$$

In this inequality, $\boldsymbol{b}^l \in \mathbb{R}^{17}$ and $\boldsymbol{b}^u \in \mathbb{R}^{17}$ are the corresponding lower and upper bounds of the constraints, with $-\infty$ the virtual numerical limit of the machine. $\boldsymbol{A}_c \in \mathbb{R}^{17\times 6}$ is the following matrix:

$$\boldsymbol{A}_c = \begin{bmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \\ \boldsymbol{A}_3 \\ \boldsymbol{A}_4 \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{A}_1 &= \begin{bmatrix} 1 & 0 & -\mu & 0 & 0 & 0 \\ -1 & 0 & -\mu & 0 & 0 & 0 \\ 0 & 1 & -\mu & 0 & 0 & 0 \\ 0 & -1 & -\mu & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & -W & 1 & 0 & 0 \\ 0 & 0 & -W & -1 & 0 & 0 \\ 0 & 0 & -L & 0 & 1 & 0 \\ 0 & 0 & -L & 0 & -1 & 0 \end{bmatrix}, \\ \boldsymbol{A}_3 &= \begin{bmatrix} W & L & -\mu(L+W) & -\mu & -\mu & -1 \\ W & -L & -\mu(L+W) & -\mu & \mu & -1 \\ -W & L & -\mu(L+W) & \mu & -\mu & -1 \\ -W & -L & -\mu(L+W) & \mu & \mu & -1 \end{bmatrix}, \\ \boldsymbol{A}_4 &= \begin{bmatrix} W & L & -\mu(L+W) & \mu & \mu & 1 \\ W & -L & -\mu(L+W) & \mu & -\mu & 1 \\ -W & L & -\mu(L+W) & -\mu & \mu & 1 \\ -W & -L & -\mu(L+W) & -\mu & -\mu & 1 \end{bmatrix}. \end{aligned} \tag{5.3}$$

$\boldsymbol{A}_1$ encodes the friction cone and minimal force constraint represented by (3.34a) and (3.34b). $\boldsymbol{A}_2$ encodes the local CoP constraint in (3.34c) and (3.34d). Lastly, $\boldsymbol{A}_3$, $\boldsymbol{A}_4$ encode the yaw minimum and maximum torque values represented by (3.34e).

Inequality (5.2) describes a linearized wrench cone with 4 facets. The linearization of friction cones has been historically used in the frame of linear predictive control with template models. Since the DDP algorithm handles nonlinear costs, it is possible to use instead the true quadratic cone defined by $||\boldsymbol{f}_{x,y}|| < \mu f_z$. Yet, doing so is unlikely to improve the solution in a significant way, since the DDP scheme does not allow to precisely take into account such inequality constraints. We then chose to stick with the 4-facet representation.

Using quadratic barrier costs to approximate the wrench cone constraints makes the problem too hard to solve in real time, as the solver struggles to discover the optimal contact forces in just one iteration. Unfortunately, the computation load of

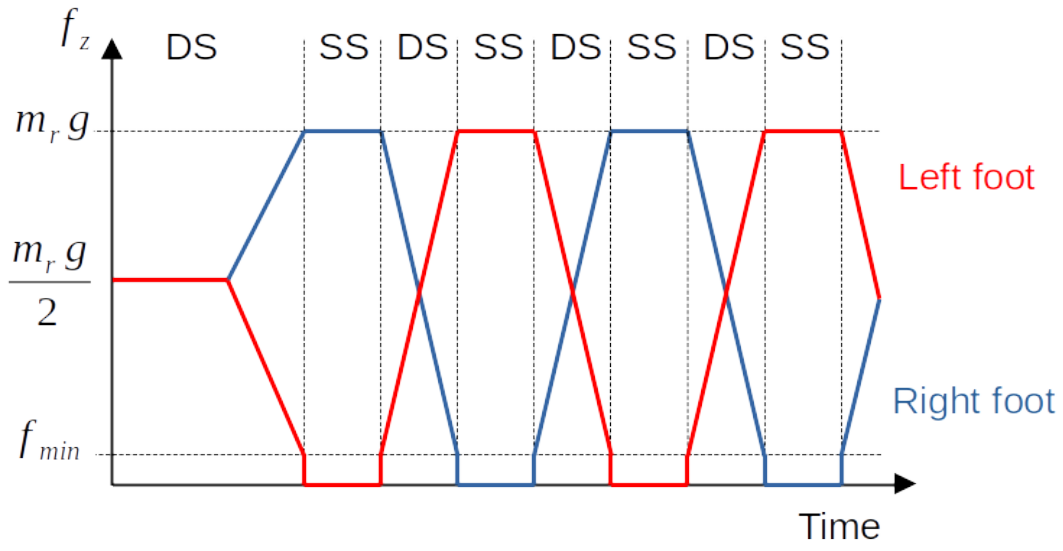

FIGURE 5.1: Force reference trajectory during double support (DS) and single support (SS) phases.

the whole-body MPC does not allow to perform more than one iteration at each control cycle. Thus, the solver appears to need more information to converge quickly toward a reliable solution. At first, we tried to introduce a reference wrench trajectory $\boldsymbol{\lambda}^* = (0, 0, f^*, 0, 0, 0)$ with f^* a normal force reference equal to $m_r g$, weight of the robot in simple support phases, and half the weight in double support phases. This approach lead to very brutal contact transitions as the solver attempted to follow a discontinuous force trajectory, switching from 0 to $m_r g/2$ and then to $m_r g$. A simple interpolation was then added to smooth the behavior of the robot between contact modes. The reference was selected to be $\boldsymbol{\lambda}^*(t) = (0, 0, f^*(t), 0, 0, 0)$ where $f^*(t)$ is equal to $m_r g$ during single support phases and switches continuously from $m_r g - f_{min}$ to f_{min} during double support phases (see Fig. 5.1). Here, f_{min} is a security margin and should be read as the minimum force reference at contact just before take-off.

The final wrench tracking cost is then defined by:

$$\ell^{\lambda}(\boldsymbol{x}, \boldsymbol{u}, t) = ||\boldsymbol{A}_c(\boldsymbol{\lambda} - \boldsymbol{\lambda}^*(t))||^2, \tag{5.4}$$

with every wrench expressed in the local frame of the contact foot. One alternative formulation of this cost consists in replacing the matrix $\boldsymbol{A}_c$ with a diagonal weight matrix of size 6×6 to regularize individually each component of the wrench. This alternative has been tested on the robot but has not resulted in stable motions. In fine, matrix $\boldsymbol{A}_c$ represents a set of empirically stabilizing gains with off-diagonal terms that cannot be cut off. The role of cost ℓ^{λ} is to ensure that the contact constraint residual $\boldsymbol{A}_c\boldsymbol{\lambda}$ does not diverge too much from the residuals computed from the normal force reference $f^*(t)$.

Local CoP regularization cost

The wrench tracking cost alone does not prevent occasional breakings of the non-tipping constraint during contact transition. Practically speaking, it has been observed in simulation and on the robot that the local CoP constraints described by

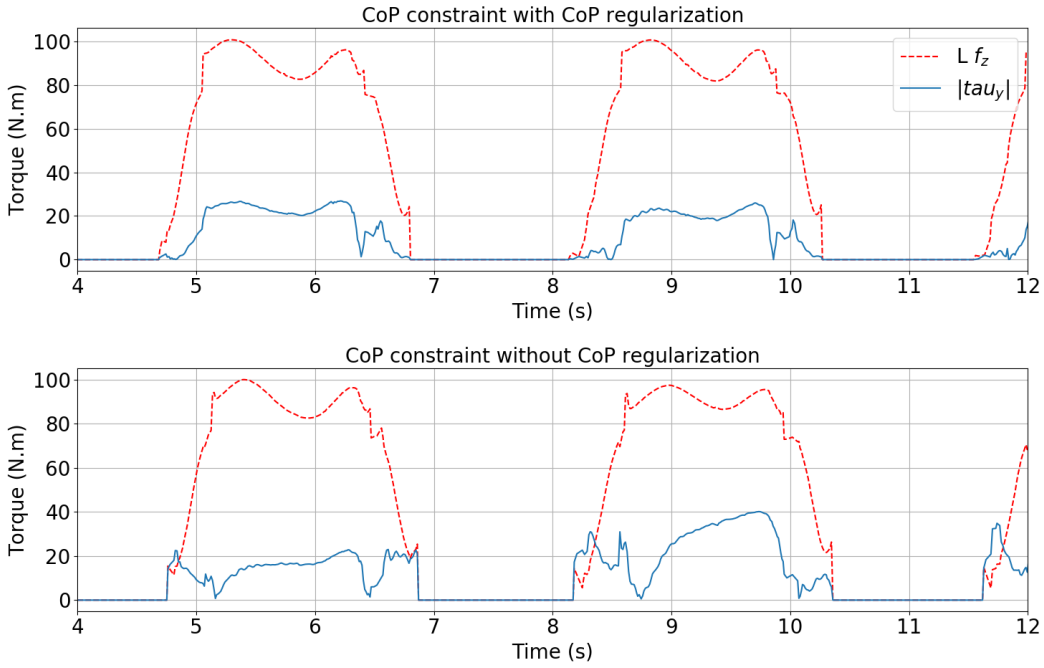

FIGURE 5.2: Left foot CoP constraint $|\tau_y| \leq Lf_z$ for simulated walk with and without CoP regularization. Blue plain line is $|\tau_y|$, red dotted line is Lf_z. Bottom plot shows violations of the constraint during contact switch, which translate into oscillating ankles in simulation. On the real robot, those constraints violations lead to falling.

inequalities (3.34c) and (3.34d) are violated during contact switches, although the wrench tracking cost ℓ^λ should regularize the contact torque to zero. To address this behavior, a local CoP regularization cost defined for each foot has been added to the formulation:

$$\ell^{cop}(\boldsymbol{x}, \boldsymbol{u}) = \frac{1}{2W^2}\left(\left|\frac{\tau_y}{f_z}\right|^2 + \left|\frac{\tau_x}{f_z}\right|^2\right). \tag{5.5}$$

The idea behind this cost is to force the non-tipping inequalities $|\frac{\tau_y}{Lf_z}| < |\frac{\tau_y}{Wf_z}| \leq 1$ and $|\frac{\tau_x}{Wf_z}| \leq 1$ by penalizing the corresponding quantities, with typically $L > W$. In order to be more conservative, W can be set smaller than the half-width of the foot, which is equivalent to increasing the CoP cost weight.

The effect of this cost is illustrated in Fig. 5.2, where a walking motion was performed in simulation with and without the CoP regularization. When $|f_z|$ is low, typically near contact transition, constraints (3.34c) and (3.34d) are harder to respect, and it becomes critical to penalize $|\frac{\tau_x}{f_z}|$ and $|\frac{\tau_y}{f_z}|$. We see here that the minimum margin imposed by f_{min} is implemented so that cost ℓ^{cop} does not become ill-conditioned when f^* tends toward 0.

5.2.2 Cost parametrization

The weight distribution used to perform locomotion is presented in Table 5.1. Compared to Chap. 4, the state and control weights have been multiplied by 5 in order to enhance motion regularization. The goal was to mitigate sharp velocity and control spikes occuring at contact switches.

	ℓ^x	ℓ^u	ℓ^p	ℓ^b	ℓ^λ	ℓ^{cop}
Running costs	0.1	0.005	1000	1000	0.001	10
Terminal costs	0.1		1000			

TABLE 5.1: Cost weights of our walking OCP.

The weight matrix $\boldsymbol{B}_u$ is again the identity matrix, while the new weight matrix $\boldsymbol{B}_x$ is described in Table 5.2. Compared to the previous chapter, $\boldsymbol{B}_x$ has been modified to reduce legs position impedance while still penalizing swing base motion. These weights have been deployed on the robot and have resulted in robust walking motions at different gaits, on flat ground and uneven terrain.

Apart from the tuning of the cost weights, one empirical heuristics has been used to improve the walking motion. We activate the foot placement cost of each foot during the corresponding swing phases, but also during double support phases. In theory, the placement constraint induced by the contact dynamics should nullify the effect of this cost during double supports, but we found out that letting the cost active lead to more stable contact transitions.

	Base pose	Base angle	Leg	Torso
Position	0	10000	10	100
Velocity	10	10	10	10

TABLE 5.2: Diagonal terms of the weight matrix $\boldsymbol{B}_x$.

5.2.3 Model and timings

The size of the horizon to be considered for locomotion is of critical importance. When it comes to centroidal planning and capturability of the LIP system, a 2-step preview horizon is sufficient to ensure balance [Zaytsev et al. 2015]. As for whole-body biped walking, it seems preferable to encompass at least one full step inside the preview horizon. Biological studies on biped walking in natural environments suggest that humans maintain a look-ahead of $\sim$ 1.5 s when planning for their next set of actions [Matthis et al. 2018].

Likewise, we choose to generate an horizon of 150 knots separated by a 10 ms timestep, for a total of 1.5 s prediction window. From a theoretical point of view, it seems preferable to have a timestep as close as possible to the average computation time of one DDP iteration. Indeed, each control cycle should correspond to a one knot shift of the preview horizon. From an empirical perspective however, we noticed that lowering the timestep length in simulation results in more dynamic walking motions: this is actually to be expected, since a lower timestep translates into a shorter preview horizon, given a constant number of knots. As a consequence, the robot is asked to perform the same motion in much less time, and it can only succeed by being more dynamic. Finally, it has been observed that a timestep higher than 10 ms tends to produce control instabilities on the robot, particularly at contact transitions. It is likely that our Euler integration scheme struggles with long timesteps coupled with brutal changes in dynamics. In the light of this observation, we choose to stick to a 10 ms timestep, as in Chap. 4.

Finding the optimal frequency at which the whole-body MPC should be recomputed is an open problem to this day. In the case of centroidal schemes deployed on humanoid robots, it has been suggested that a sampling frequency of up to 200 ms is acceptable for CoM and CoP tracking [Villa et al. 2019]. However, as far as whole-body control is concerned, fast update frequency empirically leads to better performances, as our preview scheme deals with both centroidal and actuator dynamics. In order to cut the computation load as much as possible, we choose to use a reduced robot model including a total of 14 torque-controlled joints: 6 for each leg and 2 for the torso, resulting in $n_x = 41$ and $n_u = 14$. The arm joints are controlled in position and remain fixed during the walking motion. From the viewpoint of the high-level MPC, the arm joints do not exist. It has been demonstrated that including the arms in the model allows for a better control of the centroidal angular momentum of the robot [Khazoom and Kim 2022]. Using only a half-body model makes the locomotion more difficult as the legs and torso alone have to compensate for the heavy inertia of the upper body. Nevertheless, our control scheme was proven able to handle this issue.

5.2.4 Receding horizon strategy

In order to leverage the predictive capabilities of our MPC scheme, a receding horizon strategy was implemented to perform robust locomotion on Pyrène. We consider a preview horizon composed of T action models, which encodes the OCP that will be solved through DDP. We also consider two other sets of action models, named walking cycle horizon and standing linear horizon. The walking horizon is composed of a periodic sequence encoding 2 consecutive steps, while the standing horizon is composed of only double support phases. Inside walking and standing horizons are encapsulated the reference wrench trajectories for smooth transitions between single and double supports.

The receding process is described in Alg. 2 and in Fig. 5.3. The OCP initialization step and update step are the same as in Alg. 1, so for the sake of clarity these steps are not detailed in Alg. 2, but summarized in lines 4 and 10. The user first defines a single support timing T_s, double support timing T_d and total horizon length T. At initialization, the three different horizons are built using Double Support (DS) and Single Support (SS) models (lines 1, 2 and 3). SS models are themselves divided between Left Foot support (LF) and Right Foot support (RF) models. At each MPC update, the preview horizon recedes and its last action model becomes a copy of the current action model belonging to the walking cycle (line 12). The walking cycle loops on itself as long as the user intends to keep on moving. To stop the locomotion, the user needs to update a boolean parameter that will connect the end of the preview horizon to the standing horizon (line 22). The standing horizon will begin to be depleted only when the walking cycle comes to the end of a step, to ensure contact continuity (line 11). Once the standing horizon is fully browsed, the structure of the problem ceases to change and the robot remains in resting position. With this sort of architecture, the user does not have to plan in advance how long they intend to walk.

During double support phase, at each control cycle, the reference trajectories for future swing feet are updated over the entire horizon using a minimum jerk Bezier curve. The final placement of each swing foot is set to be the current placement of the other foot in support phase, plus a desired 3-D translation $(\delta x, \delta y, \delta z)$. These curves are tuned so that during landing and take-off, foot velocity and acceleration

Algorithm 2 Receding horizon algorithm for locomotion

Require: T, T_d, T_s

- Build preview horizon with T DS models
- Build walking cycle with $2T_d$ DS and $2T_s$ SS models
- Build standing horizon with T double support models
- Solve OCP with preview horizon until convergence
- Initialize counter for walking knots: $i_w \leftarrow 0$
- Initialize counter for standing knots: $i_s \leftarrow 0$
- Initialize state of motion boolean: *IsWalking* $\leftarrow$ *True*

while *True* **do**
    - Measure new state $\hat{\boldsymbol{x}}$, set new warm-start from previous solution
    - Recede preview horizon by one knot
    if *IsWalking* **or** (walking model number i_w is not DS) **then**
        - Last model of preview horizon becomes walking model number i_w
        - $i_w \leftarrow i_w + 1$
        if $i_w = 2(T_d + T_s)$ **then**
            - $i_w \leftarrow 0$
        end if
        if First model of preview horizon is DS **then**
            - Update reference feet trajectories along horizon
        end if
    else if not(*IsWalking*) **then**
        if $i_s < T$ **then**
            - Last model of preview horizon becomes standing model number i_s
            - $i_s \leftarrow i_s + 1$
        end if
    end if
    - Perform one DDP iteration, publish optimal policy
end while

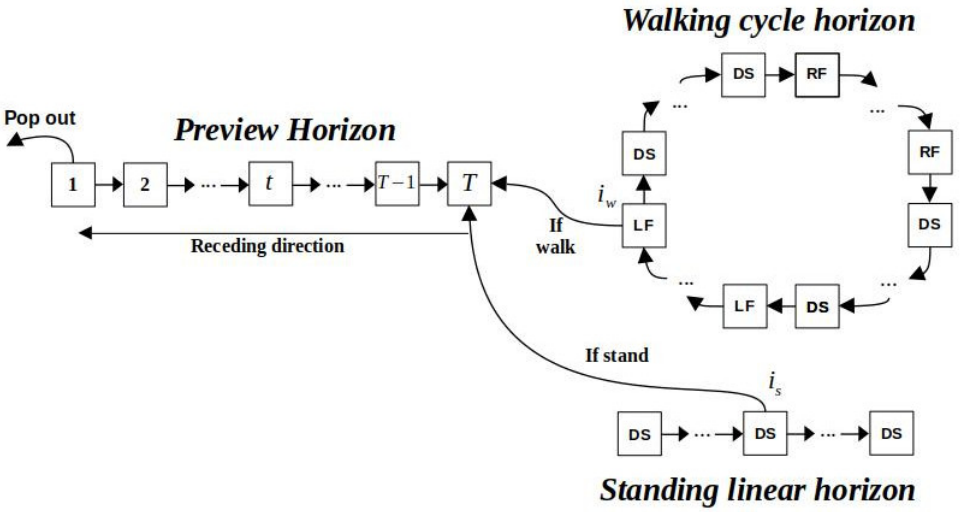

FIGURE 5.3: Receding horizon strategy combining a preview horizon, a walking cycle horizon, and a standing horizon. DS stands for Double Support, LF for Left Foot support, RF for Right Foot support.

are colinear to the normal of the contact surface. As a consequence, the entire foot area is making contact with the ground at the same time.

Empirically, we found out that fixing $\delta z = 1$ cm leads to a more stable walk on flat ground. This amounts to triggering the contact landing event when the foot reference is 1 cm above the ground. Doing so tends to make the foot go firmly against the ground, hence improving its stability. On the contrary, switching to double support after the foot actually made contact often results in motion failure, because the robot tries to go through the ground and thus applies a non-modeled force that can push it out of balance.

5.2.5 Robotic Operating System (ROS) architecture

Our MPC is embedded inside a ROS node that subscribes to the actual state of the robot measured at 2 kHz, and that publishes a feedforward optimal torque $\boldsymbol{u}_0$, Riccati gains $\boldsymbol{K}_0$ and last computed initial state $\boldsymbol{x}_0$ at 60 Hz. The final torque sent to the low-level control combines the feedforward control with a state feedback based on the Riccati gains and last measured state $\boldsymbol{x}^*$:

$$\boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{K}_0(\boldsymbol{x}_0 - \boldsymbol{x}^*). \tag{5.6}$$

This ROS scheme is identical to the one presented in Fig. 4.2, except that there is no MOCAP node or feedback on the desired end effector position. As before, the desired intensity of current at each controlled joint motor is computed through a proportional-derivative feedback on the joint torque measurement.

5.3 Experimental results

Our MPC locomotion framework was extensively tested on the humanoid robot Pyrène. Due to the critical problem of computation time, the MPC still runs on a powerful external computer (AMD Ryzen 5950X, 16 cores and 4.9GHz with 64 GB of RAM), whereas the low-level control runs on the embedded CPU of the robot.

The proposed control scheme was evaluated in two locomotion scenarios: in the first one, the robot performs a straight walk on flat floor with two sets of gait timings; in the second one, the robot climbs up and down a 10 cm-high step. In both experiments, the exact same cost weights were used on the robot, and the only varying parameters were the feet trajectory and gait timings. Since the regularization weights for the base pose have been set to 0, the robot is free to translate vertically and horizontally. This demonstrates the adaptability of the MPC framework, which can produce relevant stable trajectories over a wide range of 3-D dynamic motions. A video of all the experimental results is available at https://gepettoweb.laas.fr/articles/dantechumanoid22.html.

The following table presents the different gait parameters used during experimental validation.

	Gait 1	Gait 2	Stairstep
Forward step length δx	10 cm	20 cm	30 cm
Width btw. feet δy	20 cm	20 cm	20 cm
Single support time	1.1 s	1.5 s	3 s
Double support time	0.3 s	0.75 s	2 s

TABLE 5.3: Gait parametrization for the locomotion experiments.

5.3.1 Walking on flat floor

The results of the walking experiment are presented in figures 5.4 to 5.8. Additionnally, a snapshot of the 20 cm forward gait is displayed in Fig. 5.9. Fig. 5.4 shows the tracking of the desired feet trajectories over time. The Bezier curves used to generate the reference in position are only followed during swing phases, hence are not relevant during support phases. The reference in X-axis and Y-axis are precisely tracked during swing phases, but noisy oscillations can be observed at the edges of support phases, at take-off and landing. These oscillations are likely caused by inconsistent base pose estimation at contact switches, but could also be the consequence of a drift in calibration.

Figure 5.5 shows that a normal force is measured just before contact landing. This means that the foot impacts the ground before the solver predicts it. As no step or timing adaptation has been introduced in our framework yet, such behavior is for now difficult to avoid. Nevertheless, the MPC is able to cope with such unknown disturbances and produce a stable walking trajectory, thanks to its intrinsic compliance which helps to damp the impact.

Figures 5.6 and 5.7 show that the predicted CoP, computed from the forces predicted by the solver, matches the CoP based on the sensor forces. During the motion, the solver generates its own CoP trajectory to minimize the user-tuned cost function. It is interesting to notice that when the foot lands, the predicted CoP in Y axis may sometimes go outside the feet support area, although the real CoP remains inside. In this case, the low-level torque feedback loop is acting as a low-pass filter that smooths the control sharp edges.

Ideally, the CoP should stay close to the center of the support polygon, so as to maximize its robustness margins. In practice, the measured CoP goes very close to the front edge of the feet, as can be seen in Fig. 5.6. This behavior results from the weak regularization of contact wrenches, which does not prevent the CoP from crossing user-defined limits. One possible way to improve the locomotion would then be to compute a better reference for the CoP, adapted to each gait, using a centroidal model; however, this would come with an additionnal layer of complexity and would contradict the initial motivation based on the idea of encompassing the whole locomotion inside one unique control block. Another solution would be to use a solver able to tackle strict inequalities on forces.

Finally, Fig. 5.8 indicates that one iteration of DDP is computed in less than 15 ms, but features brutal spikes of 50 ms that happen at contact switches, when the structure of the problem suddenly changes. We do not know yet what causes these significant delays, but we suspect that bad CPU cache management is involved in some way. Despite these unexpected time overflows, the motions produced by our framework remain stable because the approximated Riccati policy takes over while the high-level control is computing.

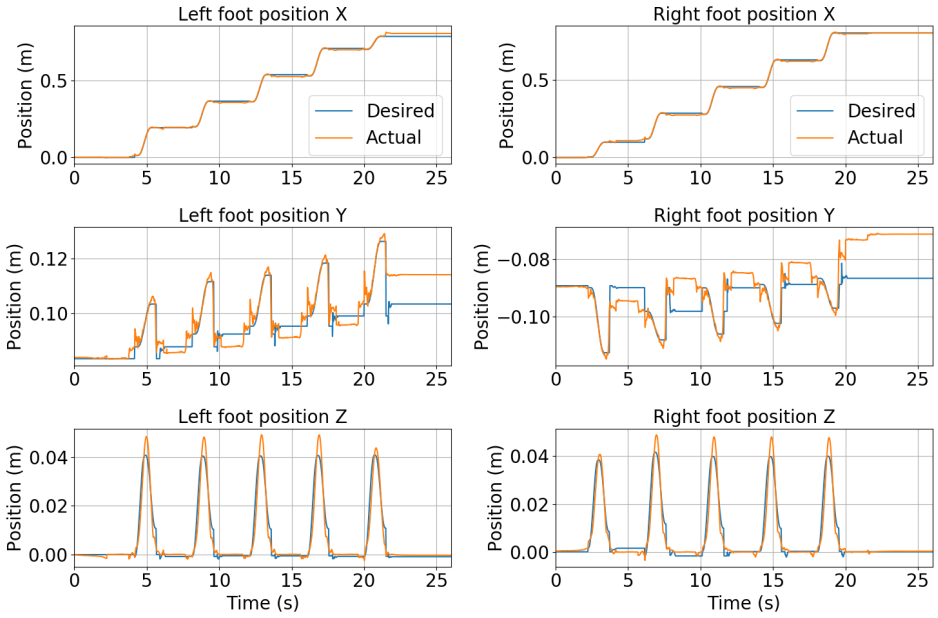

FIGURE 5.4: Measured and desired feet position during locomotion on flat floor.

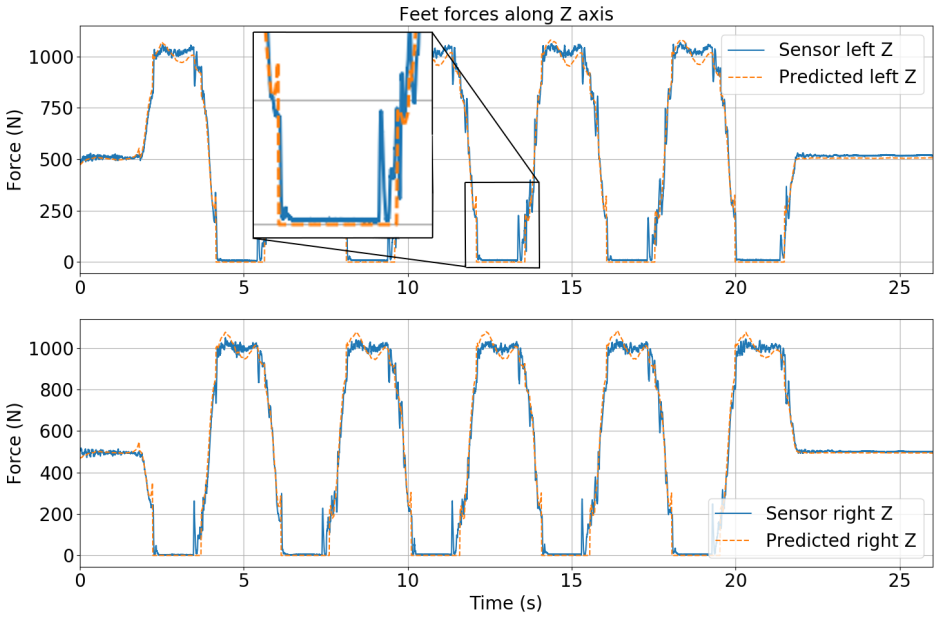

FIGURE 5.5: Predicted vs measured normal forces in left and right feet during locomotion on flat floor. The foot is touching the ground before landing as can be seen in the zoomed window.

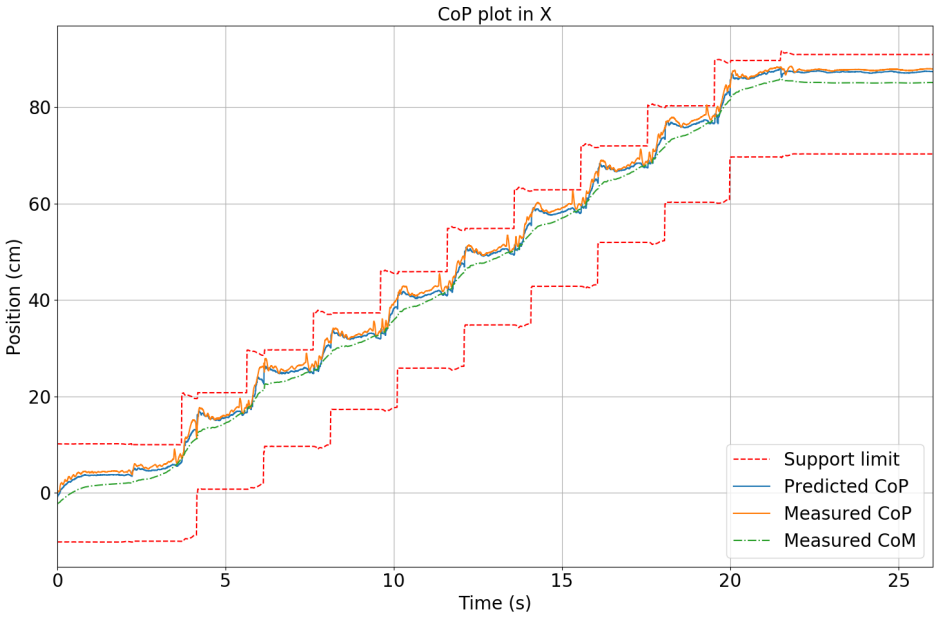

FIGURE 5.6: CoP trajectory in X axis compared to actual feet position during locomotion on flat floor. The foot of the robot is 20 cm long.

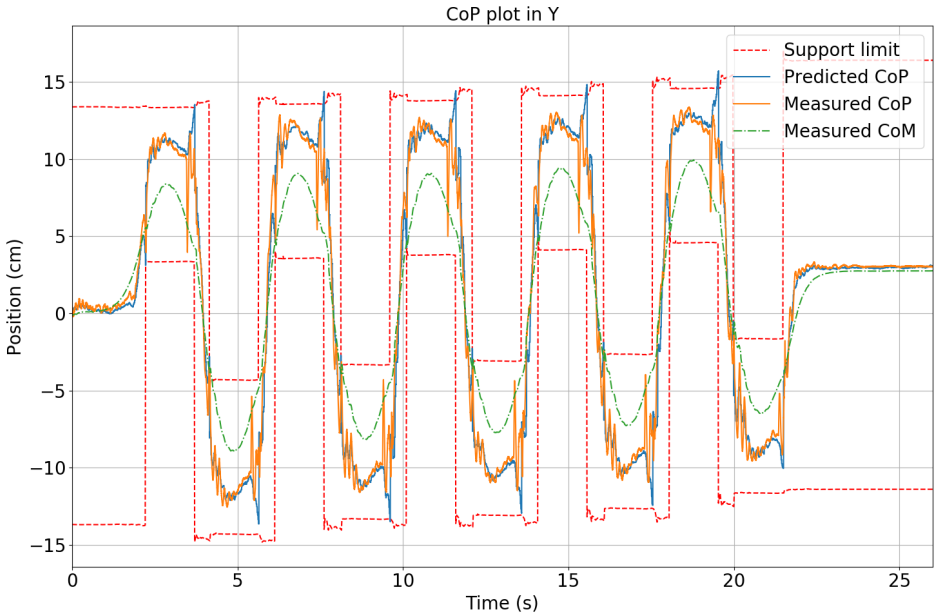

FIGURE 5.7: CoP trajectory in Y axis compared to actual feet position during locomotion on flat floor. The foot of the robot is 10 cm wide.

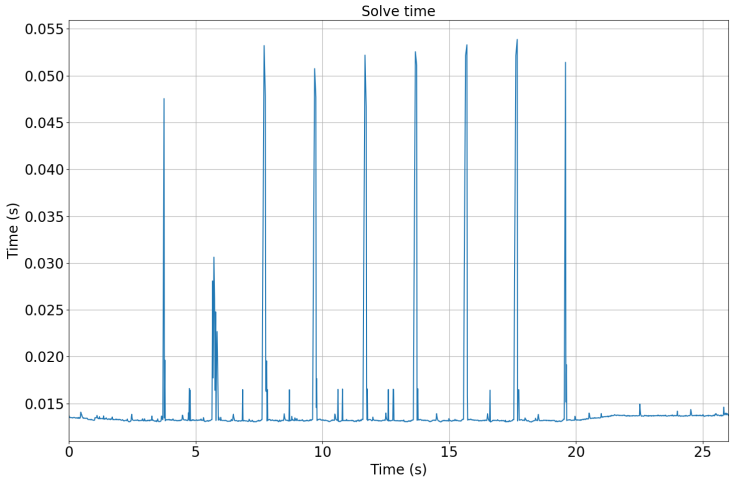

FIGURE 5.8: Time computation of one DDP iteration during locomotion on flat floor.

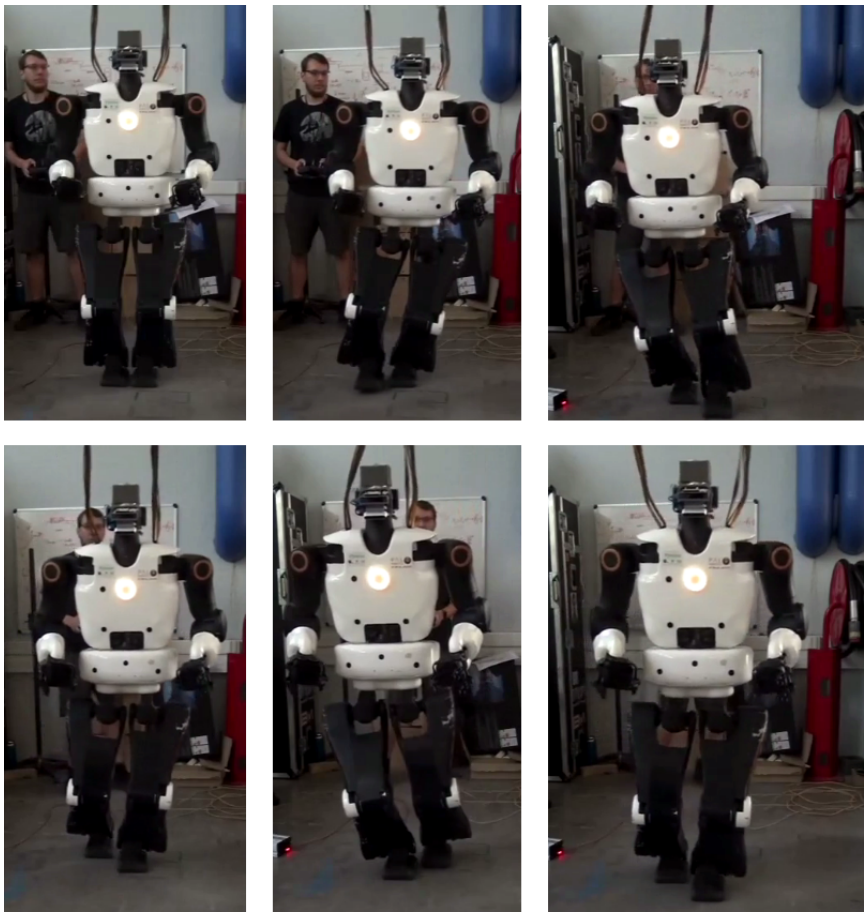

FIGURE 5.9: Snapshot of the walk experiment with 20 cm step. As the walk is quasi-static, the robot tries to keep its CoM as much as possible over the supporting foot in swing phases, generating high roll motions.

.

5.3.2 Stairstep crossing

The main objective of the stair experiment is to demonstrate the versatility of the proposed framework. Compared to the walking experiment, only the feet trajectories and contact timings have been modified, while the wrench trajectory remains the same. Figures 5.10 to 5.13 illustrate the results of this experiment, while Fig. 5.14 shows a snapshot of the stairstep crossing motion. Given the gait timings (3 s for double support, 2 s for simple support), one full step does not hold inside our preview horizon: as a consequence, the motion is quasi-static since the predictive control works in a nearly blind way, without being able to see contact landing before the foot takes off. Shorter gaits timings were attempted on simulation but were considered too hazardous to be passed on the robot. Walking the stairstep in a dynamic way remains a challenging task for a whole-body MPC subjected to very harsh computation constraints.

Figure 5.10 highlights an essential issue caused by a bad estimation of the base of the robot: during the last 5 seconds of the motion, whereas the right foot is in support phase and not moving, the actual foot position in Y shows brutal discontinuities of 1 cm when the left foot is creating and breaking contact. The estimated base of the robot shows the exact same discontinuities at the same time. The low quality of the base estimation negatively affects the feet tracking task, and as a consequence feet may be far above the ground during transition from single support to double support. This results in important impacts at contact transition, as can be seen in Fig. 5.13, where a spike force of 800 N is measured at $t = 24$ s.

Similarly to the flat floor experiment, the predicted CoP trajectory briefly goes outside the foot area when landing occurs, as seen in Fig. 5.12. This issue is also noticeable in Fig. 5.13 where the lateral forces during contact switches feature several peaks corresponding to contact mismatches caused by blind time-scheduled transitions and model uncertainties.

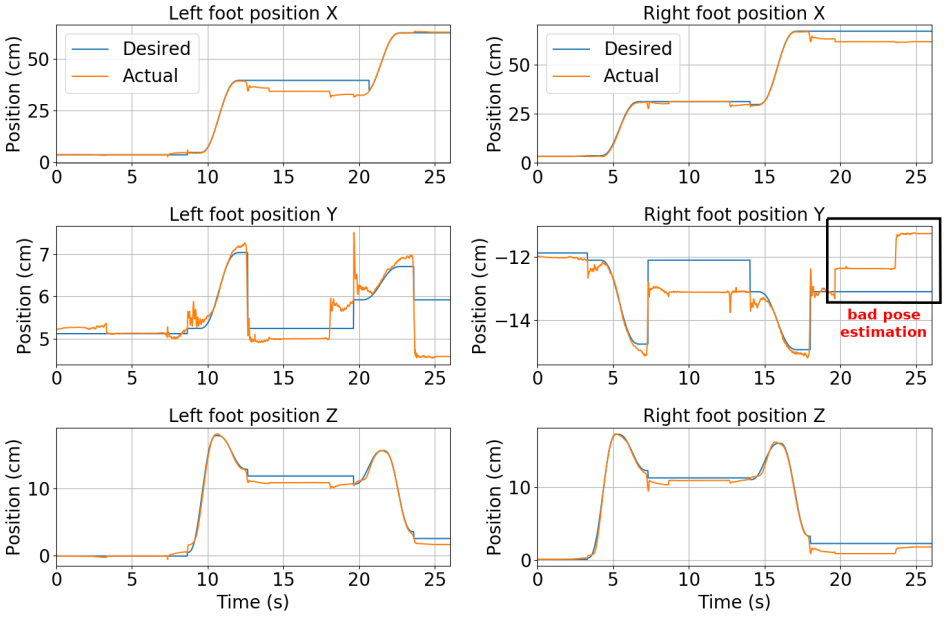

FIGURE 5.10: Measured and desired feet position during stairstep crossing. Bad pose estimation is highlighted at the end of the right foot trajectory.

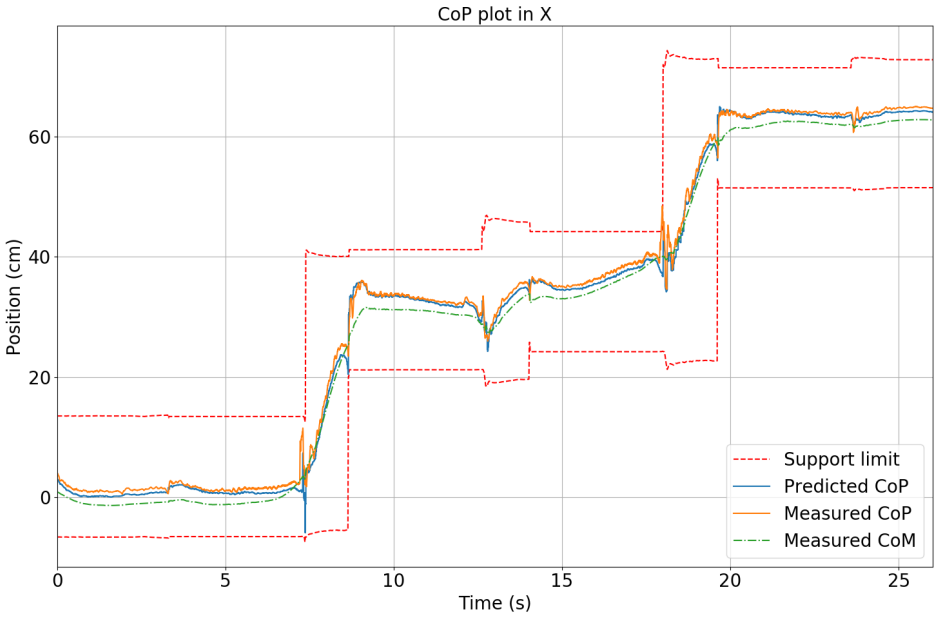

FIGURE 5.11: CoP trajectory in X axis compared to actual feet position during stairstep crossing. The foot of the robot is 20 cm long.

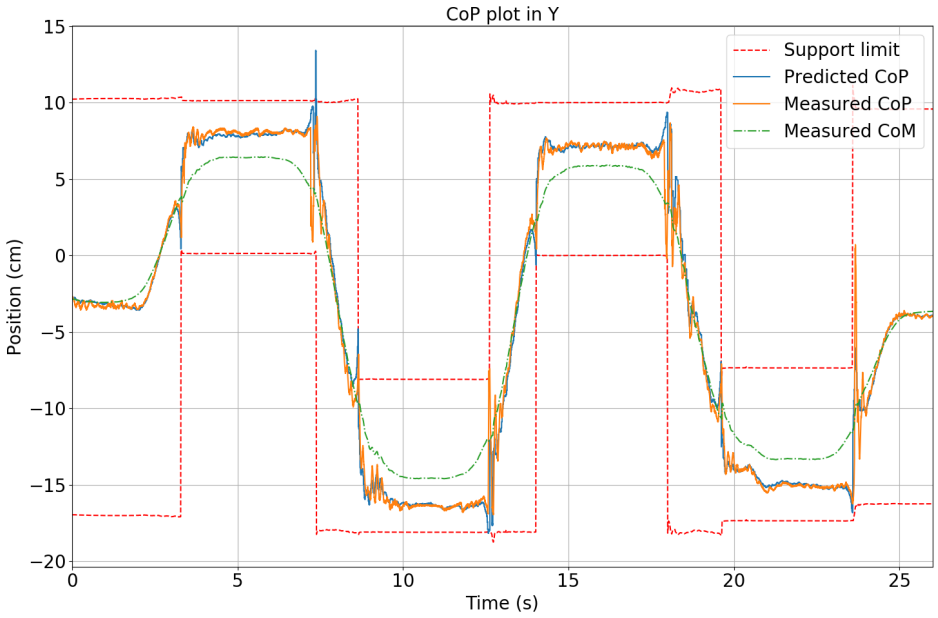

FIGURE 5.12: CoP trajectory in Y axis compared to actual feet position during stairstep crossing. The foot of the robot is 10 cm wide.

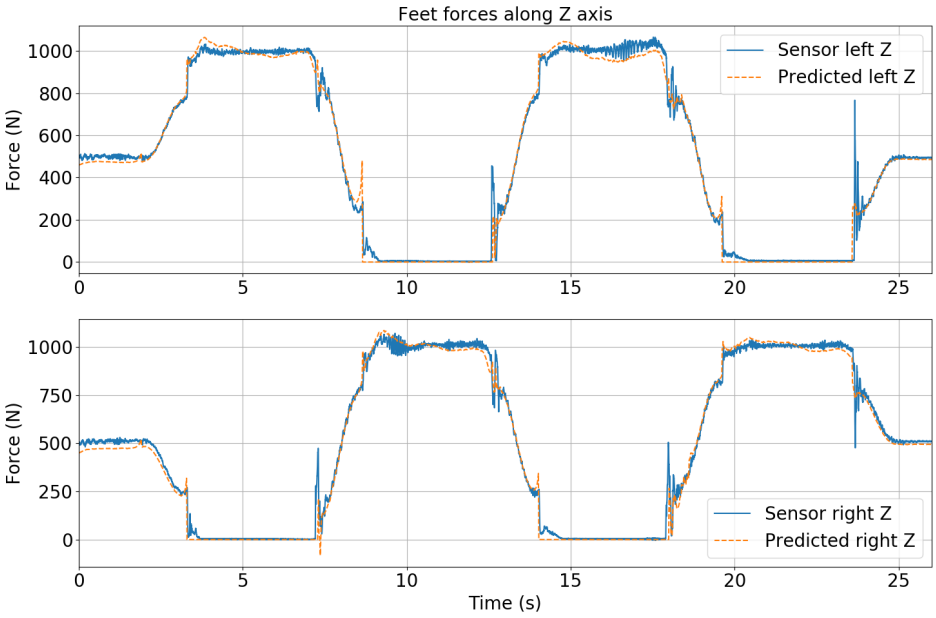

FIGURE 5.13: Predicted vs measured normal forces in left and right feet during stairstep crossing.

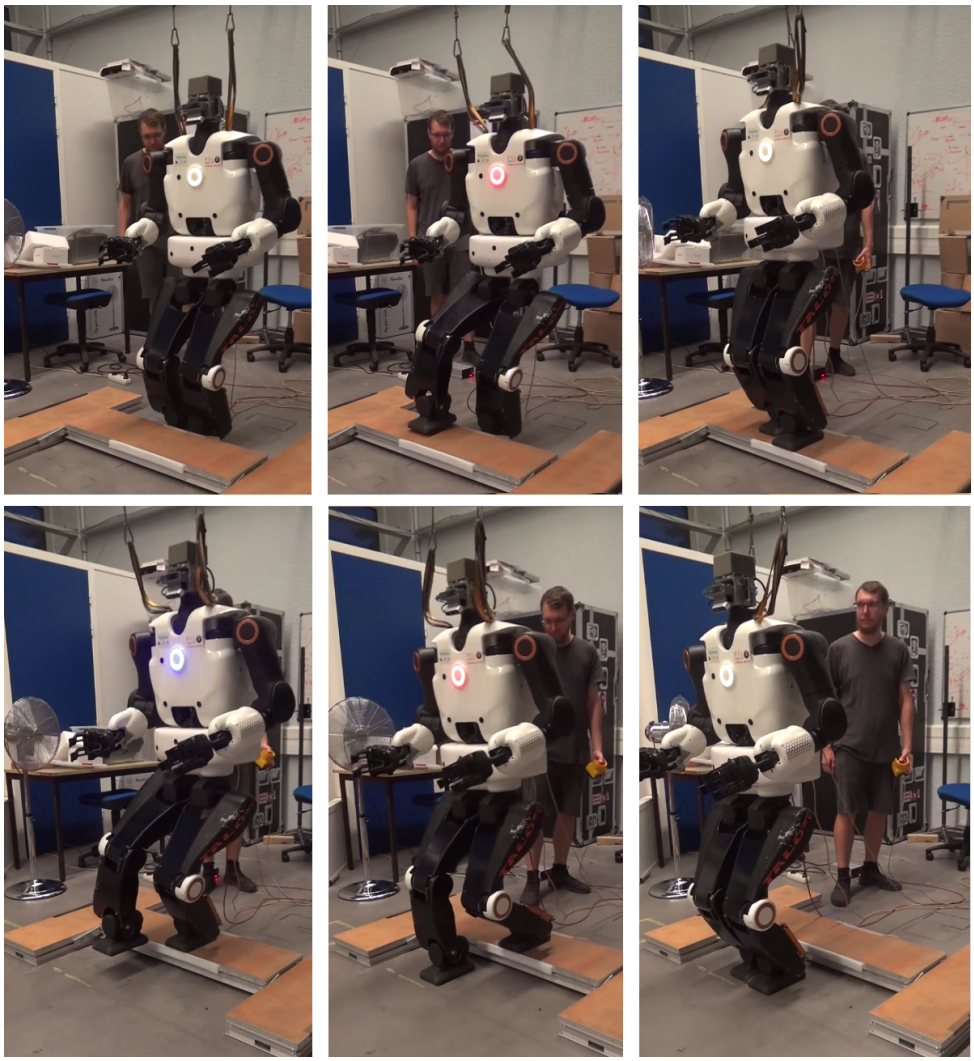

FIGURE 5.14: Snapshot of the stairstep crossing experiment.

5.4 Conclusion

This chapter introduced the first application of whole-body MPC for the locomotion of a biped robot. The proposed framework can be adapted to cross obstacles like stairstep by simply adjusting the feet reference trajectories and locomotion gaits. It has proven to be robust against model uncertainties, imprecise state estimation and unexpected latency in data transmission, caused for example by contact switches. The key to implementing this MPC on a real platform lies in the approximation of the wrench cone constraints treated as regularization costs and the use of Riccati-based feedback policy inside the low-level control of the robot. The latter improvement allows us to increase the frequency of the control scheme to 2 kHz. Without this additional state feedback loop, the motion could not be executed on the real robot, as the update frequency would not be high enough. Our single-block framework can be extended to any given task at the (small) price of designing and tuning specific costs adapted to the goal.

Although this whole-body MPC architecture produced promising results, we still need to prove that it can outperform state-of-the-art centroidal walking controllers. The proposed method used to cope with wrench cone constraints results in

dynamically unfeasible predicted CoP trajectories, even if the filtering effect of the low-level control allows for such solutions to be executed on the robot. Moreover, robustness margins cannot to be implemented inside the current framework, resulting in very conservative motions for challenging tasks like walking on uneven terrain. The behavior of the robot at contact transitions needs to be studied more thoroughly, so as to eliminate those brutal discontinuities in torques and forces. All in all, the curse of dimensionality remains an overwhelming constraint that our Riccati-based feedback scheme cannot attenuate entirely. On this matter, it would have been interesting to integrate the arm joints to the problem formulation, in order to benefit from additionnal DoF to control the total angular momentum of the system. Unfortunately, our current computer configuration does not have enough power yet to handle such a complex model.

A significant limitation of the proposed scheme lays in the high impedance of the feet tracking task. Walking as demonstrated in this chapter amounts to guide the robot by its feet like a puppeteer, whereas it would be preferable to give the robot the ability to decide autonomously how to move in its environement. Besides, push recovery cannot be done while tracking user-defined foot references, unless a fast high-level footstep planner provides stabilizing contacts in case of strong disturbances. These limitations will be tackled in the next chapter of this thesis, where a trajectory-free MPC scheme will be developed.

Chapter 6

Walking without thinking

In the previous chapter, a walking scheme based on the use of pre-defined foot trajectories has been implemented on the robot Talos. Part of the locomotion complexity is captured by these high-level references that the user should tune depending on the environment they wish to cross. Yet, this approach implies to tune a stiff cost to accurately track the references, which artificially increases the controlled impedance of the flying foot. We seek now to write a trajectory-free algorithm in which the solver can decide by itself where the robot feet should go in order to walk safely on cluterred terrain. The aim is to reduce the impedance of the controller with respect to end effector placements and to make the formulation more generic.

6.1 Motivation

Classical control methods for locomotion involve tracking a stable reference trajectory with QP or MPC schemes, depending on the problem size. If such approaches have produced satisfying results on numerous robotics platforms, they are in general not suited to face strong external disturbances. The work of [Wieber 2006b] shows that a trajectory-free MPC scheme is able to reject stronger disturbances than a control scheme that simply tracks a user-defined CoP reference. Instead of trying to follow at any cost an outdated CoP trajectory, it is way more relevant to recompute a new one in order to recover from an unexpected push. To reach this objective, the cost function of the optimal formulation in [Wieber 2006b] is set to minimize the CoM jerk while maintaining the CoP inside the support polygon. Following this approach, the work of [Herdt et al. 2010] proposed to enhance the flexibility of the walking scheme by letting the solver choose where to step on. This work advocates for the idea of a *walking without thinking* formulation of locomotion, where the user provides a desired CoM reference (final position or current velocity) along with a set of feasible regions to step on, or a sequence of contact points provided by a contact planner. In [Herdt et al. 2010], feasible regions for the next supporting foot are computed offline and derived online under the form of simple linear approximations of the robot kinematic limits. In case of strong pushes, the robot is able to keep its balance as long as it can stop its motion within one step. For even stronger pushes, computing the capture region [Pratt et al. 2006] and ensuring that the robot reaches it within a few steps may constitute a viable solution to avoid falling.

In order to enhance the reactivity and adaptability of our control scheme, we aim at transposing this *walking without thinking* approach to our whole-body MPC

formulation. The objective is to execute a high-level locomotion task (go to a given point across a cluttered terrain) with a minimal set of user-defined inputs (typically only the terrain structure and the desired final pose). In the work of [Herdt et al. 2010], real-time capacities are achieved through clever linear simplifications, paving the way for a linear centroidal MPC that can adapt contact placements online. In the context of whole-body predictive control, it implies that the solver must decide both the impact location and the trajectory of the feet in flying phase. The computational burden increases as the OCP must now encompasses some terms to push the flying foot away from collision, in coordination with the centroidal behavior. In this chapter, we propose to use a new cost introduced in [Assirelli et al. 2022] for quadruped locomotion without foot references. We adapt the walking OCP formulated in Chap. 5 by including this cost and experimentally validate its feasibility and interest on the robot Pyrène.

6.2 Trajectory-free MPC formulations

This section extends the walking controller presented in Sec. 5.2. The structure of the controller remains the same, and only some few changes are made to the cost functions in charge of shaping the desired locomotion solution. More specifically, five cost terms remain unchanged with respect to the previous chapter: state and control regularization costs (ℓ^x and ℓ^u), kinematic limit cost (ℓ^b), wrench tracking cost (ℓ^λ) and CoP regularization cost (ℓ^{cop}). These terms relate to generic locomotion in any kind of terrain.

Two different trajectory-free control formulations are examined in the frame of this thesis. The first one is designed to withstand strong external disturbances on flat floor while minimizing the distance to a user-defined CoM goal. It supposes that the floor is an infinite flat walkable surface, and as a consequence does not need to be provided with feasible stepping areas or feasible contact points. The second one aims at crossing cluttered environments like stepping stones or stairs, without the requirement to compute splines joining each contact placement. As for the second one, a high-level contact planner combined with a map of the environment is needed to define a feasible contact sequence across obstacles.

Both formulations leverage the receding horizon strategy introduced in Sec. 5.2. They also share the same reduced robot model composed of 14 actuated joints, 12 for legs and 2 for torso. We stick to the same time parametrization as used on the previous walking OCP: the preview horizon is made of 150 knots separated by a 10 ms timestep.

6.2.1 Push recovery formulation

Our push recovery formulation is based on the idea of reducing the foot controlled impedance of the walking scheme. Foot controlled impedance can be viewed as the relation between a variation in foot position and the corresponding control correction. From this perspective, the walking scheme introduced in Chap. 5 displays a very stiff foot impedance, because the weight of the associated tracking cost is very high compared to the overall weight distribution. As a consequence, the MPC penalizes very harshly any deviation from the reference foot trajectory. This means that the robot cannot leave the path given by the user, unless a high-level trajectory planner is implemented to adapt to any environmental variation or command update. Moreover, absorbing unexpected strong pushes is impossible, as the

resulting error in foot tracking would lead to a spike in control response that would trigger the actuation safeguards.

On the contrary, a lower foot controller impedance will enhance the compliance and the flexibility of the scheme, allowing to perform push recovery with very minor changes in the control law. On flat ground, the impedance can be relaxed along both horizontal directions, except when the feet are close to each other. In this precise case, a geometric barrier cost must be implemented to avoid foot collision.

Placement foot costs

Theoretically, the flying foot altitude needs to be at ground level with horizontal orientation at the beginning of each contact phase. Due to the inherent compliance of the MPC scheme, the distance between ground and foot can be a little higher than zero at contact time: this tends to favor stability at contact transitions, as observed in Chap. 5. Following this idea, we seek to penalize the foot altitude with the following cost term:

$$\ell^z(\boldsymbol{x}) = a_L(z(\boldsymbol{x})), \tag{6.1}$$

with $z(\boldsymbol{x})$ height of flying foot when the robot state is $\boldsymbol{x}$, and a_L logarithmic activation function introduced in Sec. 4.3.1. This cost is only active during double support phases. Its goal is to bring down the foot at the end of the flying phase, so as to mitigate the impact event.

Additionally to this cost term, we define a rotation cost function $\ell^{\mathbf{R}} = ||log(\mathbf{R})||^2$ whose role is to bring the rotation of the flying foot equal to zero. In future work, the rotation reference could be set to change according to the user's needs, but for the sake of the demonstration we consider it fixed for now.

High-fly cost

Without foot reference to track, we need a new heuristics to avoid collision with the ground during flying phase. The foot should follow a bell curve when making a step forward, so that it does not scratch against the ground. To tackle this requirement, we consider the following intuition, introduced in [Assirelli et al. 2022]: foot scratching is caused by high lateral foot velocity combined with low or null foot height. Penalizing lateral foot velocity with respect to foot height appears to be a logical heuristics to deal with this kind of motion. Our high-fly cost would then write:

$$\ell^{fly}(\boldsymbol{x}) = ||\boldsymbol{v}^{x,y}(\boldsymbol{x})e^{-\gamma z(\boldsymbol{x})}||^2, \tag{6.2}$$

with $\boldsymbol{v}^{x,y}(\boldsymbol{x})$ lateral foot velocity (coordinates x and y) and $\gamma = 300$ a weighting factor related to the sensibility to foot height. For an empirical viewpoint, the use of exponential to penalize lateral velocity has been observed to produce good results as compared to linear or quadratic norms.

To better visualize the effect of this cost, one might take a look at Fig. 6.1, which shows the gradient of a simplified 2-D version of ℓ^{fly}. The solver will tend to explore the opposite direction of the plotted gradient, and as a consequence will favor high

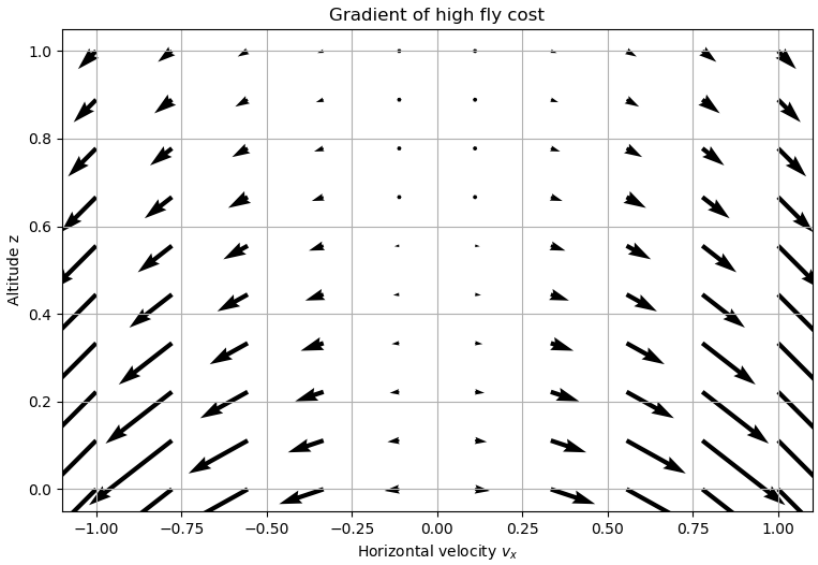

FIGURE 6.1: 2-D plot of the high-fly cost gradient with respect to altitude and lateral velocity. The vectors are pointing toward the direction of steepest increase. For the sake of the demonstration, the cost plotted here is simplified as: $\ell(v_x, z) = v_x^2 e^{-2z}$.

foot altitude when the absolute lateral velocity is high.

Collision avoidance cost

Because the foot position is unconstrained during flying phase, self-collision must be actively avoided. Those self-collisions are very likely on the Pyrène robot, because the CoM of its legs is externally shifted by a few centimers with respect to the roll joint. As a consequence, the natural resting position of the legs, obtained when the robot is lifted and no static friction is considered, results in feet collision. To prevent this behavior, a self-collision avoidance cost is written as:

$$\ell^{col}(\boldsymbol{x}) = \sum_{i=1}^{N_l} \sum_{j=1}^{N_r} b(||\boldsymbol{r}_i^l - \boldsymbol{r}_j^r||). \tag{6.3}$$

In this cost, $N_l = N_r = 3$ stands for the number of collision points on left and right foot frames: toe, center and heel of the foot. Likewise, $\boldsymbol{r}_i^l$ and $\boldsymbol{r}_i^r$ are the collision point translations for left and right frame. This makes for a total of 9 collision pairs. Finally, $b : s \mapsto ||max(d_m - s, 0)||^2$ is a quadratic barrier cost that gets activated only when the distance between frames goes below a given threshold $d_m > 0$. On Pyrène, this threshold has been taken equal to 20 cm, a bit wider than the natural distance between feet in half-sitting configuration.

CoM position cost

This cost encompasses the user-defined position of the robot at the end of its motion. It is similar to the CoM tracking cost presented in Sec. 4.3.1, save for the fact

that we use a smoothed absolute value norm instead of a quadratic one. Given $\boldsymbol{s} = (s_1, s_2, s_3) \in \mathbb{R}^3$, this pseudo-norm, which we also call activation function, writes:

$$a_1(\boldsymbol{s}) = \sum_{i=1}^{3} \sqrt{s_i^2 + \epsilon}. \tag{6.4}$$

An infinitesimal parameter ϵ is added in order to make the activation differentiable at the origin. In practice, we work with $\epsilon = 0.01$. Denoting $\boldsymbol{c}^*$ to be the desired final position, the CoM cost is then:

$$\ell^{com}(\boldsymbol{x}) = a_1(\boldsymbol{c}(\boldsymbol{x}) - \boldsymbol{c}^*). \tag{6.5}$$

The aim of $a_1(\cdot)$ is to prevent the cost gradient from becoming too high when the desired position goes far away from the current one. Thanks to this pseudo-norm, the gradient remains approximately constant with respect to the CoM task residual.

Putting this CoM tracking cost in the running model of our OCP would tend to produce brutal motions toward the goal $\boldsymbol{c}^*$, which can be unreachable within the span of the horizon. Besides, it would put too much constraint on the CoM motion, whereas the goal is to maximize the flexibility of the formulation. For these reasons, the CoM cost is only added to the terminal model of our walking OCP.

CoM velocity cost

In order to smooth the CoM trajectory and avoid brutal motions toward the end goal, we have added a regularization term on CoM velocity:

$$\ell^{vcom}(\boldsymbol{x}) = ||\dot{\boldsymbol{c}}(\boldsymbol{x})||^2. \tag{6.6}$$

It should be noted that this cost also tend to prevent falls at the end of the horizon. Indeed, falls are characterized by a loss of balance and the CoM velocity rapidly diverging until ground impact. However, if combined with a too high weight, this cost can hinder the natural oscillation of the CoM from one supporting foot to another.

6.2.2 Stairs formulation

The previous OCP is limited to flat ground locomotion and cannot consider obstacle avoidance because it does not integrate information about the robot surroundings. Since it supposes that the entire ground is a stepping surface, it does not require a contact planning algorithm. In the case of cluttered terrains, however, it is crucial to ensure that the sequence of contacts belongs to a set of stable surfaces. As a consequence, some stiffness must be injected again into the control of the foot placement at the end of flying phases, to prevent landing at forbidden locations. Nevertheless, we still aim at generating motions from previous to next contact without relying on hand-crafted references, by defining a generic high-fly cost that can adapt to the terrain structure.

As it would be too difficult to solve for contact locations and whole-body control at once, we need to rely on an external contact planner that will provide optimal foot placements in real-time. Given only the next contact to reach and the switching instant, the control block must compute a stable whole-body policy to achieve the desired task.

Final placement foot cost

Let us suppose that at each instant, the placement of the next contact $\boldsymbol{p}^* \in \mathbf{SE(3)}$ is made available to the solver. Our placement cost simply writes:

$$\ell^p(\boldsymbol{x}) = a_L(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}^*), \tag{6.7}$$

with $\boldsymbol{p}(\boldsymbol{x})$ current placement of the flying foot and a_L the logarithmic activation function defined in Sec. 4.3.1 for hand tracking. As in Chap. 5, this cost is only active during flying phase of current foot and double support phase, then get deactivated during the flying phase of the opposite foot. Final placement $\boldsymbol{p}^*$ remains constant during flying phase and is only updated during double support phase, in order to avoid brutal trajectory changes while the robot stands on one leg.

Similarly to the walking formulation of Chap. 5, the rotation reference in $\boldsymbol{p}^*$ is kept equal to the identity so as to freeze the angular motion of the flying foot. The translation part of $\boldsymbol{p}^*$ will be provided by a high-level contact planner which reasons over the feasible set of contact surfaces in the environment.

High-fly cost

Let us suppose that the next contact is located on top of a stairstep, such that a direct trajectory from current to next contact will result in the foot colliding with it. If we introduce a collision cost between the stairs and the foot, the OCP to solve is geometrically non-convex. This problem formulation has been studied on simulation but resulted in inconsistent motions, because the DDP algorithm could not cross the obstacle with only one iteration per control cycle. No good compromise could be found between the vertical repulsive field of the stairs and the attracting point of the desired contact.

For this reason, we have considered a generalization of the high-fly cost introduced for flat ground walking to hint a collision-free path to the solver. This idea partly stems from the concept of local potential fields introduced in [Khatib 1986] to deal with collision avoidance. Rather than penalizing the position of the end effector when it comes close to obstacles, we penalize here its velocity toward the obstacle. Our approach amounts to smooth the height map of the environment to seamlessly guide the solver in the region of most interest.

Let us introduce the sigmoid activation function, a S-shaped curve function mostly used in the context of artificial neural network. In the context of kinematic planning, this function will act as a smooth interpolation of an angular stairstep. The sigmoid and its first derivative are defined by:

$$\begin{aligned} s(x) &= \frac{h}{1 + e^{-(x - x_h)/w}} \\ s'(x) &= \frac{s(x)(h - s(x))}{wh}, \end{aligned} \tag{6.8}$$

with h height, w width of the sigmoid and x_h the x-coordinate of half-height. This function is of class $\mathbf{C}^\infty$ and its derivatives are very simple, leading to cheap gradient computation. It tends toward h at infinity and toward 0 at minus infinity (see Fig. 6.2).

Given a height collision map represented by a sigmoid function, we need to compute the distance between the sigmoid curve and the flying end effector. For the sake

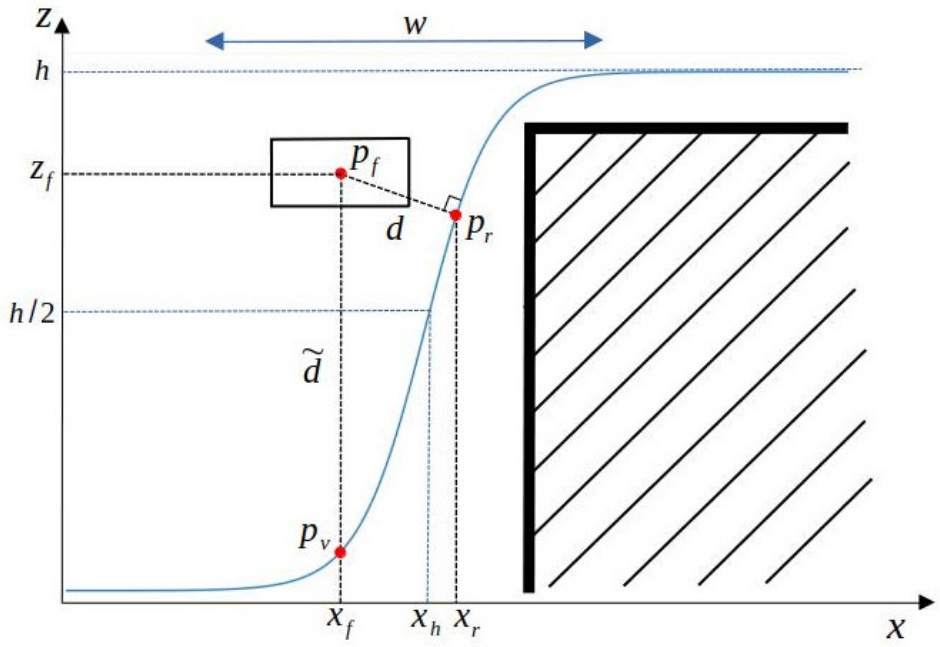

FIGURE 6.2: Sigmoid height map (in blue) interpolating a stairstep obstacle (in black). The rectangular shape is the robot foot. $\boldsymbol{p}_f$ is the foot position, $\boldsymbol{p}_r$ the witness point on the sigmoid closest to $\boldsymbol{p}_f$, and $\boldsymbol{p}_v$ is the point of the sigmoid at x_f.

of simplification, the following considerations will take place in 2-D, but the same mathematics principles apply in the 3-D world.

We define $\boldsymbol{p}_r = (x_r, s(x_r)) \in \mathbb{R}^2$ to be the witness point of the foot, that is to say, the point on the sigmoid closest to $\boldsymbol{p}_f = (x_f, z_f) \in \mathbb{R}^2$, the foot position in world frame. Likewise, we defined $\boldsymbol{p}_v = (x_f, s(x_f)) \in \mathbb{R}^2$ to be the point on the sigmoid with the same horizontal coordinate as the foot (see Fig. 6.2). Finally, d stands for the Euclidean distance between $\boldsymbol{p}_r$ and $\boldsymbol{p}_f$:

$$\begin{aligned} d^2 &= \min_x \ \big(s(x) - z_f\big)^2 + \big(x - x_f\big)^2 \\ &= \big(s(x_r) - z_f\big)^2 + \big(x_r - x_f\big)^2 \end{aligned} \tag{6.9}$$

Unfortunately, no analytical formula of d or x_r can be found by classical methods, as it amounts to finding the root of a function mixing polynoms and exponentials. If we derive (6.9) with respect to x and nullify the gradient, we end up with a reformulation of x_r:

$$x_r = x_f + (z_f - s(x_r))s(x_r)\frac{(h - s(x_r))}{hw}. \tag{6.10}$$

Reinjecting this into the expression of the Euclidean distance, we obtain:

$$\begin{aligned} d^2 &= \big(s(x_r) - z_f\big)^2 + \Big((z_f - s(x_r))s(x_r)\frac{(h - s(x_r))}{hw}\Big)^2 \\ &= \big(s(x_r) - z_f\big)^2\Big(1 + \Big(s(x_r)\frac{(h - s(x_r))}{hw}\Big)^2\Big). \end{aligned} \tag{6.11}$$

This expression of this Euclidean distance is particularly difficult to deal with. An easier approach is to penalize the distance $\tilde{d} = |s(x_f) - z_f|$ between $\boldsymbol{p}_f$ and $\boldsymbol{p}_v$. Intuitively speaking, this should also hint the solver to raise the foot if it is coming too close to the sigmoid. When the foot is above the sigmoid curve (i.e. $z_f > s(x_f)$), it is immediate to notice that $z_f > s(x_r)$ as well, because the segment $(\boldsymbol{p}_f, \boldsymbol{p}_r)$ must be perpendicular to the tangent of the sigmoid curve at $\boldsymbol{p}_r$. From (6.9) we deduce that $x_r > x_f$ since $s(x)(h - s(x))$ is always positive, which allows us to state that $\tilde{d} > d$. It appears clearly that we overestimate the real Euclidean distance to the curve by making such an approximation. Nevertheless, our guess is that using a high weight on the high-fly cost or shifting the sigmoid forward with respect to the obstacle may mitigate the error caused by making this approximation. In any case, we write our cost term as:

$$\ell^{fly}(\boldsymbol{x}) = \boldsymbol{v}^{x,y}(\boldsymbol{x})e^{-\gamma\tilde{d}(\boldsymbol{x})}. \tag{6.12}$$

It is worth noting that in case $h = 0$, this cost amounts to the high-fly cost introduced in the flat ground formulation. This illustrates the genericity of the approach, which can handle different terrain configurations based on a smoothed velocity height map of the surroundings.

6.2.3 Cost parametrization

Regarding the push recovery formulation, the weight distribution for state regularization, control regularization and kinematic limits are set to be the same as in the formulation of Sec. 5.2. The other weights used on the robot are presented in Table 6.1. The cost on collision avoidance, foot height and foot rotation are particularly high to enforce them as constraints. The weight of the high-fly cost is also high to favor vertical foot motion during lateral translation.

	ℓ^z	ℓ^R	ℓ^λ	ℓ^{cop}	ℓ^{fly}	ℓ^{col}	ℓ^{com}	ℓ^{vcom}
Run. costs	1000	1000	0.001	10	1000	3000		500
Ter. costs	1000	1000			1000	3000	100	500

TABLE 6.1: Cost weights of the push recovery OCP on flat ground.

As for the stairs formulation, the weight distribution used on the robot is presented in Table 6.2. Again, weights associated to ℓ^x, ℓ^u and ℓ^b are identical to the ones used in previous OCP formulations. The weight on placement cost is high enough to attract the flying foot toward the next contact point, but low enough to not trigger brutal control spikes that may overshoot the robot inner securities.

	ℓ^p	ℓ^λ	ℓ^{cop}	ℓ^{fly}
Run. costs	200	0.001	10	1000
Ter. costs	200			1000

TABLE 6.2: Cost weights of the stairs climbing OCP.

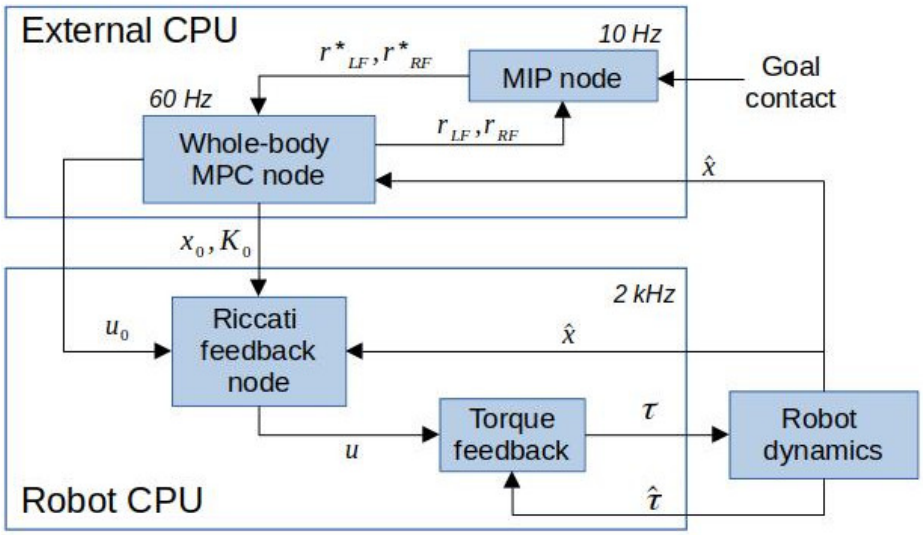

FIGURE 6.3: ROS architecture for locomotion scheme with contact planner.

6.2.4 ROS architecture with contact planner

The MPC node for push recovery and stairs climbing implements a similar version of the receding horizon algorithm introduced in Alg. 2. The only notable change is that there is no computation of the reference foot trajectories at each control cycle, but rather an update on the next desired foot position.

As for the push recovery formulation, the ROS architecture is similar to the one used in Chap. 5 for locomotion with feet references. The MPC node still computes the optimal torque, Riccati gains and last computed initial state at a frequency of $60 - 70$ Hz, then sends it to the low-level control running at 2 kHz.

Concerning the stairs formulation, a contact planner node has been added to the usual ROS scheme, as can be seen in Fig. 6.3. This new node receives the current feet placements $(\boldsymbol{r}_{LF}, \boldsymbol{r}_{RF})$ from the MPC node and computes the next eight feasible contacts which will bring the robot as close as possible to an user-defined goal contact position. In practice, since the OCP horizon only covers two steps at best, the final placement foot cost of our stairs formulation takes as reference input the next left and right planned contacts $(\boldsymbol{r}^*_{LF}, \boldsymbol{r}^*_{RF})$.

Our contact planner [Tonneau et al. 2019] runs at a frequency of approximately 10 Hz thanks to a Mixed-Integer Program (MIP) scheme based on the Gurobi Optimizer software [Gurobi Optimization, LLC 2023], a modern off-the-shell tool for solving complex combinatorial optimization problems. MIP approaches address the multi-contact planning problem on uneven terrain by performing continuous footstep optimization on convex surfaces while using integer variables to select the appropriate surfaces. Although mixing integer and continuous variables result in a

much harder planning problem, powerful commercial solvers like Gurobi can still provide global solutions in reasonable time. A MIP scheme was successfully deployed on Atlas to plan biped footstep sequences on stepping stones in [Deits and Tedrake 2014]. Similarly, MIP was integrated inside a complete multi-legged locomotion scheme based on centroidal dynamics in [Aceituno-Cabezas et al. 2018].

6.3 Experimental results

The proposed controller was validated through locomotion experiments on the robot Pyrène and on simulation. In both cases, the MPC node and contact planner node are running on a powerful CPU (AMD Ryzen 5950X, 16 cores and 4.9 GHz with 64 GB of RAM).

Our experimental setup aims at testing two assumptions: first, the trajectory-free locomotion scheme is flexible enough to perform push recovery in real time (A1); second, feet reference trajectory can be replaced by potential height map to deal with obstacles during uneven terrain crossing (A2).

6.3.1 Push recovery experiment (A1)

To validate (A1), the robot was set to walk on flat floor while being subjected to violent external disturbances. Gait timings were fixed at 0.2 s for double support time and 0.8 s for single support time. Given that the total horizon length is 1.5 s, this choice of timings allows to predict the dynamics of a full step plus half of one. The desired terminal CoM position of the robot was set to 1 m forward its initial configuration.

Experimental results for push recovery are presented in figures 6.4 to 6.7. Additionally, a snapshot of the experiment is displayed in Fig. 6.8. Figure 6.5 shows that the walking motion is dynamic, or in other words, the CoM trajectory in Y remains almost always outside of the support polygon. After approximately 10 steps forward, the robot base is pushed several times from different angles using a stick. The first push is particularly visible at $t \approx 17.5$ s on Fig. 6.4 and 6.5: it can be observed that the robot CoM brutally goes from 5 to -15 cm in Y axis and 100 to 110 cm in X-axis. After the push, the CoP goes near its boundaries in order to absorb the energy of the disturbance. The measured CoP trajectory oscillates a lot and even briefly goes out of the support polygon in Fig. 6.5, meaning that the contact foot is tilting. Despite nearly losing its balance, the robot manages to recover and to keep on moving forward. Figure 6.6 displays the same oscillatory pattern for the measured vertical force of the right foot. It also features a sharp spike of 200 N in lateral forces, corresponding to the moment when the robot dissipates the energy coming from the disturbance.

It can be noticed on Fig. 6.6 that the left foot force measures along X-axis do not match the solver predictions. Moreover, the force profile is very different from the right foot measures along X-axis. One possible explanation for this observation is that the left foot force sensor is badly calibrated along the X direction. Another interesting insight is that the measured weight of the robot is slighlty higher than the predicted one, as can be seen on left foot force measures along Z-axis. There is a clear discrepancy between our dynamics model and the real physics of the robot.

Starting from $t = 10$ s, small vertical forces are exerted on the foot in flying phase in Fig. 6.6. This means that the robot is dragging its feet toward the end goal. The high-fly cost actually does not prevent the flying foot to make contact with the

ground when lateral velocity is insufficient. As a consequence, when the robot is close to its desired CoM, its feet tend to brush against the ground.

A second push is observable at $t \approx 22$ s and makes the robot overshoot its CoM target located at ($x = 100$ cm, $y = 0$ cm). After these disturbances, the robot slowly goes back to the desired pose, without reaching it completely. To achieve better tracking precision, the weight on the final CoM position task may be increased relatively to the user's needs. However, this gain in precision could hinder the whole compliance of the system and result in abrupt motions.

At the beginning of the experiment, the robot also drifts toward the left (see Fig. 6.4). This behavior may be caused by bad kinematic or dynamic calibration, as it has been observed on the Pyrène robot in Chap. 5 and with other walking controllers based on centroidal dynamics. Due to successive impact events, it is frequent to observe a drift in the torque offsets of the low-level control, which then requires a recalibration process.

Finally, the computation load of the experiment is presented in Fig. 6.7. Average time computation matches the one obtained during locomotion with foot references in Chap.5. Similarly, time spikes of up to 50 ms are visible at contact transitions, but they do not affect the overall quality of the motion. The push event at $t = 17.5$ s is barely noticeable on the time plot.

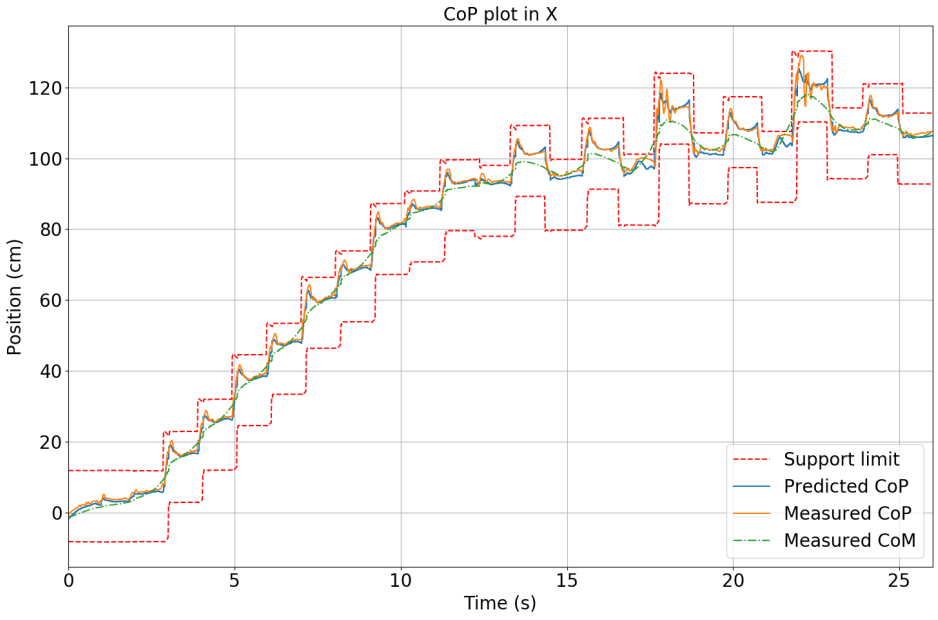

FIGURE 6.4: CoP and CoM trajectories along X axis for the push recovery experiment on flat floor. The foot of the robot is 20 cm long.

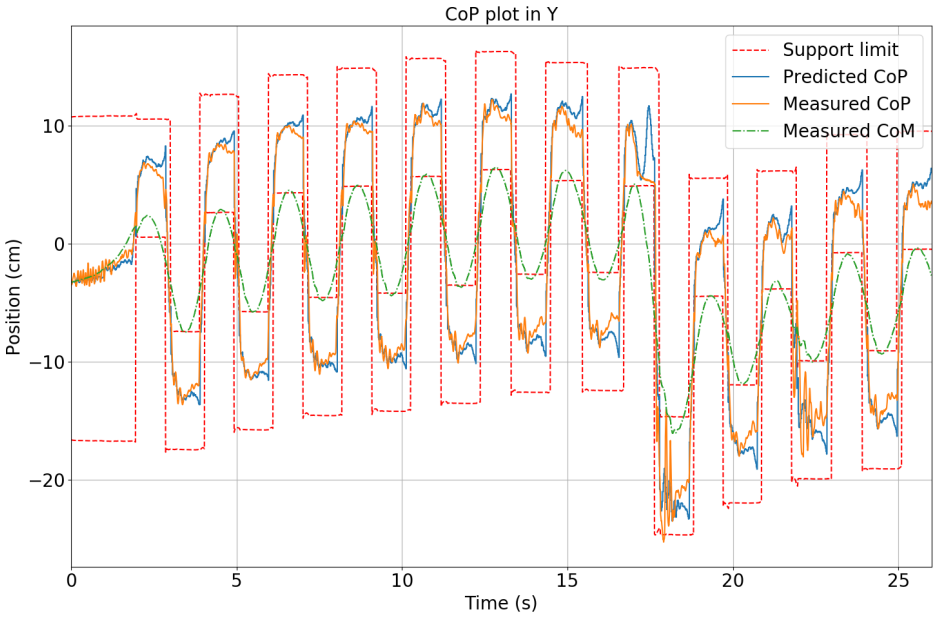

FIGURE 6.5: CoP and CoM trajectories along Y axis for the push recovery experiment on flat floor. The foot of the robot is 10 cm wide.

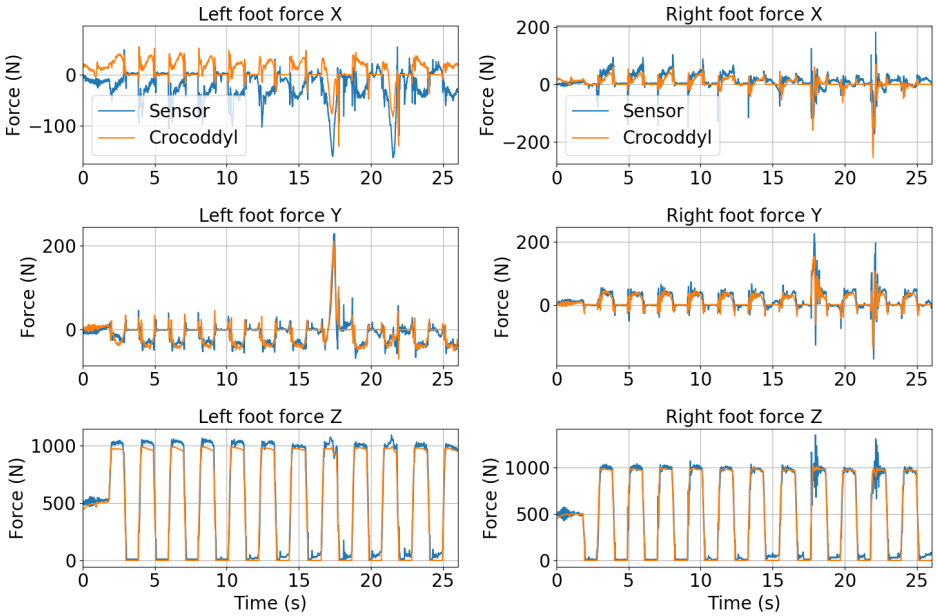

FIGURE 6.6: Predicted vs measured forces in left and right feet during push recovery experiment.

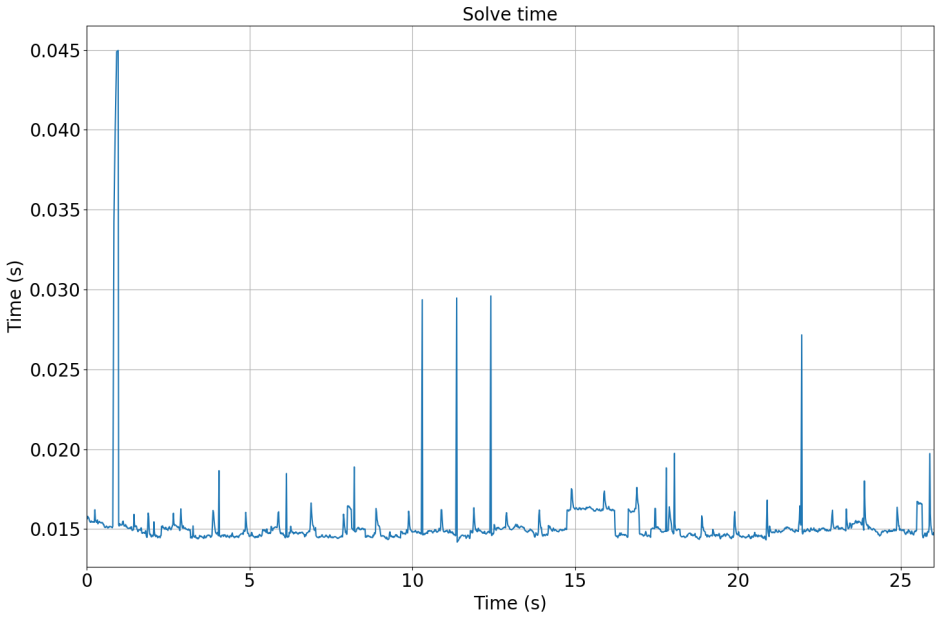

FIGURE 6.7: Time computation of one DDP iteration during push recovery experiment.

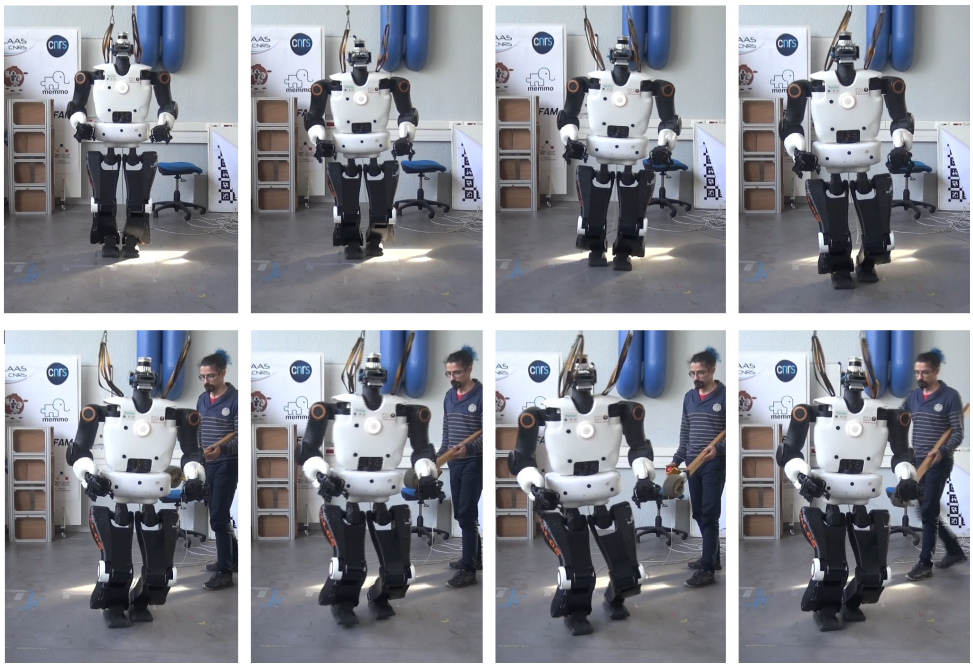

FIGURE 6.8: Snapshot of the push recovery experiment. The robot walks without disturbance in the first 4 images, then gets pushed by a stick and recovers from the fall by stepping aside. The push happens approximately in the lower left frame, at $t \approx 17.5$ s. The video of the experiment is available at https://peertube.laas.fr/w/gtbFKTS8TA6wtwegeSrrBt

6.3.2 Stairs experiment (A2)

The experimental validation of (A2) was performed in simulation[1] with a stairs model of six 10 cm-high stairsteps, except for the first one whose height is 6 cm. The robot is expected to perform 30 cm-length steps in order to put its foot in the center of the next stairstep. The final base position given to the contact planner was fixed to be at the top of the stairs. The gait timings were fixed at 1.5 s for single support phase and 1 s for double support phase.

During each double support phase, the sigmoid parameters were updated based on the next contact position to fit the form of the stairstep to climb. If $\boldsymbol{r}^* = (r_x^*, r_y^*, r_z^*) \in \mathbb{R}^3$ is the desired contact position provided by the contact planner, then the height of the sigmoid is set to $h = r_z^* + 0.03$ and its half-height coordinate is set to $x_h = r_x^* - 0.16$. The width of the sigmoid remains constant at $w = 0.01$. Thanks to this heuristics, the robot can climb stairsteps of different heights using the same locomotion parameters.

The results of the stairs experiment are presented in figures 6.9 to 6.12. A snapshot of the experiment is displayed in Fig. 6.13. Figures 6.9 and 6.10 show that the CoP trajectory remains inside its support limits, save for some isolated spikes of control caused by contact switches. The motion is very conservative since CoM and CoP plots are roughly the same. A drift in Y-axis can be observed on Fig. 6.10: this drift is caused by the contact planner as can be seen on Fig. 6.12, where the desired contact positions tend to go left.

Figure 6.11 displays the simulated contact forces exerted on left and right feet. Contrary to Fig. 6.6, the measured lateral forces along X-axis match the predicted forces of both feet. This advocates for a bad sensor calibration on the real robot during the locomotion experiments of Chap.5. Numerous force spikes are visible at contact landing but do not hinder the execution of the motion. Similarly to the stairs climbing experiment with reference trajectories, the flying foot makes contact with the stairstep a little bit before the solver actually decides to switch to double support. Interestingly enough, the DDP solver chooses to briefly overshoot on vertical forces at transition between double and single supports. This phenomenon is caused by a very brutal foot takeoff with high upward acceleration. In this case, the solver tries to minimize as soon as possible the distance between current and desired foot locations.

Feet trajectories are shown in Fig. 6.12 along with desired feet locations on stairs. The foot placement cost brings the flying end effector close to the target location given by the contact planner, but fails to reach it precisely. The weight on placement cost was chosen as a compromise between the distance to the target at the end of flying phases and the motion abruptness at takeoff. As a consequence, the foot is slighlty above the stairstep when the solver switches to double support, but the compliance of the control scheme allows to handle this discrepancy.

The flying foot trajectory between initial and next contact may be considered erratic, with unnecessary motion along Y-axis and brutal acceleration at takeoff. This behavior is very difficult to prevent with our current problem formulation because we would need to regularize the end effector acceleration, which amouts to solving the dynamics at the 4th order.

[1] We did not have the time to transfer the experiment on the real robot, but hope to be able to do it before the thesis defense.

In brief, the motion is performed in simulation with a gait timing twice as fast as compared to Chap. 5. The general behavior of the solver is satisfactory, with very little unnecessary motions like torso tilting or excessive base swing. The decrease in foot controlled impedance boosted the flexibility of the robot at the price of its tracking precision. On the downside, upward feet velocity is high during take-off and low during landing, when the desired contact placement is near and the effect of the tracking cost weakens.

Due to unplanned logistic delays, we did not have the experimental time to perform stairs climbing with Pyrène using this walking formulation. This remains nonetheless a work in progress, and we expect to get new results on this matter in the following weeks after the completion of the thesis manuscript.

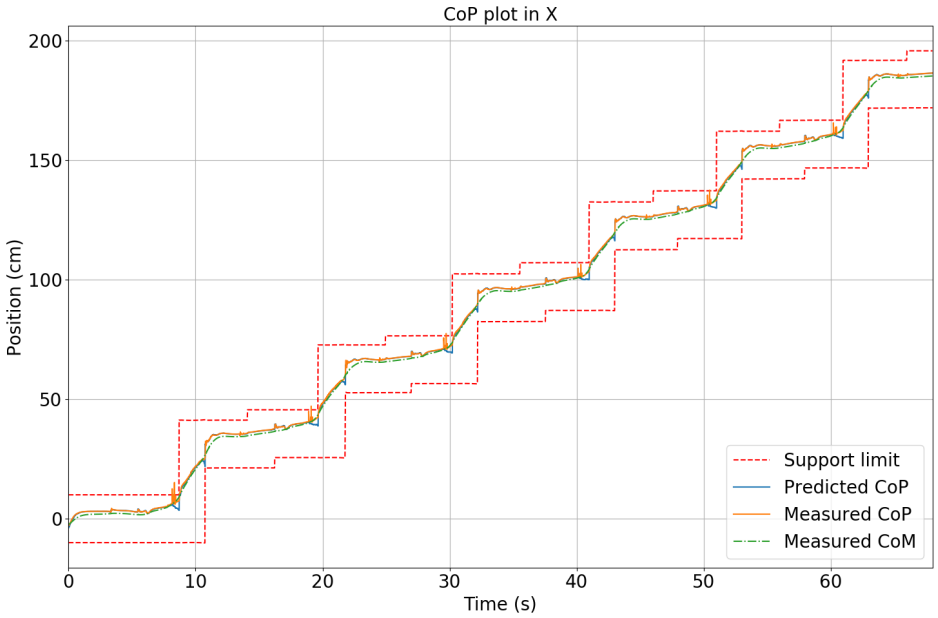

FIGURE 6.9: CoP and CoM trajectories along X axis for the stairs climbing experiment. The foot of the robot is 20 cm long.

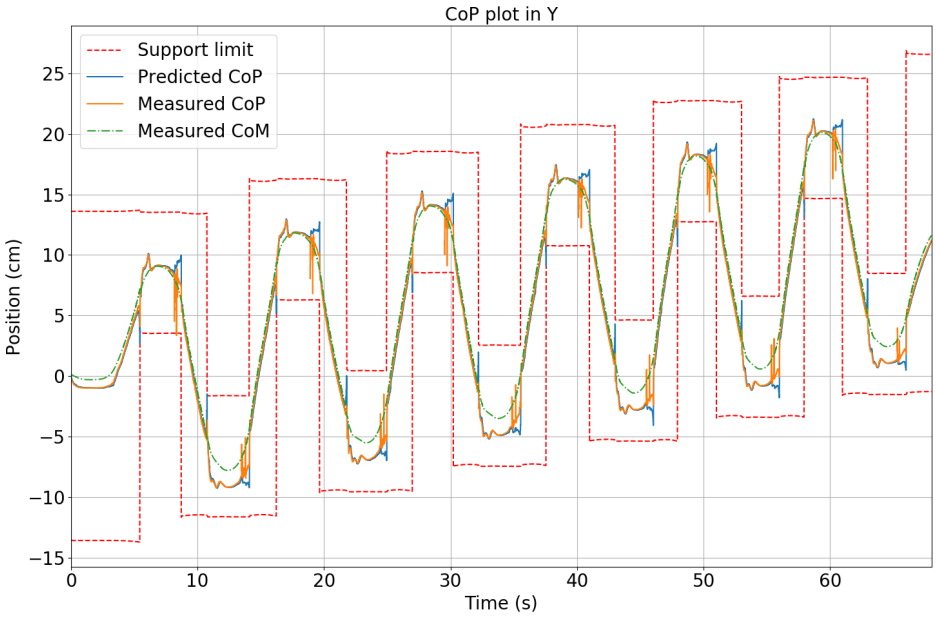

FIGURE 6.10: CoP and CoM trajectories along Y axis for the stairs climbing experiment on flat floor. The foot of the robot is 10 cm wide.

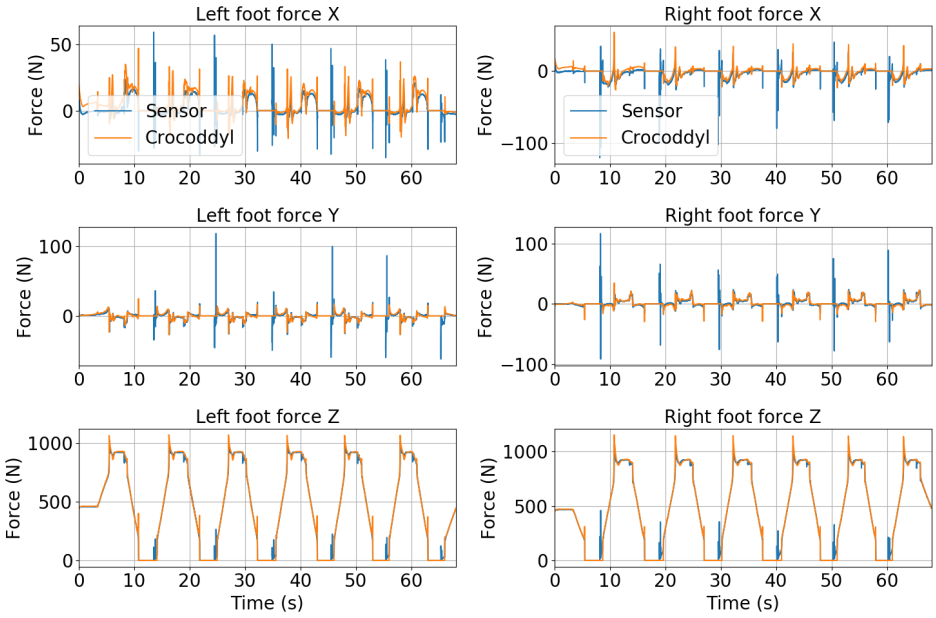

FIGURE 6.11: Predicted vs measured forces in left and right feet during stairs climbing experiment.

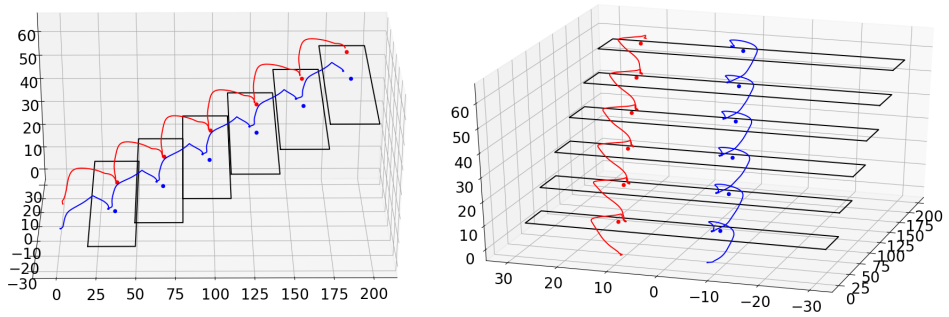

FIGURE 6.12: Feet trajectories during stairs climbing experiment. Stairsteps are displayed in black, left foot trajectory in red, right foot trajectory in blue. Axis are in cm. Desired feet positions are displayed by spherical points.

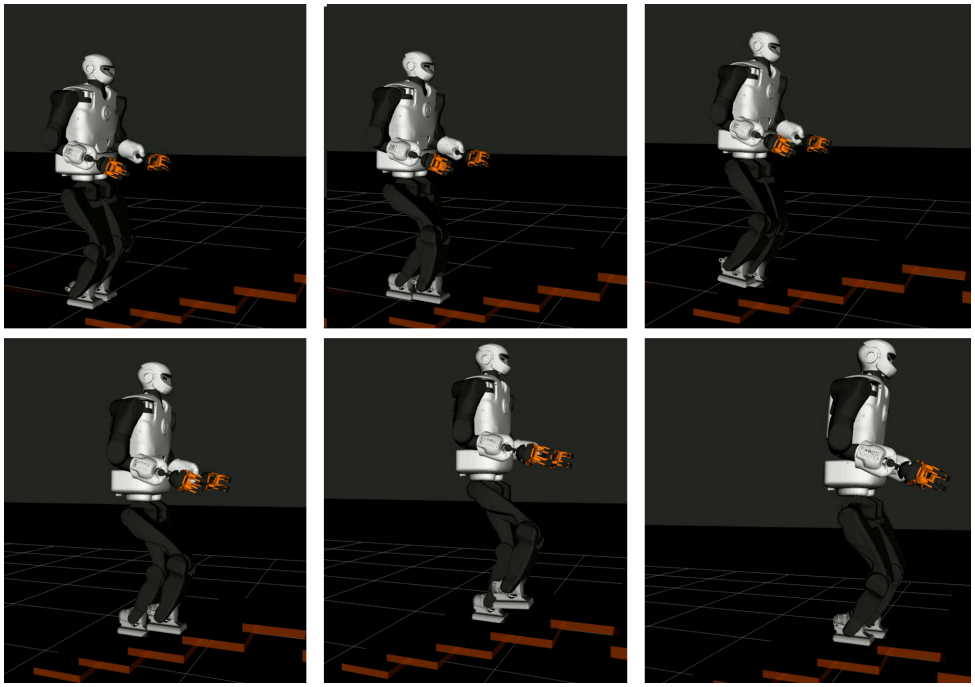

FIGURE 6.13: Snapshot of the stairs climbing experiment in simulation. A video of the experiment is available at https://peertube.laas.fr/w/89oA2BnRte2cJvVxVA4U9Q.

6.4 Conclusion

This chapter has proposed and validated a whole-body locomotion control scheme working with minimal high-level references on flat and uneven terrain. The increased flexibility of the control policy allowed to perform dynamic walking motions and push recovery on the humanoid robot Pyrène. Combined with a closed-loop contact planner and a velocity height map, the same controller could climb stairs in simulation and avoid obstacles in a reactive manner. Further experimental work on the real robot is ongoing.

Nevertheless, gains in flexibility come with precision loss at the task level: to cite an example, the CoM goal was not exactly reached in experiment (A1). In order to be able to compute on the fly a satisfying trajectory, the solver should still be provided with relevant optimization guidelines which depend on the problem structure. In the case of experiment (A2), these guidelines are represented by a finely tuned velocity height map which gives hints about how the solver needs to move its end effectors. From a certain viewpoint, the reference trajectories used in Chap. 5 have been translated into a potential field mapping the terrain configuration. Still, potential fields contain way less information about the desired motion and do not constrained tighlty the end effector velocity and acceleration, contrary to reference trajectories. The problem is, as a consequence, much harder to solve. Whether reference-based or potential-based methods are more relevant to dynamic locomotion remains to be decided. Reference-based approach allows for precise completion of high-level tasks at the price of a higher impedance, while potential-based approach favors flexibility over kinematic precision. In case of a reduced impedance scheme, fine gain scaling may mitigate the issue of precision by increasing placement weights near the target; however, such an heuristics quickly leads to overfitting issues and is equivalent to defining a reference trajectory for gains rather than for end effector.

A drawback of trajectory-free methods is that constraining acceleration is difficult without the use of a reference. The solver can allow very abrupt motion while we would have prefered smooth transitions from initial to desired positions. In order to fix this behavior, it may be necessary to consider minimizing the jerk of the dynamics, or equivalently the change in force and control. Doing so may allow to get rid of the wrench reference cost and to replace it with a penalization on force derivatives. Unfortunately, including the jerk inside our optimal control scheme would make the computation load explode, with little hope of reaching real-time capacities on complex systems. So far, jerk approaches have been considered only in the frame of instantaneous control [Gazar et al. 2021], or for reduced whole-body models limited to simple manipulation tasks [Kleff et al. 2022].

In the frame of this chapter, contact optimization is outsourced to an external planning block, but this fact contradicts the orginal paradigm of solving the locomotion problem through one fullscale optimization process. In its current form, our DDP formulation cannot handle contact adaptation and is bound to follow fixed modes of contact, under the supervision of its user. A major solver improvement is needed to gain further autonomy and adaptability.

Chapter 7

A Memory of Motion for non-convex scenarios

This chapter aims at combining the previously introduced model-based optimization tools with data-driven strategies applied to problem initialization. As gradient-based algorithms struggle to escape local minima in real time, practical non-convex scenarios, which in our experience are problematic as soon as realistic constraints such as obstacles are considered, additionally require whole-body warm-starts in order to converge toward the global solution. In this chapter, a library of trajectories is built offline and accessed online by a MPC solver that refines in real time the proposed solution before executing it on the robot. The task to complete consists in reaching a moving target with the hand while avoiding a fixed obstacle. Although simple, this task cannot be performed without the help of a memory of motion, for the presence of the obstacle breaks the convexity of the reaching problem.

7.1 Motivation

An OCP for locomotion or manipulation typically features nonlinear costs and constraints that need to be tackled by dedicated solvers. Solving this kind of problem at the frequency of the system actuation is extremely challenging, despite the ever-increasing computational power of current machines. Applications of MPC in robotics have yet been limited by computational limits, as complexity scales at least with the cube of the system dimension. Moreover, nonlinearities in the robot model make the OCP non-convex, resulting in two main consequences: the computational load increases again, and local minima start to appear in the problem.

Efficient numerical tools for optimal control still require several costly iterations to converge, in particular when it comes to whole-body control. Direct transcription methods [Hargraves and Paris 1987; Pardo et al. 2016], in which both state and control are treated as variables, are common to solve such problem. Others transcriptions of the OCP are now considered to be more suitable to MPC, in particular shooting methods like DDP or iLQR. Efficient solvers based on DDP have already been implemented [Grandia et al. 2019; Howell et al. 2019; Mastalli et al. 2022a] and are able to deal with nonlinear costs, constraints and dynamics. Nevertheless, it remains a challenge to reduce the number of iterations needed to converge. Currently, state-of-the-art whole-body planning algorithms remain too slow to provide an optimal solution in a reactive way, given new information updates. One approach to

enhance reactivity is to formulate the DDP scheme as a more convex problem [Majumdar et al. 2020], easier to solve. Another one consists in providing the solver with a good initial guess [Mansard et al. 2018], so as to start the optimization process as close as possible to the desired solution. With the second method, the question arises as to how this initial guess should be computed, and how much information it should contain.

Outsourcing the complexity of the problem to an offline data library has first been proposed in [Stolle and Atkeson 2006], where the stored control solutions are selected online through a nearest-neighbor approach. Many different quantities can be pre-computed to alleviate the computational load of optimization algorithms, e.g. value function [Zhong et al. 2013; Lowrey et al. 2018], system dynamics [Lenz et al. 2015], contact feasibility [Orthey and Stasse 2013], or cost function [Tamar et al. 2017]. In particular, it has been shown in [Mansard et al. 2018] that learning the state and control trajectories to warm-start an MPC produces better results than learning the optimal control policy. In a similar vein, the work of [Lembono et al. 2020a] compares several regression algorithms for learning a good warm-start from a pre-computed database in the context of locomotion. Such a learned initialization is often called a Memory of Motion and is articulated between two phases: first learn offline, then refine online.

In the frame of the present chapter, we will be using a Memory of Motion strategy to overcome the non-convexity of a typical obstacle avoidance task. As our feasibility-prone DDP algorithm is particularly suited to handle unfeasible warm-starts, combining it with a learned initialization is straightforward and efficient.

7.2 Building a Memory of Motion

Our goal is to create a library of trajectory solutions for a whole-body OCP involving a reaching task, similar to Chap. 4. Let us re-examine the control problem defined by (3.9). As the transcription of this equation is non-convex for complex robots, the behavior of the OCP solver cannot be guaranteed: it may get stuck in a poor local minima if the initial warm-start happens to be in the wrong convergence basin. The MPC needs to be augmented with an external process to infer proper candidate initialization. The problem of inferring the warm-start is formulated as a regression problem $g(\boldsymbol{\theta}) = \underline{z}$ where the input task $\boldsymbol{\theta}$ contains any relevant state or sensor measures, and the output $\underline{z}$ represents the corresponding state and control trajectories. Such a memory of motion relies on a dataset of optimal trajectories built off-line and encoded by machine learning.

7.2.1 Dataset generation

Our dataset is generated by a sampling-based planner that builds an extensive library of state trajectories dedicated to the task to perform, ensuring that a solution is obtained if it exists. The approach leverages an efficient path planning algorithm for configuration spaces with multiple constraints, namely the Constrained Bi-directional RRT (CBIRRT) [Berenson et al. 2009]. This algorithm extends the Bi-directional RRT (BiRRT) approach and allows to explore the constraint space manifolds of the problem via a clever projection strategy. It has proven to be particularly efficient to find bridges between two constraint manifolds.

From the state trajectory given by our planning algorithm, we infer a control trajectory by simply computing the quasi-static torques corresponding to the state

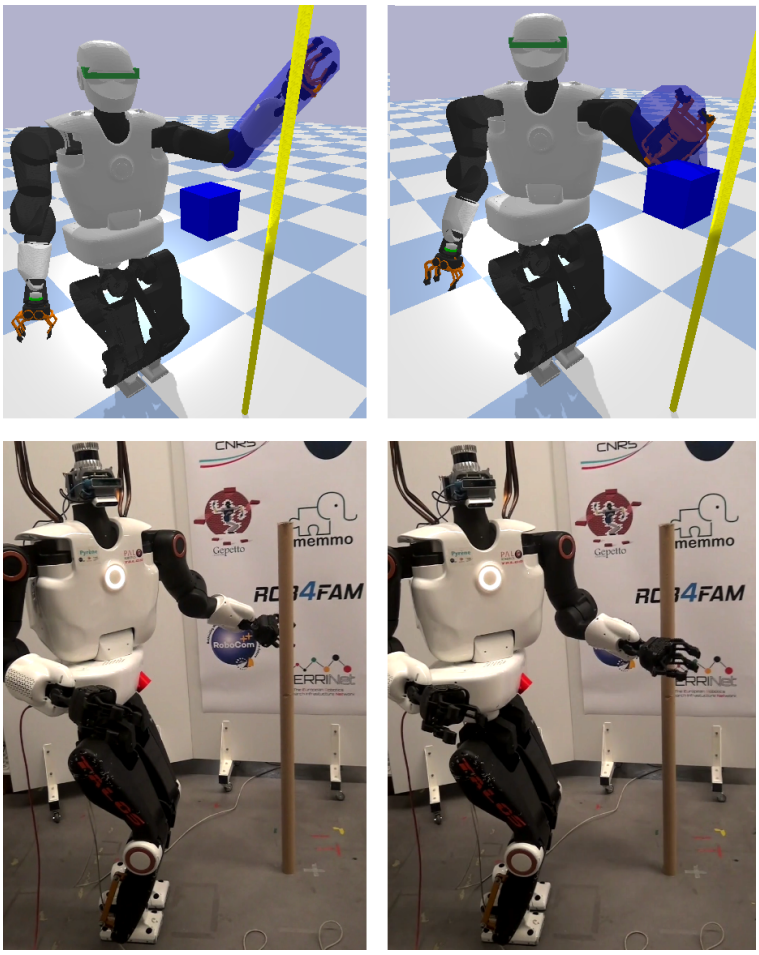

FIGURE 7.1: Pyrène performing collision avoidance in real time, in simulation and with the real platform. In simulation, the blue shape encapsulating the left arm is an inflated capsule used to compute collision distance with the yellow pole. The video of the experiment is available at https://peertube.laas.fr/w/rakM3PzkmXhrVA1wgTWFM8?start=37s.

sequence. In other words, we perform a pass of Recursive Newton-Euler Algorithm with null acceleration to generate our control warm-start.

7.2.2 Decoding the memory

The dataset that we build offline is stored in the format $\{\boldsymbol{\theta}_i, \underline{\boldsymbol{z}}_i\}, \forall i = 0 \cdots N_m$, where $N_m = 1000$. Given a new task $\hat{\boldsymbol{\theta}}$, we rely on a Nearest Neighbor (NN) strategy to compute the closest $\boldsymbol{\theta}_i$ to $\hat{\boldsymbol{\theta}}$ in the dataset and output the corresponding $\underline{\boldsymbol{z}}$ as the warm-start. The NN algorithm implements an Euclidean metric on an efficient data structure based on K-D tree [Bentley 1975]. Gaussian Process Regression (GPR) was shown in [Lembono et al. 2020a] to perform much better than NN in producing DDP warm-starts for unimodal tasks. Yet the task considered in this chapter is multimodal, i.e., there exist several qualitatively different trajectories to achieve one given task. For such high redundancy problem, GPR performs poorly as observed in [Lembono et al. 2020b], hence NN was chosen instead to perform the reaching task. More advanced memory encoding should be considered in future works, but

this is not limiting for the experimental scope tackled in this chapter.

The candidate trajectories $\underline{z}$ given by memory inference is not directly comparable to the current solution of our MPC scheme. In order to measure the interest of the memory, we need to refine this candidate by performing a few DDP iterations on it using the same OCP formulation as in the MPC. The goal of this refining process is also to smooth the candidate trajectory and to make it feasible before sending it to the MPC.

7.3 Obstacle avoidance MPC formulation

This section describes the complete formulation of a MPC scheme for target tracking in cluttered environments. The main contribution of this section lays in the addition of a memory of motion to the general control framework. At run time, a good initialization is frequently inferred from the memory, then refined (as exemplified by Fig. 7.3) and provided to the MPC as a potential best solution to the current problem configuration. Finally, a Riccati feedback block is integrated inside the low-level control of the framework in order to speed up the MPC frequency and be able to use the whole-body model of the robot.

The OCP structure presented in the following corresponds to the one used for evaluating the interest of a memory of motion in Sec. 7.4.1. It will be also implemented inside our MPC framework to experimentally validate the importance of warm-starting the solver online.

7.3.1 OCP formulation

Our OCP is built on top of the one presented in Sec. 4.3.1. To the five cost functions already introduced in previous sections (state and control regularizations, CoM position, end effector tracking and kinematic limits), we add a sixth cost for collision avoidance with external obstacles.

Collision avoidance cost

Given a pair of collision between bodies A and B, we denote $d_{AB}(\boldsymbol{x})$ to be the minimal distance between these two objects in configuration $\boldsymbol{x}$. Defining d_{min} as a distance threshold for the cost activation, the collision penalization function writes:

$$l_{col}(\boldsymbol{x}) = \begin{cases} (d_{AB}(\boldsymbol{x}) - d_{\text{min}})^2 & \text{if} \quad d_{AB}(\boldsymbol{x}) < d_{\text{min}} \\ 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

The distance $d_{AB}(t)$ is computed from the two body placements using standard proximity algorithm [Pan et al. 2012]. In order to simplify the problem, we only consider collisions between the end effector (left arm of the robot) and a fixed obstacle in the environment, while modeling both arm and obstacle with inflated capsules (see Fig. 7.1). Capsule collisions are indeed easier to deal with because the minimal distance between them are the minimal distance between their core segments, plus their respective radius. We expect that this approach should be able to nicely generalize at least from a theoretical viewpoint [Montaut et al. 2022], but would lead to challenging computational perspectives.

7.3.2 Model and timings

Two different models were used to perform obstacle avoidance in the frame of this thesis. The first one includes only 6 joints, 4 for the left arm and 2 for the torso. This very simple model leads to a computation-light OCP which can be solved in a few milliseconds. The resulting MPC scheme was deployed on the real robot without the integration of the Riccati feedback loop. The second model is identical to the one used in Chap. 4, with 22 actuated joints (12 for the legs, 8 for the arms, 2 for the torso), plus the free flyer joint state. This second MPC needs to be paired with the Riccati feedback scheme to produce stable motions, since its computational load is much higher.

Regardless of the model, both OCP are composed of 100 knots separated by a 10 ms time step. With this specific time parametrization, the dynamics of the robot is previewed over 1 s in the future, with a mean time computation of 3 ms for the reduced model and 13-15 ms for the complete model. Again, only one DDP iteration is performed to compute the next control [Diehl et al. 2005], so a warm-start close to the optimal solution becomes essential in case of non-convexity.

7.3.3 Cost parametrization

The weight distribution introduced in Table 7.1 was selected to perform obstacle avoidance in real time with the fullscale model. Most weights are identical to the ones used in Chap. 4, because the task to complete is very similar in both cases. Collision avoidance weight is set to be high in order to implement the cost as a hard constraint.

	ℓ^x	ℓ^u	ℓ^r	ℓ^b	ℓ^{com}	ℓ^{col}
Running costs	0.02	0.01	5	1000	500	10000
Terminal costs	0.02		50		500	10000

TABLE 7.1: Cost weights of our collision avoidance OCP.

Compared to the reaching OCP formulated in Sec. 4.3.1, the control weight is increased to penalize abrupt motions caused by updates in warm-start when the target is far from the end effector. The control matrix $\boldsymbol{B}_u$ is again the identity matrix, and the state matrix $\boldsymbol{B}_x$ is presented in Table 7.2. The velocity weights in $\boldsymbol{B}_x$ are increased to smooth the reaching motion. Torso displacements are also heavily penalized to favor arm motions.

	Base pose	Base angle	Leg	Torso	Left Arm	Right Arm
Position	100	500	500	10000	20	500
Velocity	100	100	100	100	1000	100

TABLE 7.2: Diagonal terms of the weight matrix $\boldsymbol{B}_x$ for full collision experiment.

7.3.4 Control scheme overview

The structure of the MPC algorithm including a memory of motion is presented in Alg. 3. Every 15 ms or so, the OCP is updated by setting the initial state to the

latest estimated state $\hat{\boldsymbol{x}}$ (line 23), and by updating the position of the target to reach (line 12). In our previous MPC formulations, we used the current solution shifted by one knot as a warm-start for the DDP solver (line 20 and 21). This initialization is acceptable for simple problems like reaching a target in a collision-free environment or walking on flat ground. However, when the structure of the problem changes radically, for example when an end effector target goes from one side of an obstacle to another, the optimal solution can be impossible to retrieve with a gradient-based algorithm starting in the wrong convergence basin. Figure 7.3 indeed illustrates that a bad warm-start may get the solver stuck in poor local minima, despite the fact that a better trajectory exists.

Algorithm 3 MPC algorithm for goal tracking with obstacle avoidances

Require: T, T_{tot}
1: - Measure initial state $\hat{\boldsymbol{x}}$
2: - Initialize reference state: $\boldsymbol{x}_d \leftarrow \hat{\boldsymbol{x}}$
3: - Initialize reference control: $\boldsymbol{u}_d \leftarrow RNEA(\hat{\boldsymbol{x}})$
4: - Initialize target CoM: $\boldsymbol{c}^* \leftarrow CoM(\hat{\boldsymbol{x}})$
5: - Build preview horizon with T action models
6: - Set target r^* in OCP
7: - $\underline{\boldsymbol{x}} \leftarrow (\boldsymbol{x}_d)_{i=0}^{T}$
8: - $\underline{\boldsymbol{u}} \leftarrow (\boldsymbol{u}_d)_{i=0}^{T-1}$
9: - $\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*, K[0] \leftarrow ddp.solve(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ until convergence
10: - Initialize iteration counter: $i_c \leftarrow 0$
11: **while** *True* **do**
12: - Update target r^* in OCP
13: - Measure new state $\hat{\boldsymbol{x}}$
14: - Publish $r^*, \hat{\boldsymbol{x}}$ to memory node
15: - Get new trajectory $\underline{\boldsymbol{x}}^e, \underline{\boldsymbol{u}}^e$ and corresponding cost $cost(\underline{\boldsymbol{x}}^e, \underline{\boldsymbol{u}}^e)$
16: **if** $cost(\underline{\boldsymbol{x}}^e, \underline{\boldsymbol{u}}^e) < cost(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ **then**
17: - $\underline{\boldsymbol{x}} \leftarrow \underline{\boldsymbol{x}}^e$
18: - $\underline{\boldsymbol{u}} \leftarrow \underline{\boldsymbol{u}}^e$
19: **else**
20: - $\underline{\boldsymbol{x}}[0:T-1] \leftarrow \underline{\boldsymbol{x}}[1:T]$
21: - $\underline{\boldsymbol{u}}[0:T-2] \leftarrow \underline{\boldsymbol{u}}[1:T-1]$
22: **end if**
23: - $\underline{\boldsymbol{x}}[0] = \hat{\boldsymbol{x}}$
24: - $\underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*, \boldsymbol{K}_0 \leftarrow ddp.solve(\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}})$ with only one iteration
25: - Publish optimal policy $\boldsymbol{x}_0^*, \boldsymbol{u}_0^*, \boldsymbol{K}_0$
26: - $\underline{\boldsymbol{x}}, \underline{\boldsymbol{u}} \leftarrow \underline{\boldsymbol{x}}^*, \underline{\boldsymbol{u}}^*$
27: - $i_c \leftarrow i_c + 1$
28: **if** $i_c >= T_{tot}$ **and** $i_c <= T_{tot} + T$ **then**
29: - Deactivate hand translation cost in action model number $T - (i_c - T_{tot})$
30: **end if**
31: **end while**

The memory of motion described in Sec. 7.2 is implemented to provide the MPC with alternative trajectories when the previous warm-start strategy fails to perform the task. Given a target to reach, obstacle positions and state estimate, the memory infers the corresponding state and control trajectory and further refines this candidate by several DDP iterations, using the same DDP solver as for the MPC. In order

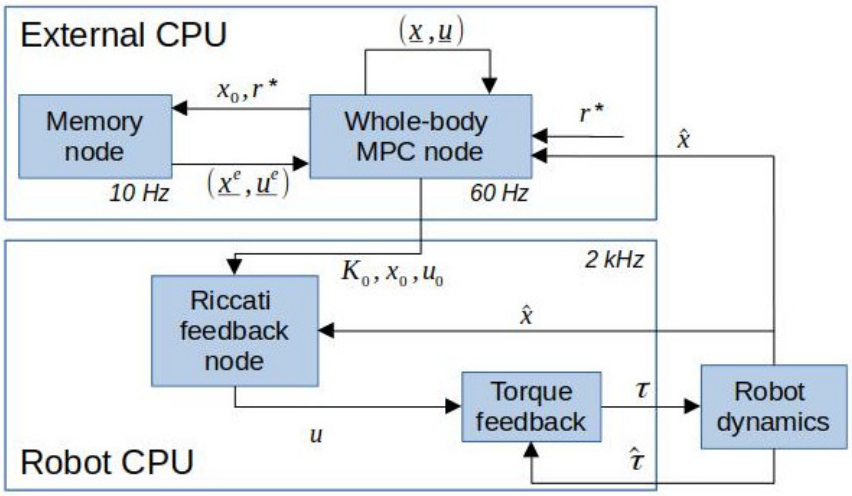

FIGURE 7.2: Diagram of the ROS architecture of the collision avoidance scheme. $(\boldsymbol{x}_0, \boldsymbol{r}^*)$ are the initial state and current target position sent to the memory node. $(\boldsymbol{x}, \boldsymbol{u})$ are the current optimal state and control trajectories produced by the MPC. $(\boldsymbol{x}^e, \boldsymbol{u}^e)$ are the warm-start trajectories computed by the memory of motion.

to output a good refined warm-start in a reasonable amount of time, we chose to limit the refinement computation to 2 iterations. This operation is performed inside the memory of motion node, and as a consequence does not appear on Alg. 3. The average frequency of the memory node is about 10 Hz, which is slow as compared to the MPC. Further work may include an optimization of the NN algorithm used to infer a solution from our library of trajectories.

Once received by the MPC, the cost of the warm-start is compared against the cost of the current MPC trajectory (line 16). The memory warm-start is accepted by the MPC if its cost is lower than the current solution (line 17 and 18). At the end of the programmed motion, the tracking cost is sequentially removed from the OCP, starting from the end of the horizon (line 29).

The implementation needs three parallel processes to run simultaneously: the MPC, the memory and the low-level torque control. The ROS architecture of our control framework is presented in Fig. 7.2.

7.4 Experimental results

As a first step, the effect of the memory of motion was evaluated on simulation with a reduced robot model composed of 6 joints (left and arm torso), with no free-flyer. Later, the same robot model was used to perform collision avoidance on Pyrène without the integration of the Riccati feedback loop. Results were published in [Dantec et al. 2021]. This work provided a preliminary proof of concept of our memory scheme, but was limited to a simple model with fixed base, similar to a robotic arm. A whole-body version of the same algorithm including 22 actuated joints, a free-flyer base and a Riccati feedback policy has later been developed to confirm that obstacle avoidance could be done with more complex dynamics. We present here this improved MPC version running on the real robot and highlight some key experimental results.

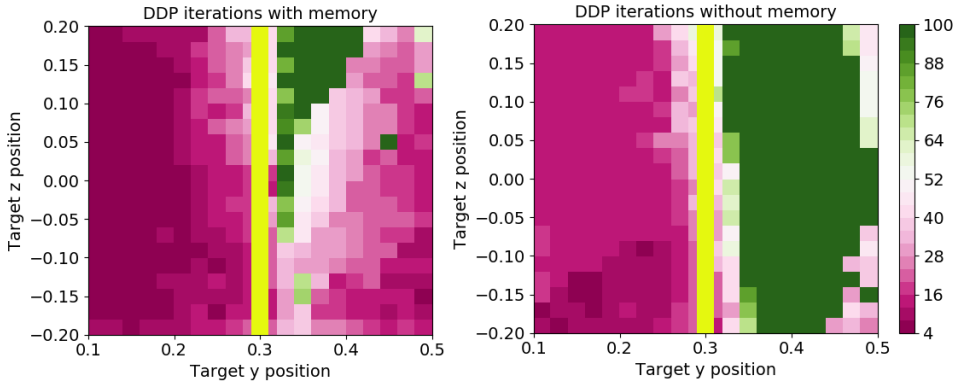

FIGURE 7.3: Comparison of convergence speed of the OCP solver (left) with and (right) without warm-start, for a grid sampling of the target position, while the initial configuration is always the same. The metric is the number of iterations to convergence, with an upper limit of 100 iterations. The arm initial position is on the bottom left corner of each picture. The obstacle (a pole) is highlighted in yellow.

7.4.1 Memory evaluation

In order to understand how the initialization improves the behavior of the DDP solver when facing a non-convex scenario, let us consider the end-effector reaching problem described in Sec. 7.3.1, and illustrated by Fig. 7.1. For the sake of the demonstration, the model used here has only 6 joints, but the problem stays essentially the same when going fullscale.

In simulation, the OCP is solved several hundred times with slightly different target positions: while the X coordinate of the targets remains at coordinate 0.07 m, the Y and Z coordinates are bound to go respectively from 0.1 m to 0.5 m and from -0.2 m to 0.2 m (in the robot frame). The obstacle is a 2 cm diameter vertical pole set at coordinate ($x = 0.6$ m, $y = 0.3$ m). This set of problems is solved in two ways: first with a memory warm-start corresponding to the current target position, secondly with a static initialization identical for all target positions. The number of iterations until convergence is chosen as a metric to compare the behavior of the solver with and without memory. If this number exceeds 100, we consider that the optimal solution was not found by the solver. The results of this experiment is presented in Fig. 7.3.

As the arm starts from the bottom left corner, the left target configurations are easier to reach. Without memory, the solver cannot reach the target in less than 100 iterations when it is located at the right side of the obstacle. Practically speaking, the arm becomes stuck in front of the pole because the DDP algorithm finds the best local compromise between the repulsive obstacle cost and attractive hand target cost. Even if the FDDP algorithm benefits from some globalization capacities thanks to its multiple shooting formulation, it is not enough to deal with strongly non-convex scenarios.

With a memory initialization, one can observe that the number of iterations until convergence generally decreases, no matter whether the target is at the left or right side of the obstacle. Configurations previously unreachable can now be achieved in a reasonable amount of iterations. However, a fraction of them remain impossible to reach by the solver, even with memory initialization. Those configurations are

especially challenging because the target is located at the opposite of the initial pose of the arm with respect to the obstacle.

In summary, the use of a memory warm-start allows our solver to speed up the convergence process and get out of tricky local minima arising from non-convexity. Even if some problem configurations could still not be solved with such an approach, the quality of the memory of motion was deemed good enough to be coupled with a whole-body MPC.

7.4.2 Whole-body obstacle avoidance

The objectives of the obstacle avoidance experiment is to prove that a memory of motion can be used to warm-start online the whole-body MPC in order to escape from local minima arising in non-convex manipulation scenarios. The experimental protocol consists in reaching a moving target with the left hand while it goes behind a pole-like obstacle of 2 cm radius and 2 m long, placed in coordinates ($x = 0.6$ m, $y = 0.3$ m) (see Fig. 7.1). To take into account security margins, the collision threshold in cost ℓ^{col} is fixed to $d_{min} = 0.15$ m. This means that the solver will start to consider the obstacle when the end effector is less than 15 cm away from the center of the pole. The target to track $\boldsymbol{r}^*(t) = (r_x^*, r_y^*(t), r_z^*)$ is moving along the Y-axis according to a sinusoid motion going from $r_y^* = -0.1$ m to 0.7 m, with $r_x^* = 0.65$ m and $r_z^* = 1$ m fixed X and Z coordinates. The particular geometry of this reaching task results in a challenging non-convex problem to solve for the DDP. The resulting movement is depicted in Fig. 7.1.

The resulting plots of the full collision avoidance experiment are presented in figures 7.4 to 7.6. As can be seen on Fig. 7.4, the commanded torques sent to the low-level control feature some brutal discontinuities at a regular frequency. Such discontinuities are also observed at the same instants on Fig. 7.5. Each time the target to track goes behind the obstacle, the MPC becomes stranded in a local minimum until a circumventing solution is provided by the memory node. However, this solution tends to be very different to the one computed by MPC, and it triggers a sharp change in control, proportional to the distance between end effector and target. The solver then tries to bridge the distance as fast as possible, resulting in a very dynamic motion.

The memory solution is taken into account each time its cost becomes lower than the trajectory computed at the last control cycle, as can be observed in Fig. 7.6. When the arm is blocked by the obstacle, the target keeps on moving away from it, resulting in an increase of the total DDP cost. The spikes in the DDP cost plot match the control discontinuities of Fig. 7.4 and dynamic avoidance motions of Fig. 7.5.

Lastly, one can notice that the target is not reached perfectly in Fig. 7.5, although the obstacle is correctly avoided. This is because the weight of the tracking task is too low to incite the solver to come closer. A higher weight would cause sharper control spikes during memory updates and would make the motion too brutal to be executed. Indeed, sharp controls tend to produce high joint accelerations which trigger the low-level security fuses of the robot. Several workarounds have been tested to mitigate this issue, including an increase in arm joint velocity weights or the introduction of a end effector velocity penalization, but none of them produced satisfying results in simulation or on the real platform.

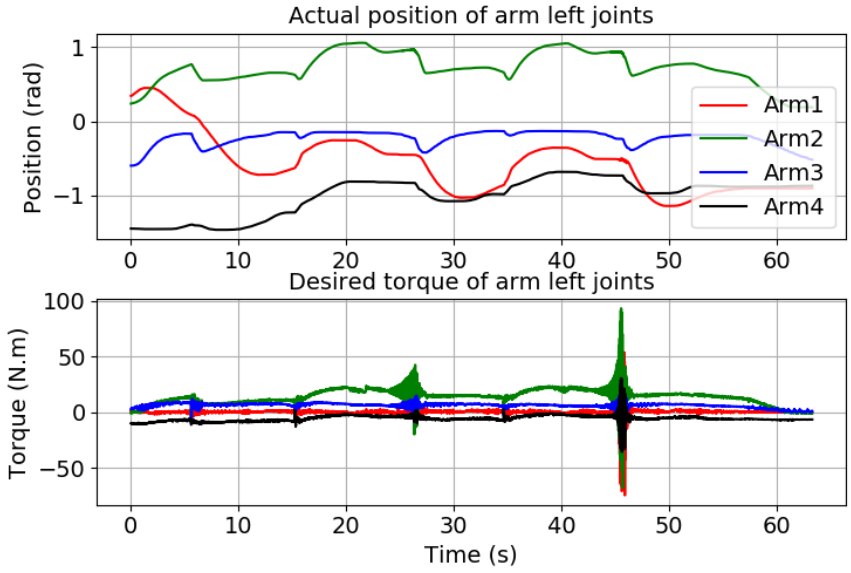

FIGURE 7.4: Measured positions and commanded torques of the left arm joints during collision avoidance experiment.

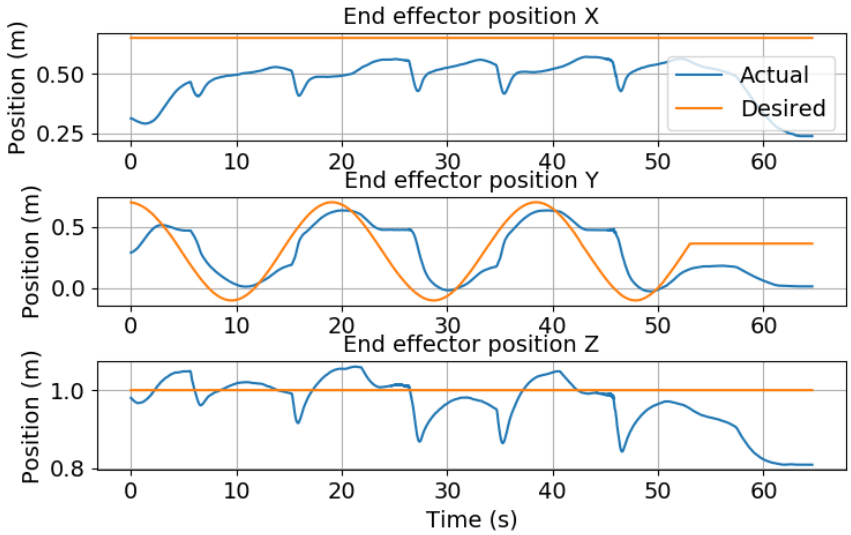

FIGURE 7.5: Tracking of the desired end effector position during collision avoidance experiment.

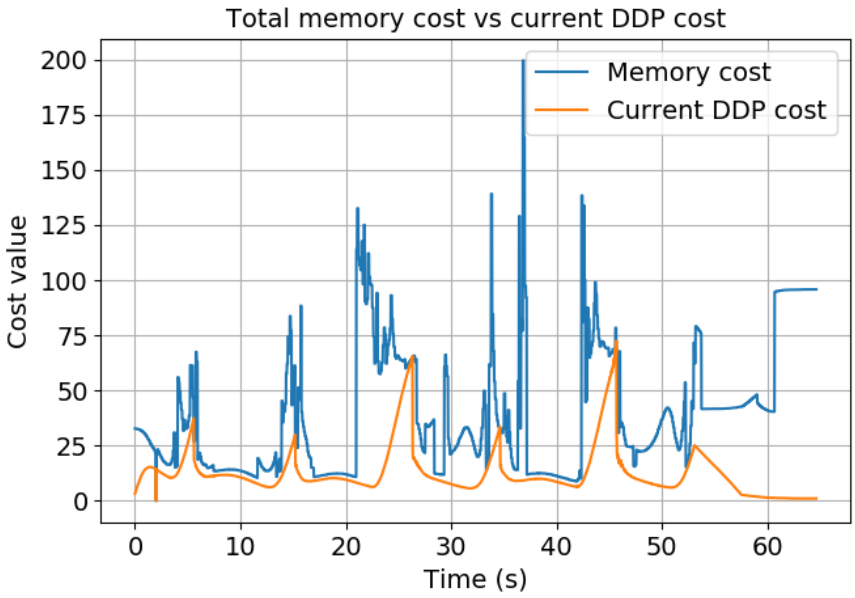

FIGURE 7.6: Comparison between total DDP cost and memory cost during collision avoidance experiment, as performed in line 16 of Alg. 3.

7.5 Conclusion

This chapter introduced a strategy to incorporate an efficient DDP initialization scheme based on a memory of motion inside our whole-body MPC implementation. This scheme has allowed to complete reaching tasks in cluttered environments with the Pyrène platform, using a reduced model and a fullscale model of the robot. Experiments showed that a MPC framework without memory struggles to perform obstacle avoidance in real time and tends to remain stranded in local minima, at the points where collision cost and target tracking cost reach a compromise. In weak non-convex scenarios, the memory of motion still helped to reduce the number of iterations needed to converge toward a global solution.

Our implementation of MPC with memory of motion suffers from significant control discontinuities at memory initialization, caused by the high impedance of our tracking cost. A lower impedance would lead to lower precision performances and a more conservative control scheme. One solution to counter this behavior would be to penalize fast control changes inside the OCP, but such an approach requires to consider the 4th order of the whole-body dynamics, which is too costly to compute online on current hardwares.

For simple reaching task involving only one obstacle, it is likely that a MPC based on simple potential fields could perform collision-free tracking of a target, as was demonstrated in Chap. 6. A memory of motion, however, may be needed for any complex scenario in cluttered environments, where local field methods may get stuck more easily. This memory of motion comes with the price of having to compute offline a whole library of trajectories for each particular task to perform. An interesting perspective would then be to develop a better memory based on advanced RL techniques rather than RRT algorithms. In the light of recent RL achievements, it

may be relevant to write a generic framework combining a MPC scheme for online refinement with a data-driven warm-start policy to boost globalization capacities.

A natural extension of this chapter is to study how the complexity of the formulation scales with the number and structure of obstacles. For the sake of the proof, the obstacle considered here is very simple and the corresponding collision test is cheap to compute inside the OCP. More challenging scenarios involving multiple obstacles would call for efficient collision detection heuristics [Montaut et al. 2022].

Chapter 8

Conclusion and perspectives

This thesis has proposed a complete solution to leverage the whole-body system dynamics inside a predictive control of a real humanoid robot, along with the experimental validation of the concept. The resulting control formulation is able to generate complex whole-body motions in real-time for both manipulation and locomotion, and has proven to be generic, adaptable and robust to model discrepancies.

This achievement relied on several theoretical cornerpieces, in particular an efficient optimal control solver, a generalization of the Riccati gains for high frequency feedback, and the use of a memory of motion for globalization. It also benefited from key technological advances, in particular high frequency torque control [Stasse et al. 2017], efficient rigid body dynamics computations and derivatives [Carpentier et al. 2019] and a fast DDP solver implementation [Mastalli et al. 2020].

Chapter 4 introduced the key feature of our optimal control formulation: a first-order state feedback policy based on the Riccati gains computed inside the DDP backward pass. This simple feedback scheme has allowed to approximate the optimal torque command at the low-level control frequency, unlocking new dynamic motions that could not be done with current MPC frequency. Moreover, we showed that other relevant feedback policies can be deduced by derivating the backward pass with respect to key task parameters, like the desired position of the end effector. Computing generic sensitivities is particularly interesting in case a high-frequency measure of the given parameter is available to the user.

Next, the Riccati feedback policy has been integrated into a locomotion scheme in **Chapter 5**. Thanks to a clever contact forces regularization scheme and hand-crafted feet splines, the humanoid Pyrène performed locomotion on flat floor at different gaits and climbed a 10 cm-high stairstep, despite significant model differences and drifting base estimator. All motions were executed with the same set of gains to illustrate the genericity of the approach.

In order to increase the flexibility of the control scheme, another approach to locomotion was proposed in **Chapter 6**, consisting in replacing the user-defined feet trajectories with a foot velocity cost depending on the distance to a given obstacle, ground or stairs. The goal was to free the solver from the foot placement constraint and to make it able to adapt on the fly to violent disturbances that require sidestepping to be dissipated. Thanks to this scheme, push recovery on flat floor was performed on the robot Pyrène. Using a high-level contact planner to define the sequence of next contacts, the same robot climbed 6 stairsteps in simulation without

relying on hand-crafted references to avoid collisions with its environment. Additional experimental work is on-going to transfer the motion on the real robot.

Finally, the extension to non-convex scenarios was studied in **Chapter 7**. A memory of motion was built offline and tested to warm-start online the optimal control solver in order to reach a moving target while avoiding obstacles. The main goal was to demonstrate that a whole-body MPC scheme can easily be combined with learning strategies to overcome non-convexity an improve performances without burdening computational load.

In the context of this work, a major amount of time was spent into seeking the correct formulation of the full dynamics MPC. As our framework merges together the planning block and control block, every parameter of the problem has to be carefully tuned with respect to each other in order to produce a robust walking motion. As such, one of our main objectives was to identify the minimum set of cost functions that leads to a satisfying solution, so as to reduce the tuning effort on the robot. Still, the number of weights to tune is proportional to the number of controlled joints, which can quickly increase for complex machines.

Timing parameters, as well as costs, were the subject of an intensive exploratory work in simulation. We have tested numerous combinations of gait parameters, horizon length, iteration number, integration timestep and contact switch before finding the right recipe for locomotion. While the novelty of the approach for humanoid robots explains in part the difficulty of parameter tuning, we also foresee that more advanced theoretical solvers, such as one able to account for hard constraints, would allow to alleviate the experimental burden.

Perspectives

The present work can be seen as a proof of concept for whole-body dynamics MPC on real humanoid robots. Whether this control solution performs better than current state-of-the-art approaches to locomotion still remains to be demonstrated. On flat ground, centroidal pattern generators display very advanced performances that may be impossible to exceed with whole-body approaches. When it comes to more complex locomotion scenarios, no generic benchmark is available to compare the performances of our MPC against other control frameworks. In our opinion, the advantage of whole-body predictive control mainly lies in its capacity to tackle a wide range of manipulation and locomotion tasks inside a single control block, while providing a reduced implementation effort. In any case, the comparison of our MPC framework against state-of-the-art walking schemes should be made from this perspective.

Our optimal control solver combined with a low-level Riccati feedback policy allowed us to reach satisfactory control frequency on the robot Pyrène, but at the price of some compromises. The upper body part was not taken into account inside the model, and our DDP algorithm does not consider the Hessian of the dynamics, even if doing so ensures a quadratic convergence rate. Despite these simplifications, the whole-body optimization still needs to run on a powerful external CPU, whereas centroidal approaches can be embedded very easily. At the current state of development, the heavy computational load and lack of robustness margins of our MPC scheme may hinder its deployment on modern robots. It is still unclear if the performances of the predictive scheme could be improved by increasing again its working frequency, or which horizon length needs to be considered. A better

compromise is certainly to be found between preview capacities and computational load. A short term perspective would be to enhance the computation efficiency of the general scheme, and we propose here some avenues for future development on this subject.

We believe that a lot of time could be gained by optimizing the sparsity of the backward pass inside the DDP algorithm, by using code generation techniques or by working on parallelizing the formulation [Laine and Tomlin 2019]. Using a variable timestep inside our horizon could be another lever to reduce the computation load by cutting the number of knots, but this research path requires to develop better integration schemes able to cope with longer timesteps. Moreover, the question is open as to how the horizon should recede when the control knots are separated by variable time intervals. The adaptation may also be done at the level of the model: hence, mixed control solutions, based on the combination of a short horizon accounting for whole-body dynamics and long-term horizon with centroidal preview, may offer interesting insights to save computation time [Li et al. 2021]. Aside from any frequency optimization discussion, we believe this work paves the way for the use of complex dynamics inside preview control locomotion frameworks. Because swing leg motion drastically affects angular momentum during walking, we advocate that taking into account the whole-body dynamics is necessary for heavy robots with high-inertia links.

On a more deep note, this thesis poses the general question of how to correctly formulate a control problem in the field of legged robotics. Whole-body dynamics MPC is still a very recent approach to locomotion, and not very well understood. In this work, a major amount of efforts has been spent in cost engineering and weight tuning, in an attempt to provide a first experimental validation of the concept of whole-body predictive control. The effect of these costs on the shaping of the solution needs to be understood more precisely in future developments. Likewise, the nature of the high-level references given to the solver is to be studied thoroughly. These references represent the information provided by the user to hint the desired optimal behavior to the solver, but it is still unclear if they can be generalized to perform any set of tasks. In this work, constant references for state and control were used along all experiments, but they tend to favor more conservative motions and do not prevent sharp spikes of torques during contact switches. An interesting solution to this issue consists in computing the dynamics to the 4th order, so as to be able to penalize jerk and acceleration in the cost function. Again, such an approach is hindered by the computational limits of current hardwares. This motivates even more the development of efficient optimization solvers leveraging parallelization techniques or fast sparse computation.

An alternative perspective on problem formulation is to integrate sensor feedback into the optimization process, as an extension to state feedback. The collaborative work of [Kleff et al. 2022] has taken a first step in this direction by introducing joint torque measures into the state, which amounts to modeling observation delays in sensors. Other information (touch, visual...) could be considered inside the framework to enhance the reactivity of the control with respect to unexpected changes in the environment. The question is then open as to how the huge amount of incoming data should be filtered to keep the optimization tractable.

In addition to costs, references and measures, contact optimization should be carefully considered when formulating the locomotion problem. This thesis has adopted a hybrid dynamics viewpoint, where contact timings are decided beforehand. The resulting framework is relatively simple to implement but requires to be

combined with an external planner to decide the optimal contact sequence ahead. Future perspectives on this matter include the integration of contact optimization into the control framework to further enhance the autonomy, reactivity and genericity of our approach. The motivation is to build an extensive and general scheme to tackle seamlessly all sort of manipulation and locomotion problems, with minimal changes in weights and cost functions. To this end, several paths can be explored. The first one consists in performing mixed-integer optimization directly inside the whole-body solver, at the price of having to use linear simplifications to make the problem tractable. A more promising solution would be to leverage an implicit contact formulation based on linear complementarity constraint inside the locomotion problem, so as to be able to continuously optimize force and control in the same process. However, this would mean to increase the state size along with the complexity of the resulting algorithm, hindering its deployment for real-time systems. On this topic, Augmented Lagrangian solvers [Jallet et al. 2022a] may provide the key to fast and efficient exploration of the hybrid modes of locomotion. Other possible approaches include the use of Monte-Carlo Tree Search to further accelerate the optimization of contact modes [Zhu et al. 2022], or leveraging efficient differentiable contact dynamics inside the simulation pipeline [Todorov et al. 2012].

Stepping aside from a purely software-oriented viewpoint, the importance of hardware design for robotics should also be considered when it comes to developing optimal policies. In our example, extensive tests of our MPC framework on the robot Pyrène were hindered by frequent hardware failures which prompted time-consuming reparations. To make things worse, it was sometimes difficult to tell if a particular behavior on the robot was caused by miscalibration, bad pose estimation or issues in the control law. Additionally, the heavy legs of Pyrène prevent it from doing very fast swing motions required when performing dynamic locomotion like running or jumping. To reach even higher performances, a promising angle of work would be to optimize conjointly the design of the robot with the control law used to make it move, as proposed in the work of [Fadini et al. 2022].

The integration of whole-body dynamics inside a predictive control framework contributes to show the limits of model-based optimization. Techniques like MPC, which exhibit local convergence at best, struggle to cope with huge search space dimensions. MPC actually amounts to solving repeatedly a very expensive control problem while starting from the previous computed solution: as such, it can hardly discover online a global optimum that stands far from its initial point in trajectory space. Besides, optimization schemes heavily rely on gradients to converge toward the optimum, but these gradients are difficult to define for discrete contact modes of locomotion.

On the other hand, RL techniques offer a complementary support for optimization by alleviating some parts, if all, of the problem, and boosting its exploration capacities. While full RL pipelines are starting to become common in robotics, particularly in the realm of quadruped locomotion, they still suffer from unpredictable convergence patterns, unstable behaviors and the inability to handle hard constraints. They are also very hard to scale to realistic applications and generally involve expert knowledge to be guided toward regions of most promising interest. Besides, data-based techniques have the disadvantage of operating like a black box solver and do not offer relevant insights on the complexity of human motion. In our opinion, the combination of RL and trajectory optimization has the potential to cancel the dowsides of both frameworks while strengthening their respective advantages. Using

optimization techniques to guide the RL agent through supervised learning allows to quickly discover relevant solutions and to learn how to generalize from future partial observations. Similarly, leveraging the synergy between local model-based control and global policy learning appears as a promising research direction [Mordatch et al. 2015; Lowrey et al. 2018]. Under this approach, the policy evaluated online is refined through optimization to enforce hard constraints and ensure stability. The resulting trajectories are plugged back into the learning process to accelerate value function learning. Another interesting angle of work consists in differentiating optimal problems with respect to tunable parameters so as to enable end-to-end learning of optimal policies, dynamics or value function [Jin et al. 2020]. Despite some recent achievements, hybrid techniques have not yet been implemented on fullscale systems performing in complex environments, partly because a common theoretical framework between RL and optimization remains to be built.

As research follows its way, we are approaching a level of autonomy in legged robotics that may be sufficient to let robots become our everyday life companions. User-friendly quadrupeds are already available to purchase and their potential applications in industrial, rescue or military contexts are on the rise. Humanoids are expected to follow the same path in the coming years, if we consider the latest impressive demonstrations of human-robot collaboration performed by Boston Dynamics. To the best of our knowledge, these demonstrations rely on a MPC framework with centroidal dynamics and full kinematics preview coupled with whole-body instantaneous control and cutting-edge actuation technology. Aside from these results, a pending question in robotics remains to decide which kind of autonomous behavior is to be implemented, on which robot, for which application. Genericity may indeed be reached at the price of efficiency, and perhaps there is a compromise to be found between a versatile but clumsy robot and a very precise but specialized machine.

Bibliography

Aceituno-Cabezas, Bernardo, Carlos Mastalli, Hongkai Dai, Michele Focchi, Andreea Radulescu, Darwin G. Caldwell, José Cappelletto, Juan C. Grieco, Gerardo Fernández-López, and Claudio Semini (July 2018). "Simultaneous Contact, Gait, and Motion Planning for Robust Multilegged Locomotion via Mixed-Integer Convex Optimization". In: *IEEE Robotics and Automation Letters* 3.3. Conference Name: IEEE Robotics and Automation Letters, pp. 2531–2538. ISSN: 2377-3766. DOI: 10.1109/LRA.2017.2779821.

Ajallooeian, Mostafa, Soha Pouya, Alexander Sproewitz, and Auke J. Ijspeert (May 2013). "Central Pattern Generators augmented with virtual model control for quadruped rough terrain locomotion". In: *2013 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 3321–3328. DOI: 10.1109/ICRA.2013.6631040.

Aller, Felix, Monika Harant, and Katja Mombaur (2022). "Optimization of Dynamic Sit-to-Stand Trajectories to Assess Whole-Body Motion Performance of the Humanoid Robot REEM-C". In: *Frontiers in Robotics and AI* 9. ISSN: 2296-9144. URL: https://www.frontiersin.org/articles/10.3389/frobt.2022.898696.

Amato, N.M. and Y. Wu (Apr. 1996). "A randomized roadmap method for path and manipulation planning". In: *Proceedings of IEEE International Conference on Robotics and Automation*. Vol. 1. ISSN: 1050-4729, 113–120 vol.1. DOI: 10.1109/ROBOT.1996.503582.

Amos, Brandon and J. Zico Kolter (Aug. 2017). "OptNet: differentiable optimization as a layer in neural networks". In: vol. 70. ICML'17. Sydney, NSW, Australia: JMLR.org, pp. 136–145.

Arimoto, S. (1984). "Stability and robustness of PID feedback control for robot manipulators of sensory capability". In: URL: https://www.semanticscholar.org/paper/Stability-and-robustness-of-PID-feedback-control-of-Arimoto/e6efcf314567f75f8202c404ae5e1f41e72bfcd9.

Assirelli, Alessandro, Fanny Risbourg, Gianni Lunardi, Thomas Flayols, and Nicolas Mansard (2022). *Whole-Body MPC without Foot References for the Locomotion of an Impedance-Controlled Robot*. en.

Audren, Hervé and Abderrahmane Kheddar (Apr. 2018). "3-D Robust Stability Polyhedron in Multicontact". In: *IEEE Transactions on Robotics* 34.2. Conference Name: IEEE Transactions on Robotics, pp. 388–403. ISSN: 1941-0468. DOI: 10.1109/TRO.2017.2786683.

Avila Belbute-Peres, Filipe de, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J. Zico Kolter (2018). "End-to-End Differentiable Physics for Learning and Control". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. URL: https://papers.nips.cc/paper/2018/hash/842424a1d0595b76ec4fa03c46e8d755-Abstract.html.

Aydinoglu, Alp and Michael Posa (Mar. 2022). "Real-Time Multi-Contact Model Predictive Control via ADMM". In: arXiv:2109.07076 [cs]. arXiv. DOI: 10.48550/arXiv.2109.07076. URL: http://arxiv.org/abs/2109.07076.

Bai, Long, Hao Hu, Xiaohong Chen, Yuanxi Sun, Chaoyang Ma, and Yuanhong Zhong (Aug. 2019). "CPG-Based Gait Generation of the Curved-Leg Hexapod Robot with Smooth Gait Transition". In: *Sensors* 19, p. 3705. DOI: 10.3390/s19173705.

Barraquand, Jerome and Jean-Claude Latombe (Dec. 1991). "Robot Motion Planning: A Distributed Representation Approach". In: *International Journal of Robotic Research - IJRR* 10, pp. 628–649. DOI: 10.1177/027836499101000604.

Bellman, Richard Ernest (Jan. 1954). *The Theory of Dynamic Programming*. en. Tech. rep. RAND Corporation. URL: https://www.rand.org/pubs/papers/P550.html (visited on 11/29/2022).

Bemporad, Alberto, Manfred Morari, Vivek Dua, and Efstratios N. Pistikopoulos (Jan. 2002). "The explicit linear quadratic regulator for constrained systems". en. In: *Automatica* 38.1, pp. 3–20. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(01)00174-1. URL: https://www.sciencedirect.com/science/article/pii/S0005109801001741.

Bentley, Jon Louis (Sept. 1975). "Multidimensional binary search trees used for associative searching". In: *Commun. ACM* 18.9, pp. 509–517. ISSN: 0001-0782. DOI: 10.1145/361002.361007. URL: https://doi.org/10.1145/361002.361007.

Berenson, Dmitry, Siddhartha S. Srinivasa, Dave Ferguson, and James J. Kuffner (May 2009). "Manipulation planning on constraint manifolds". In: *2009 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 625–632. DOI: 10.1109/ROBOT.2009.5152399.

Bledt, Gerardo and Sangbae Kim (Nov. 2019). "Implementing Regularized Predictive Control for Simultaneous Real-Time Footstep and Ground Reaction Force Optimization". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 6316–6323. DOI: 10.1109/IROS40897.2019.8968031.

— (May 2020). "Extracting Legged Locomotion Heuristics with Regularized Predictive Control". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 406–412. DOI: 10.1109/ICRA40945.2020.9197488.

Bobrow, J. E. (1982). "Optimal Control of Robotic Manipulators". Ph. D. dissertation. Los Angeles: University of California.

Bouyarmane, Karim, Kevin Chappellet, Joris Vaillant, and Abderrahmane Kheddar (Feb. 2019). "Quadratic Programming for Multirobot and Task-Space Force Control". In: *IEEE Transactions on Robotics* 35.1. Conference Name: IEEE Transactions on Robotics, pp. 64–77. ISSN: 1941-0468. DOI: 10.1109/TRO.2018.2876782.

Brasseur, Camille, Alexander Sherikov, Cyrille Collette, Dimitar Dimitrov, and Pierre-Brice Wieber (Nov. 2015). "A robust linear MPC approach to online generation of 3D biped walking motion". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 595–601. DOI: 10.1109/HUMANOIDS.2015.7363423.

Brogliato, Bernard (2016). *Nonsmooth Mechanics: Models, Dynamics and Control*. en. Communications and Control Engineering. Cham: Springer International Publishing. ISBN: 978-3-319-28662-4 978-3-319-28664-8. DOI: 10.1007/978-3-319-28664-8. URL: http://link.springer.com/10.1007/978-3-319-28664-8.

Brooks, Rodney A. (Jan. 1991). "Intelligence without representation". en. In: *Artificial Intelligence* 47.1, pp. 139–159. ISSN: 0004-3702. DOI: 10.1016/0004-3702(91)90053-M. URL: https://www.sciencedirect.com/science/article/pii/000437029190053M.

Buchli, Jonas, Freek Stulp, Evangelos Theodorou, and Stefan Schaal (June 2011). "Learning variable impedance control". In: *I. J. Robotic Res.* 30, pp. 820–833. DOI: 10.1177/0278364911402527.

Budhiraja, Rohan, Justin Carpentier, Carlos Mastalli, and Nicolas Mansard (Nov. 2018). "Differential Dynamic Programming for Multi-Phase Rigid Contact Dynamics". en. In: *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. Beijing, China: IEEE, pp. 1–9. ISBN: 978-1-5386-7283-9. DOI: 10.1109/HUMANOIDS.2018.8624925. URL: https://ieeexplore.ieee.org/document/8624925/.

Budhiraja, Rohan, Justin Carpentier, and Nicolas Mansard (May 2019). "Dynamics Consensus between Centroidal and Whole-Body Models for Locomotion of Legged Robots". In: *2019 International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 6727–6733. DOI: 10.1109/ICRA.2019.8793878.

Canny, John F. (1988). *The complexity of robot motion planning*. Cambridge, MA, USA: MIT Press. ISBN: 978-0-262-03136-3.

Caron, Stephane, Quang Cuong Pham, and Yoshihiko Nakamura (July 2015a). "Leveraging Cone Double Description for Multi-contact Stability of Humanoids with Applications to Statics and Dynamics". en. In: *Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation. ISBN: 978-0-9923747-1-6. DOI: 10.15607/RSS.2015.XI.028. URL: http://www.roboticsproceedings.org/rss11/p28.pdf.

Caron, Stéphane, Quang-Cuong Pham, and Yoshihiko Nakamura (May 2015b). "Stability of surface contacts for humanoid robots: Closed-form formulae of the Contact Wrench Cone for rectangular support areas". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729, pp. 5107–5112. DOI: 10.1109/ICRA.2015.7139910.

Caron, Stéphane, Abderrahmane Kheddar, and Olivier Tempier (May 2019). "Stair Climbing Stabilization of the HRP-4 Humanoid Robot using Whole-body Admittance Control". In: *2019 International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 277–283. DOI: 10.1109/ICRA.2019.8794348.

Carpentier, Justin and Nicolas Mansard (June 2018a). "Analytical Derivatives of Rigid Body Dynamics Algorithms". en. In: *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation. ISBN: 978-0-9923747-4-7. DOI: 10.15607/RSS.2018.XIV.038. URL: http://www.roboticsproceedings.org/rss14/p38.pdf.

— (Dec. 2018b). "Multicontact Locomotion of Legged Robots". In: *IEEE Transactions on Robotics* 34.6. Conference Name: IEEE Transactions on Robotics, pp. 1441–1460. ISSN: 1941-0468. DOI: 10.1109/TRO.2018.2862902.

Carpentier, Justin, Steve Tonneau, Maximilien Naveau, Olivier Stasse, and Nicolas Mansard (May 2016). "A versatile and efficient pattern generator for generalized legged locomotion". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3555–3561. DOI: 10.1109/ICRA.2016.7487538.

Carpentier, Justin, Rohan Budhiraja, and Nicolas Mansard (July 2017). "Learning Feasibility Constraints for Multicontact Locomotion of Legged Robots". en. In: *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation. ISBN: 978-0-9923747-3-0. DOI: 10.15607/RSS.2017.XIII.031. URL: http://www.roboticsproceedings.org/rss13/p31.pdf.

Carpentier, Justin, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard (Jan. 2019). "The Pinocchio C++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives". In: *2019 IEEE/SICE International Symposium on*

System Integration (SII). ISSN: 2474-2325, pp. 614–619. DOI: 10.1109/SII.2019.8700380.

Cauligi, A., P. Culbertson, E. Schmerling, M. Schwager, B. Stellato, and M. Pavone (July 2021). "CoCo: Online Mixed-Integer Control via Supervised Learning". In: *IEEE Robotics and Automation Letters* 7.2. arXiv:2107.08143 [cs], pp. 1447–1454. DOI: 10.48550/arXiv.2107.08143. URL: http://arxiv.org/abs/2107.08143.

Channon, P. H., S. H. Hopkins, and D. T. Pham (Mar. 1992). "Derivation of optimal walking motions for a bipedal walking robot". en. In: *Robotica* 10.2. Publisher: Cambridge University Press, pp. 165–172. ISSN: 1469-8668, 0263-5747. DOI: 10.1017/S026357470000758X. URL: https://www.cambridge.org/core/journals/robotica/article/abs/derivation-of-optimal-walking-motions-for-a-bipedal-walking-robot/25E27DD62C73AE14587062423D901F32.

Chatzinikolaidis, Iordanis and Zhibin Li (Apr. 2021). "Trajectory Optimization of Contact-Rich Motions Using Implicit Differential Dynamic Programming". In: *IEEE Robotics and Automation Letters* 6.2. Conference Name: IEEE Robotics and Automation Letters, pp. 2626–2633. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3061341.

Cheng, Jianguo, Richard B. Stein, Ksenija Jovanovic, Ken Yoshida, David J. Bennett, and Yingchun Han (June 1998). "Identification, Localization, and Modulation of Neural Networks for Walking in the Mudpuppy (Necturus Maculatus) Spinal Cord". In: *J Neurosci* 18.11, pp. 4295–4304. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.18-11-04295.1998. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6792799/.

Chernousko, F. L. (1994). "Optimization in Control of Robots". en. In: *Computational Optimal Control*. Ed. by R. Bulirsch and D. Kraft. ISNM International Series of Numerical Mathematics. Basel: Birkhäuser, pp. 19–28. ISBN: 978-3-0348-8497-6. DOI: 10.1007/978-3-0348-8497-6_2. URL: https://doi.org/10.1007/978-3-0348-8497-6_2.

Collette, Cyrille, Alain Micaelli, Claude Andriot, and Pierre Lemerle (May 2008). "Robust balance optimization control of humanoid robots with multiple non coplanar grasps and frictional contacts". In: *2008 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 3187–3193. DOI: 10.1109/ROBOT.2008.4543696.

Collins, Steven H., Martijn Wisse, and Andy Ruina (July 2001). "A Three-Dimensional Passive-Dynamic Walking Robot with Two Legs and Knees". en. In: *The International Journal of Robotics Research* 20.7. Publisher: SAGE Publications Ltd STM, pp. 607–615. ISSN: 0278-3649. DOI: 10.1177/02783640122067561. URL: https://doi.org/10.1177/02783640122067561.

Corbères, Thomas, Thomas Flayols, Pierre-Alexandre Léziart, Rohan Budhiraja, Philippe Souères, Guilhem Saurel, and Nicolas Mansard (May 2021). "Comparison of predictive controllers for locomotion and balance recovery of quadruped robots". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 5021–5027. DOI: 10.1109/ICRA48506.2021.9560976.

Dai, Hongkai and Russ Tedrake (Nov. 2016). "Planning robust walking motion on uneven terrain via convex optimization". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 579–586. DOI: 10.1109/HUMANOIDS.2016.7803333.

Dai, Hongkai, Andrés Valenzuela, and Russ Tedrake (Nov. 2014). "Whole-body motion planning with centroidal dynamics and full kinematics". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 295–302. DOI: 10.1109/HUMANOIDS.2014.7041375.

Dantec, Ewen, Rohan Budhiraja, Adria Roig, Teguh Lembono, Guilhem Saurel, Olivier Stasse, Pierre Fernbach, Steve Tonneau, Sethu Vijayakumar, Sylvain Calinon, Michel Taix, and Nicolas Mansard (May 2021). "Whole Body Model Predictive Control with a Memory of Motion: Experiments on a Torque-Controlled Talos". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 8202–8208. DOI: 10.1109/ICRA48506.2021.9560742.

Dantec, Ewen, Michel Taïx, and Nicolas Mansard (Apr. 2022a). "First Order Approximation of Model Predictive Control Solutions for High Frequency Feedback". In: *IEEE Robotics and Automation Letters* 7.2. Conference Name: IEEE Robotics and Automation Letters, pp. 4448–4455. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3149573.

Dantec, Ewen, Maximilien Naveau, Pierre Fernbach, Nahuel Villa, Guilhem Saurel, Olivier Stasse, Michel Taix, and Nicolas Mansard (Nov. 2022b). "Whole-Body Model Predictive Control for Biped Locomotion on a Torque-Controlled Humanoid Robot". In: *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 638–644. DOI: 10.1109/Humanoids53995.2022.10000129.

De Luca, Andrea and Giuseppe Oriolo (Nov. 1997). "Nonholonomic behavior in redundant robots under kinematic control". In: *Robotics and Automation, IEEE Transactions on* 13, pp. 776–782. DOI: 10.1109/70.631239.

Degrave, Jonas, Michiel Hermans, Joni Dambre, and Francis wyffels (2019). "A Differentiable Physics Engine for Deep Learning in Robotics". In: *Frontiers in Neurorobotics* 13. ISSN: 1662-5218. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2019.00006.

Deits, Robin and Russ Tedrake (Nov. 2014). "Footstep planning on uneven terrain with mixed-integer convex optimization". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 279–286. DOI: 10.1109/HUMANOIDS.2014.7041373.

Deits, Robin, Twan Koolen, and Russ Tedrake (May 2019). "LVIS: Learning from Value Function Intervals for Contact-Aware Robot Controllers". In: *2019 International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 7762–7768. DOI: 10.1109/ICRA.2019.8794352.

Di Carlo, Jared, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim (Oct. 2018). "Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 1–9. DOI: 10.1109/IROS.2018.8594448.

Diedam, Holger, Dimitar Dimitrov, Pierre-Brice Wieber, Katja Mombaur, and Moritz Diehl (Sept. 2008). "Online walking gait generation with adaptive foot positioning through Linear Model Predictive control". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 1121–1126. DOI: 10.1109/IROS.2008.4651055.

Diehl, Moritz, Hans Bock, and Johannes Schlöder (Jan. 2005). "A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control". In: *SIAM J. Control and Optimization* 43, pp. 1714–1736. DOI: 10.1137/S0363012902400713.

Diehl, Moritz, Hans Bock, Holger Diedam, and Pierre-Brice Wieber (July 2007). "Fast Direct Multiple Shooting Algorithms for Optimal Robot Control". In: *Lecture Notes in Control and Information Sciences* 340. ISSN: 978-3-540-36118-3. DOI: 10.1007/978-3-540-36119-0_4.

Diftler, M. A., T. D. Ahlstrom, R. O. Ambrose, N. A. Radford, C. A. Joyce, N. De La Pena, A. H. Parsons, and A. L. Noblitt (Mar. 2012). "Robonaut 2 — Initial

activities on-board the ISS". In: *2012 IEEE Aerospace Conference*. ISSN: 1095-323X, pp. 1–12. DOI: 10.1109/AERO.2012.6187268.

Ding, Yanran, Abhishek Pandala, Chuanzheng Li, Young-Ha Shin, and Hae-Won Park (Dec. 2020). "Representation-Free Model Predictive Control for Dynamic Motions in Quadrupeds". In: *IEEE Transactions on Robotics* PP. DOI: 10.1109/TRO.2020.3046415.

Donald, Bruce, Patrick Xavier, John Canny, and John Reif (Nov. 1993). "Kinodynamic Motion Planning." In: *J. ACM* 40, pp. 1048–1066. DOI: 10.1145/174147.174150.

Doshi, Neel, Kaushik Jayaram, Benjamin Goldberg, Zac Manchester, Robert Wood, and Scott Kuindersma (June 2018). "Contact-Implicit Optimization of Locomotion Trajectories for a Quadrupedal Microrobot". In: DOI: 10.15607/RSS.2018.XIV.041.

Dzeladini, Florin, Nadine Ait-Bouziad, and Auke Ijspeert (2018). "CPG-Based Control of Humanoid Robot Locomotion". en. In: *Humanoid Robotics: A Reference*. Ed. by Ambarish Goswami and Prahlad Vadakkepat. Dordrecht: Springer Netherlands, pp. 1–35. ISBN: 978-94-007-7194-9. DOI: 10.1007/978-94-007-7194-9_49-1. URL: https://doi.org/10.1007/978-94-007-7194-9_49-1.

Englsberger, Johannes and Christian Ott (Nov. 2012). "Integration of vertical COM motion and angular momentum in an extended Capture Point tracking controller for bipedal walking". In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. ISSN: 2164-0580, pp. 183–189. DOI: 10.1109/HUMANOIDS.2012.6651518.

Englsberger, Johannes, Christian Ott, Maximo A. Roa, Alin Albu-Schäffer, and Gerhard Hirzinger (Sept. 2011). "Bipedal walking control based on Capture Point dynamics". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 4420–4427. DOI: 10.1109/IROS.2011.6094435.

Englsberger, Johannes, Christian Ott, and Alin Albu-Schäffer (Nov. 2013). "Three-dimensional bipedal walking control using Divergent Component of Motion". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 2600–2607. DOI: 10.1109/IROS.2013.6696723.

Englsberger, Johannes, Twan Koolen, Sylvain Bertrand, Jerry Pratt, Christian Ott, and Alin Albu-Schäffer (Sept. 2014). "Trajectory generation for continuous leg forces during double support and heel-to-toe shift based on divergent component of motion". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 4022–4029. DOI: 10.1109/IROS.2014.6943128.

Erez, Tom, Kendall Lowrey, Yuval Tassa, Vikash Kumar, Svetoslav Kolev, and Emanuel Todorov (Oct. 2013). "An integrated system for real-time model predictive control of humanoid robots". In: *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 292–299. DOI: 10.1109/HUMANOIDS.2013.7029990.

Fadini, Gabriele, Thomas Flayols, Andrea Del Prete, and Philippe Souères (Oct. 2022). "Simulation Aided Co-Design for Robust Robot Optimization". In: *IEEE Robotics and Automation Letters* 7.4. Conference Name: IEEE Robotics and Automation Letters, pp. 11306–11313. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3200142.

Farshidian, Farbod, Diego Pardo, and Jonas Buchli (Sept. 2016). "Sequential Linear Quadratic Optimal Control for Nonlinear Switched Systems". In: *IFAC-PapersOnLine* 50. DOI: 10.1016/j.ifacol.2017.08.291.

Farshidian, Farbod, Edo Jelavić, Asutosh Satapathy, Markus Giftthaler, and Jonas Buchli (Nov. 2017). "Real-Time Motion Planning of Legged Robots: A Model Predictive Control Approach". In: *2017 IEEE-RAS 17th International Conference*

on Humanoid Robotics (Humanoids). arXiv:1710.04029 [cs], pp. 577–584. DOI: 10.1109/HUMANOIDS.2017.8246930. URL: http://arxiv.org/abs/1710.04029.

Featherstone, Roy (2008). *Rigid Body Dynamics Algorithms*. en. Boston, MA: Springer US. ISBN: 978-0-387-74314-1. DOI: 10.1007/978-1-4899-7560-7. URL: http://link.springer.com/10.1007/978-1-4899-7560-7.

Flash, Tamar, Yaron Meirovitch, and Avi Barliya (Apr. 2013). "Models of human movement: Trajectory planning and inverse kinematics studies". en. In: *Robotics and Autonomous Systems*. Models and Technologies for Multi-modal Skill Training 61.4, pp. 330–339. ISSN: 0921-8890. DOI: 10.1016/j.robot.2012.09.020. URL: https://www.sciencedirect.com/science/article/pii/S0921889012001741.

Focchi, Michele, Andrea del Prete, Ioannis Havoutis, Roy Featherstone, Darwin G. Caldwell, and Claudio Semini (Jan. 2017). "High-slope terrain locomotion for torque-controlled quadruped robots". en. In: *Auton Robot* 41.1, pp. 259–272. ISSN: 0929-5593, 1573-7527. DOI: 10.1007/s10514-016-9573-1. URL: http://link.springer.com/10.1007/s10514-016-9573-1.

Galliker, Manuel Y., Noel Csomay-Shanklin, Ruben Grandia, Andrew J. Taylor, Farbod Farshidian, Marco Hutter, and Aaron D. Ames (2022). "Bipedal Locomotion with Nonlinear Model Predictive Control: Online Gait Generation using Whole-Body Dynamics". In: arXiv:2203.07429 [cs]. IEEE. DOI: 10.48550/arXiv.2203.07429. URL: http://arxiv.org/abs/2203.07429.

Gangapurwala, Siddhant, Mathieu Geisert, Romeo Orsolino, Maurice Fallon, and Ioannis Havoutis (May 2021). "Real-Time Trajectory Adaptation for Quadrupedal Locomotion using Deep Reinforcement Learning". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 5973–5979. DOI: 10.1109/ICRA48506.2021.9561639.

Garcia, Mariano, Anindya Chatterjee, Andy Ruina, and Michael Coleman (Apr. 1998). "The simplest walking model: Stability, complexity, and scaling". In: *Journal of biomechanical engineering* 120, pp. 281–8.

Garofalo, Gianluca, Christian Ott, and Alin Albu-Schäffer (May 2012). "Walking control of fully actuated robots based on the Bipedal SLIP model". In: *2012 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 1456–1463. DOI: 10.1109/ICRA.2012.6225272.

Gazar, Ahmad, Gabriele Nava, Francisco Javier Andrade Chavez, and Daniele Pucci (Feb. 2021). "Jerk Control of Floating Base Systems With Contact-Stable Parameterized Force Feedback". In: *IEEE Transactions on Robotics* 37.1. Conference Name: IEEE Transactions on Robotics, pp. 1–15. ISSN: 1941-0468. DOI: 10.1109/TRO.2020.3005547.

Gienger, M., H. Janssen, and C. Goerick (Dec. 2005). "Task-oriented whole body motion for humanoid robots". In: *5th IEEE-RAS International Conference on Humanoid Robots, 2005.* ISSN: 2164-0580, pp. 238–244. DOI: 10.1109/ICHR.2005.1573574.

Gill, Philip E., Walter Murray, and Michael A. Saunders (Jan. 2002). "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization". In: *SIAM J. Optim.* 12.4. Publisher: Society for Industrial and Applied Mathematics, pp. 979–1006. ISSN: 1052-6234. DOI: 10.1137/S1052623499350013. URL: https://epubs.siam.org/doi/10.1137/S1052623499350013.

Gong, Yukai, Ross Hartley, Xingye Da, Ayonga Hereid, Omar Harib, Jiunn-Kai Huang, and Jessy Grizzle (July 2019). "Feedback Control of a Cassie Bipedal Robot: Walking, Standing, and Riding a Segway". In: *2019 American Control Conference (ACC)*. ISSN: 2378-5861, pp. 4559–4566. DOI: 10.23919/ACC.2019.8814833.

Grandia, Ruben, Farbod Farshidian, René Ranftl, and Marco Hutter (Nov. 2019). "Feedback MPC for Torque-Controlled Legged Robots". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 4730–4737. DOI: 10.1109/IROS40897.2019.8968251.

Griffin, Robert J. and Alexander Leonessa (May 2016). "Model predictive control for dynamic footstep adjustment using the divergent component of motion". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1763–1768. DOI: 10.1109/ICRA.2016.7487320.

Grizzle, Jessy and Chevallereau Christine (June 2017). "Virtual Constraints and Hybrid Zero Dynamics for Realizing Underactuated Bipedal Locomotion". In:

Guo, Yijie, Mingwei Zhang, Hao Dong, and Mingguo Zhao (June 2021). "Fast Online Planning for Bipedal Locomotion via Centroidal Model Predictive Gait Synthesis". In: *IEEE Robotics and Automation Letters* PP, pp. 1–1. DOI: 10.1109/LRA.2021.3092268.

Gurobi Optimization, LLC (2023). *Gurobi Optimizer Reference Manual*. URL: https://www.gurobi.com.

Hargraves, C. and Stephen Paris (July 1987). "Direct Trajectory Optimization Using Nonlinear Programming and Collocation". In: *AIAA J. Guidance* 10, pp. 338–342. DOI: 10.2514/3.20223.

Hartline, Jeffrey and R. Libeskind-Hadas (Sept. 2003). "The Computational Complexity of Motion Planning". In: *Siam Review - SIAM REV* 45, pp. 543–557. DOI: 10.1137/S0036144501395517.

Henze, Bernd, Alexander Dietrich, and Christian Ott (July 2016). "An Approach to Combine Balancing with Hierarchical Whole-Body Control for Legged Humanoid Robots". In: *IEEE Robotics and Automation Letters* 1.2. Conference Name: IEEE Robotics and Automation Letters, pp. 700–707. ISSN: 2377-3766. DOI: 10.1109/LRA.2015.2512933.

Herdt, Andrei, Nicolas Perrin, and Pierre-Brice Wieber (Oct. 2010). "Walking without thinking about it". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 190–195. DOI: 10.1109/IROS.2010.5654429.

Hereid, Ayonga, Christian M. Hubicki, Eric A. Cousineau, Jonathan W. Hurst, and Aaron D. Ames (May 2015). "Hybrid zero dynamics based multiple shooting optimization with applications to robotic walking". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729, pp. 5734–5740. DOI: 10.1109/ICRA.2015.7140002.

Herzog, Alexander, Ludovic Righetti, Felix Grimminger, Peter Pastor, and Stefan Schaal (Sept. 2014). "Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 981–988. DOI: 10.1109/IROS.2014.6942678.

Herzog, Alexander, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti (Mar. 2016a). "Momentum Control with Hierarchical Inverse Dynamics on a Torque-Controlled Humanoid". In: *Auton Robot* 40.3. arXiv:1410.7284 [cs], pp. 473–491. ISSN: 0929-5593, 1573-7527. DOI: 10.1007/s10514-015-9476-6. URL: http://arxiv.org/abs/1410.7284.

Herzog, Alexander, Stefan Schaal, and Ludovic Righetti (Oct. 2016b). "Structured contact force optimization for kino-dynamic motion generation". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 2703–2710. DOI: 10.1109/IROS.2016.7759420.

Hirai, K., M. Hirose, Y. Haikawa, and T. Takenaka (May 1998). "The development of Honda humanoid robot". In: *Proceedings. 1998 IEEE International Conference on*

Robotics and Automation (Cat. No.98CH36146). Vol. 2. ISSN: 1050-4729, 1321–1326 vol.2. DOI: 10.1109/ROBOT.1998.677288.

Hirukawa, H., S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa (May 2006). "A universal stability criterion of the foot contact of legged robots - adios ZMP". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* ISSN: 1050-4729, pp. 1976–1983. DOI: 10.1109/ROBOT.2006.1641995.

Hirukawa, Hirohisa, Shizuko Hattori, Shuuji Kajita, Kensuke Harada, Kenji Kaneko, Fumio Kanehiro, Mitsuharu Morisawa, and Shinichiro Nakaoka (Apr. 2007). "A Pattern Generator of Humanoid Robots Walking on a Rough Terrain". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 2181–2187. DOI: 10.1109/ROBOT.2007.363644.

Holmes, Philip, Robert J. Full, Dan Koditschek, and John Guckenheimer (Jan. 2006). "The Dynamics of Legged Locomotion: Models, Analyses, and Challenges". In: *SIAM Rev.* 48.2. Publisher: Society for Industrial and Applied Mathematics, pp. 207–304. ISSN: 0036-1445. DOI: 10.1137/S0036144504445133. URL: https://epubs.siam.org/doi/10.1137/S0036144504445133.

Hopkins, Michael A., Dennis W. Hong, and Alexander Leonessa (Nov. 2014). "Humanoid locomotion on uneven terrain using the time-varying divergent component of motion". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 266–272. DOI: 10.1109/HUMANOIDS.2014.7041371.

Howell, Taylor A., Brian E. Jackson, and Zachary Manchester (Nov. 2019). "ALTRO: A Fast Solver for Constrained Trajectory Optimization". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 7674–7679. DOI: 10.1109/IROS40897.2019.8967788.

Huang, Qiang, K. Kaneko, K. Yokoi, S. Kajita, T. Kotoku, N. Koyachi, H. Arai, N. Imamura, K. Komoriya, and K. Tanie (Apr. 2000). "Balance control of a piped robot combining off-line pattern with real-time modification". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 4. ISSN: 1050-4729, 3346–3352 vol.4. DOI: 10.1109/ROBOT.2000.845227.

Huang, Qiang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, and K. Tanie (June 2001). "Planning walking patterns for a biped robot". In: *IEEE Transactions on Robotics and Automation* 17.3. Conference Name: IEEE Transactions on Robotics and Automation, pp. 280–289. ISSN: 2374-958X. DOI: 10.1109/70.938385.

Hwangbo, Jemin, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter (Jan. 2019). "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26. Publisher: American Association for the Advancement of Science, eaau5872. DOI: 10.1126/scirobotics.aau5872. URL: https://www.science.org/doi/10.1126/scirobotics.aau5872.

Hyon, Sang-Ho, Joshua G. Hale, and Gordon Cheng (Oct. 2007). "Full-Body Compliant Human–Humanoid Interaction: Balancing in the Presence of Unknown External Forces". In: *IEEE Transactions on Robotics* 23.5. Conference Name: IEEE Transactions on Robotics, pp. 884–898. ISSN: 1941-0468. DOI: 10.1109/TRO.2007.904896.

IEEE Spectrum (2015). *A Compilation of Robots Falling Down at the DARPA Robotics Challenge*. URL: https://www.youtube.com/watch?v=g0TaYhjpOfo.

Islam, Shafiqul and Peter X. Liu (Feb. 2011). "PD Output Feedback Control Design for Industrial Robotic Manipulators". In: *IEEE/ASME Transactions on Mechatronics* 16.1. Conference Name: IEEE/ASME Transactions on Mechatronics, pp. 187–197. ISSN: 1941-014X. DOI: 10.1109/TMECH.2009.2038374.

Jallet, Wilson, Antoine Bambade, Nicolas Mansard, and Justin Carpentier (Oct. 2022a). "Constrained Differential Dynamic Programming: A primal-dual augmented Lagrangian approach". en. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Kyoto, Japan: IEEE, pp. 13371–13378. ISBN: 978-1-66547-927-1. DOI: 10.1109/IROS47612.2022.9981586. URL: https://ieeexplore.ieee.org/document/9981586/.

Jallet, Wilson, Nicolas Mansard, and Justin Carpentier (May 2022b). "Implicit Differential Dynamic Programming". In: *2022 International Conference on Robotics and Automation (ICRA)*, pp. 1455–1461. DOI: 10.1109/ICRA46639.2022.9811647.

Jin, Wanxin, Zhaoran Wang, Zhuoran Yang, and Shaoshuai Mou (2020). "Pontryagin Differentiable Programming: An End-to-End Learning and Control Framework". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 7979–7992. URL: https://proceedings.neurips.cc/paper/2020/hash/5a7b238ba0f6502e5d6be14424b20ded-Abstract.html.

KAGAMI, S., K. NISHIWAKI, T. KITAGAWA, T Sugihiara, Masayuki Inaba, and H. INOUE (Jan. 2000). "A Fast Generation Method of a Dynamically Stable Humanoid Robot Trajectory with Enhanced ZMP Constraint". In:

Kajita, S. and K. Tani (Apr. 1991). "Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode". In: *1991 IEEE International Conference on Robotics and Automation Proceedings*, 1405–1411 vol.2. DOI: 10.1109/ROBOT.1991.131811.

Kajita, S., F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa (Sept. 2003). "Biped walking pattern generation by using preview control of zero-moment point". In: *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*. Vol. 2. ISSN: 1050-4729, 1620–1626 vol.2. DOI: 10.1109/ROBOT.2003.1241826.

Kalakrishnan, Mrinal, Jonas Buchli, Peter Pastor, Michael Mistry, and Stefan Schaal (Feb. 2011). "Learning, planning, and control for quadruped locomotion over challenging terrain". In: *I. J. Robotic Res.* 30, pp. 236–258. DOI: 10.1177/0278364910388677.

Kamel, Mina, Kostas Alexis, Markus Achtelik, and Roland Siegwart (Sept. 2015). "Fast nonlinear model predictive control for multicopter attitude tracking on SO(3)". In: *2015 IEEE Conference on Control Applications (CCA)*. ISSN: 1085-1992, pp. 1160–1166. DOI: 10.1109/CCA.2015.7320769.

Kanoun, Oussama, Florent Lamiraux, Pierre-Brice Wieber, Fumio Kanehiro, Eiichi Yoshida, and Jean-Paul Laumond (May 2009). "Prioritizing linear equality and inequality systems: Application to local motion planning for redundant robots". In: *2009 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 2939–2944. DOI: 10.1109/ROBOT.2009.5152293.

Karaman, Sertac and Emilio Frazzoli (June 2011). "Sampling-based Algorithms for Optimal Motion Planning". In: *International Journal of Robotic Research - IJRR* 30, pp. 846–894. DOI: 10.1177/0278364911406761.

Katayama, Sotaro and Toshiyuki Ohtsuka (Oct. 2022). "Whole-body model predictive control with rigid contacts via online switching time optimization". In: arXiv:2203.00997 [cs, math]. IEEE. DOI: 10.48550/arXiv.2203.00997. URL: http://arxiv.org/abs/2203.00997.

Kavraki, L.E., P. Svestka, J.-C. Latombe, and M.H. Overmars (Aug. 1996). "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE Transactions on Robotics and Automation* 12.4. Conference Name: IEEE Transactions on Robotics and Automation, pp. 566–580. ISSN: 2374-958X. DOI: 10.1109/70.508439.

Kazdadi, Sarah, Justin Carpentier, and Jean Ponce (2021). "Equality Constrained Differential Dynamic Programming". In: DOI: 10.1109/ICRA48506.2021.9561339.

Khatib, O. (Feb. 1987). "A unified approach for motion and force control of robot manipulators: The operational space formulation". In: *IEEE Journal on Robotics and Automation* 3.1. Conference Name: IEEE Journal on Robotics and Automation, pp. 43–53. ISSN: 2374-8710. DOI: 10.1109/JRA.1987.1087068.

Khatib, Oussama (Mar. 1986). "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots". en. In: *The International Journal of Robotics Research* 5.1. Publisher: SAGE Publications Ltd STM, pp. 90–98. ISSN: 0278-3649. DOI: 10.1177/027836498600500106. URL: https://doi.org/10.1177/027836498600500106.

Khatib, Oussama, Luis Sentis, Jaeheung Park, and James Warren (Mar. 2004). "Whole-Body Dynamic Behavior and Control of Human-like Robots." In: *International Journal of Humanoid Robotics* 10, pp. 29–43. DOI: 10.1142/S0219843604000058.

Khazoom, Charles and Sangbae Kim (May 2022). "Humanoid Arm Motion Planning for Improved Disturbance Recovery Using Model Hierarchy Predictive Control". In: *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6607–6613. DOI: 10.1109/ICRA46639.2022.9811878.

Kheddar, Abderrahmane, Stephane Caron, Pierre Gergondet, Andrew Comport, Arnaud Tanguy, Christian Ott, Bernd Henze, George Mesesan, Johannes Englsberger, Máximo A. Roa, Pierre-Brice Wieber, François Chaumette, Fabien Spindler, Giuseppe Oriolo, Leonardo Lanari, Adrien Escande, Kevin Chappellet, Fumio Kanehiro, and Patrice Rabaté (Dec. 2019). "Humanoid Robots in Aircraft Manufacturing: The Airbus Use Cases". In: *IEEE Robotics & Automation Magazine* 26.4. Conference Name: IEEE Robotics & Automation Magazine, pp. 30–45. ISSN: 1558-223X. DOI: 10.1109/MRA.2019.2943395.

Kim, Sangbae, Cecilia Laschi, and Barry Trimmer (May 2013). "Soft robotics: a bioinspired evolution in robotics". en. In: *Trends in Biotechnology* 31.5, pp. 287–294. ISSN: 0167-7799. DOI: 10.1016/j.tibtech.2013.03.002. URL: https://www.sciencedirect.com/science/article/pii/S0167779913000632.

Kleff, Sébastien, Ewen Dantec, Guilhem Saurel, Nicolas Mansard, and Ludovic Righetti (Oct. 2022). "Introducing Force Feedback in Model Predictive Control". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 13379–13385. DOI: 10.1109/IROS47612.2022.9982003.

Kleida, Danae (2018). "On the Technological Conditions of the Representation of Movement: Dance Notation Systems & Annotation Practices as Gestures". PhD thesis. DOI: 10.13140/RG.2.2.27077.29921.

Knust, Albrecht (1978). *Dictionary of Kinetography Laban (Labanotion)*. en. Google-Books-ID: 8_Y9AQAACAAJ. Plays, Incorporated. ISBN: 978-0-8238-0225-8.

Koenemann, J., A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard (Sept. 2015). "Whole-body model-predictive control applied to the HRP-2 humanoid". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3346–3351. DOI: 10.1109/IROS.2015.7353843.

Komura, T., A. Nagano, H. Leung, and Y. Shinagawa (Sept. 2005). "Simulating pathological gait using the enhanced linear inverted pendulum model". In: *IEEE Transactions on Biomedical Engineering* 52.9. Conference Name: IEEE Transactions on Biomedical Engineering, pp. 1502–1513. ISSN: 1558-2531. DOI: 10.1109/TBME.2005.851530.

Koolen, Twan, Michael Posa, and Russ Tedrake (Nov. 2016a). "Balance control using center of mass height variation: Limitations imposed by unilateral contact". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 8–15. DOI: 10.1109/HUMANOIDS.2016.7803247.

Koolen, Twan, Sylvain Bertrand, Gray Thomas, Tomas de Boer, Tingfan Wu, Jesper Smith, Johannes Englsberger, and Jerry Pratt (Mar. 2016b). "Design of a Momentum-Based Control Framework and Application to the Humanoid Robot Atlas". en. In: *Int. J. Human. Robot.* 13.01, p. 1650007. ISSN: 0219-8436, 1793-6942. DOI: 10.1142/S0219843616500079. URL: https://www.worldscientific.com/doi/abs/10.1142/S0219843616500079.

Kudruss, M., M. Naveau, O. Stasse, N. Mansard, C. Kirches, P. Soueres, and K. Mombaur (Nov. 2015). "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)*, pp. 684–689. DOI: 10.1109/HUMANOIDS.2015.7363428.

Kuo, Arthur D. and Felix E. Zajac (Jan. 1993). "Human standing posture: multi-joint movement strategies based on biomechanical constraints". en. In: *Progress in Brain Research*. Ed. by J. H. J. Allum, D. J. Allum-Mecklenburg, F. P. Harris, and R. Probst. Vol. 97. Natural and Artificial Control of Hearing and Balance. Elsevier, pp. 349–358. DOI: 10.1016/S0079-6123(08)62294-3. URL: https://www.sciencedirect.com/science/article/pii/S0079612308622943.

Kuwata, Yoshiaki, Justin Teo, Gaston Fiore, Sertac Karaman, Emilio Frazzoli, and Jonathan P. How (Sept. 2009). "Real-Time Motion Planning With Applications to Autonomous Urban Driving". In: *IEEE Transactions on Control Systems Technology* 17.5. Conference Name: IEEE Transactions on Control Systems Technology, pp. 1105–1118. ISSN: 1558-0865. DOI: 10.1109/TCST.2008.2012116.

Laine, Forrest and Claire Tomlin (Dec. 2019). "Parallelizing LQR Computation Through Endpoint-Explicit Riccati Recursion". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. ISSN: 2576-2370, pp. 1395–1402. DOI: 10.1109/CDC40024.2019.9029974.

Lam, Donald (Nov. 2013). *Maintaining Atlas*. en-US. URL: https://www.engineering.hku.hk/tecHKU/2013/11/11/maintaining-atlas/.

Lantoine, Gregory and Ryan P. Russell (Aug. 2012). "A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 1: Theory". en. In: *J Optim Theory Appl* 154.2, pp. 382–417. ISSN: 1573-2878. DOI: 10.1007/s10957-012-0039-0. URL: https://doi.org/10.1007/s10957-012-0039-0.

Laumond, Jean-Paul (Jan. 2013). *La robotique : une récidive d'Héphaïstos : Leçon inaugurale prononcée le jeudi 19 janvier 2012*. fr. Leçons inaugurales. Paris: Collège de France. ISBN: 978-2-7226-0169-7. URL: http://books.openedition.org/cdf/498.

LaValle, Steven M. (1998). *Rapidly-exploring random trees : a new tool for path planning*. Tech. rep. Iowa State University.

Le Lidec, Quentin, Igor Kalevatykh, Ivan Laptev, Cordelia Schmid, and Justin Carpentier (Apr. 2021). "Differentiable Simulation for Physical System Identification". In: *IEEE Robotics and Automation Letters* 6.2. Conference Name: IEEE Robotics and Automation Letters, pp. 3413–3420. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3062323.

Lee, Jinoh, Houman Dallali, Maloin Jin, Darwin Caldwell, and Nikolaos Tsagarakis (May 2016). "Robust and adaptive whole-body controller for humanoids with multiple tasks under uncertain disturbances". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5683–5689. DOI: 10.1109/ICRA.2016.7487790.

Lembono, Teguh Santoso, Carlos Mastalli, Pierre Fernbach, Nicolas Mansard, and Sylvain Calinon (2020a). "Learning How to Walk: Warm-starting Optimal Control Solver with Memory of Motion". In: arXiv:2001.11751 [cs, math]. DOI: 10.48550/arXiv.2001.11751. URL: http://arxiv.org/abs/2001.11751.

Lembono, Teguh Santoso, Antonio Paolillo, Emmanuel Pignat, and Sylvain Calinon (Apr. 2020b). "Memory of Motion for Warm-Starting Trajectory Optimization". In: *IEEE Robot. Autom. Lett.* 5.2, pp. 2594–2601. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2020.2972893. URL: https://ieeexplore.ieee.org/document/8990042/.

Lenz, Ian, Ross Knepper, and Ashutosh Saxena (July 2015). "DeepMPC: Learning Deep Latent Features for Model Predictive Control". en. In: *Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation. ISBN: 978-0-9923747-1-6. DOI: 10.15607/RSS.2015.XI.012. URL: http://www.roboticsproceedings.org/rss11/p12.pdf.

Levine, Sergey and Vladlen Koltun (May 2013). "Guided Policy Search". en. In: *Proceedings of the 30th International Conference on Machine Learning*. ISSN: 1938-7228. PMLR, pp. 1–9. URL: https://proceedings.mlr.press/v28/levine13.html.

Li, He, Robert J. Frei, and Patrick M. Wensing (Apr. 2021). "Model Hierarchy Predictive Control of Robotic Systems". In: *IEEE Robotics and Automation Letters* 6.2. Conference Name: IEEE Robotics and Automation Letters, pp. 3373–3380. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3061322.

Liu, Chenggang, Christopher Atkeson, and Jianbo Su (Mar. 2013). "Biped walking control using a trajectory library". In: *Robotica* 31. DOI: 10.1017/S0263574712000203.

Lowrey, Kendall, Aravind Rajeswaran, Sham M. Kakade, Emanuel Todorov, and Igor Mordatch (2018). "Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control". In: *CoRR* abs/1811.01848. arXiv: 1811.01848. URL: http://arxiv.org/abs/1811.01848.

Lozano-Perez (Feb. 1983). "Spatial Planning: A Configuration Space Approach". In: *IEEE Transactions on Computers* C-32.2. Conference Name: IEEE Transactions on Computers, pp. 108–120. ISSN: 1557-9956. DOI: 10.1109/TC.1983.1676196.

Ma, Wen-Loong, Kaveh Akbari Hamed, and Aaron D. Ames (Sept. 2019). "First Steps Towards Full Model Based Motion Planning and Control of Quadrupeds: A Hybrid Zero Dynamics Approach". In: arXiv:1909.08124 [cs, eess]. IEEE. DOI: 10.48550/arXiv.1909.08124. URL: http://arxiv.org/abs/1909.08124.

Majumdar, Anirudha, Georgina Hall, and Amir Ahmadi (May 2020). "Recent Scalability Improvements for Semidefinite Programming with Applications in Machine Learning, Control, and Robotics". In: *Annual Review of Control, Robotics, and Autonomous Systems* 3. DOI: 10.1146/annurev-control-091819-074326.

Manchester, Zac, Neel Doshi, Robert Wood, and Scott Kuindersma (May 2019). "Contact-implicit trajectory optimization using variational integrators". In: *The International Journal of Robotics Research* 38, p. 027836491984923. DOI: 10.1177/0278364919849235.

Mansard, N. (May 2012). "A dedicated solver for fast operational-space inverse dynamics". In: *2012 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 4943–4949. DOI: 10.1109/ICRA.2012.6224851.

Mansard, N., A. DelPrete, M. Geisert, S. Tonneau, and O. Stasse (May 2018). "Using a Memory of Motion to Efficiently Warm-Start a Nonlinear Predictive Controller". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 2986–2993. DOI: 10.1109/ICRA.2018.8463154.

Mansard, Nicolas and Franois Chaumette (Feb. 2007). "Task Sequencing for High-Level Sensor-Based Control". In: *IEEE Transactions on Robotics* 23.1. Conference

Name: IEEE Transactions on Robotics, pp. 60–72. ISSN: 1941-0468. DOI: 10.1109/TRO.2006.889487.

Marhefka, D.W. and D.E. Orin (Nov. 1999). "A compliant contact model with nonlinear damping for simulation of robotic systems". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 29.6. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, pp. 566–572. ISSN: 1558-2426. DOI: 10.1109/3468.798060.

Mason, Sean, Rotella Rotella, Stefan Schaal, and Ludovic Righetti (2016). "Balancing and walking using full dynamics LQR control with contact constraints". In: IEEE. DOI: 10.1109/humanoids.2016.7803255.

Mastalli, Carlos, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Sethu Vijayakumar, and Nicolas Mansard (May 2020). "Crocoddyl: An Efficient and Versatile Framework for Multi-Contact Optimal Control". In: *ICRA 2020 IEEE International Conference on Robotics and Automation*. Paris / Virtual, France. DOI: 10.1109/ICRA40945.2020.9196673. URL: https://hal.archives-ouvertes.fr/hal-02294059.

Mastalli, Carlos, Wolfgang Merkt, Guiyang Xin, Jaehyun Shim, Michael Mistry, Ioannis Havoutis, and Sethu Vijayakumar (July 2022a). *Agile Maneuvers in Legged Robots: a Predictive Control Approach*. arXiv:2203.07554 [cs, eess]. DOI: 10.48550/arXiv.2203.07554. URL: http://arxiv.org/abs/2203.07554.

Mastalli, Carlos, Saroj Prasad Chhatoi, Thomas Corbères, Steve Tonneau, and Sethu Vijayakumar (Sept. 2022b). *Inverse-Dynamics MPC via Nullspace Resolution*. arXiv:2209.05375 [cs, eess]. DOI: 10.48550/arXiv.2209.05375. URL: http://arxiv.org/abs/2209.05375.

Matsuoka, Kiyotoshi (Feb. 1987). "Mechanisms of frequency and pattern control in the neural rhythm generators". In: *Biological cybernetics* 56, pp. 345–53. DOI: 10.1007/BF00319514.

Matthis, Jonathan Samir, Jacob L. Yates, and Mary M. Hayhoe (Apr. 2018). "Gaze and the Control of Foot Placement When Walking in Natural Terrain". eng. In: *Curr Biol* 28.8, 1224–1233.e5. ISSN: 1879-0445. DOI: 10.1016/j.cub.2018.03.008.

Mayne, DAVID Q. (Jan. 1973). "Differential Dynamic Programming–A Unified Approach to the Optimization of Dynamic Systems". en. In: *Control and Dynamic Systems*. Ed. by C. T. Leondes. Vol. 10. Academic Press, pp. 179–254. DOI: 10.1016/B978-0-12-012710-8.50010-8. URL: https://www.sciencedirect.com/science/article/pii/B9780120127108500108.

McGeer, Tad (1990). "Passive Dynamic Walking". In: *The International Journal of Robotics Research* 9.2, pp. 62–82.

Meduri, Avadesh, Paarth Shah, Julian Viereck, Majid Khadiv, Ioannis Havoutis, and Ludovic Righetti (Jan. 2022). *BiConMP: A Nonlinear Model Predictive Control Framework for Whole Body Motion Planning*. Tech. rep. Publication Title: arXiv e-prints ADS Bibcode: 2022arXiv220107601M Type: article. URL: https://ui.adsabs.harvard.edu/abs/2022arXiv220107601M.

Melon, Oliwier, Romeo Orsolino, David Surovik, Mathieu Geisert, Ioannis Havoutis, and Maurice Fallon (May 2021). "Receding-Horizon Perceptive Trajectory Optimization for Dynamic Legged Locomotion with Learned Initialization". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 2577-087X, pp. 9805–9811. DOI: 10.1109/ICRA48506.2021.9560794.

Messina, Elena (Jan. 2006). "Performance standards for urban search and rescue robots". In: 34.

Miki, Takahiro, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter (Jan. 2022). "Learning robust perceptive locomotion for quadrupedal

robots in the wild". In: *Science Robotics* 7.62. Publisher: American Association for the Advancement of Science, eabk2822. DOI: 10.1126/scirobotics.abk2822. URL: https://www.science.org/doi/10.1126/scirobotics.abk2822.

Montaut, Louis, Quentin Le Lidec, Vladimir Petrik, Josef Sivic, and Justin Carpentier (June 2022). "Collision Detection Accelerated: An Optimization Perspective". en. In: *Robotics: Science and Systems XVIII*. Robotics: Science and Systems Foundation. ISBN: 978-0-9923747-8-5. DOI: 10.15607/RSS.2022.XVIII.039. URL: http://www.roboticsproceedings.org/rss18/p039.pdf.

Mordatch, Igor and Emo Todorov (2014). "Combining the benefits of function approximation and trajectory optimization". In: vol. 4. DOI: 10.15607/RSS.2014.X.052.

Mordatch, Igor, Martin de Lasa, and Aaron Hertzmann (July 2010). "Robust Physics-Based Locomotion Using Low-Dimensional Planning". In: *ACM Trans. Graph.* 29. DOI: 10.1145/1833349.1778808.

Mordatch, Igor, Emanuel Todorov, and Zoran Popović (July 2012). "Discovery of Complex Behaviors through Contact-Invariant Optimization". In: *ACM Transactions on Graphics - TOG* 31. DOI: 10.1145/2185520.2185539.

Mordatch, Igor, Kendall Lowrey, Galen Andrew, Zoran Popovic, and Emanuel V. Todorov (2015). "Interactive Control of Diverse Complex Characters with Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/hash/2612aa892d962d6f8056b195ca6e550d-Abstract.html.

Naderi, Kourosh, Joose Rajamäki, and Perttu Hämäläinen (Nov. 2015). "RT-RRT*: a real-time path planning algorithm based on RRT*". In: pp. 113–118. DOI: 10.1145/2822013.2822036.

Nagasaka, K., Y. Kuroki, S. Suzuki, Y. Itoh, and J. Yamaguchi (Apr. 2004). "Integrated motion control for walking, jumping and running on a small bipedal entertainment robot". In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. Vol. 4. ISSN: 1050-4729, 3189–3194 Vol.4. DOI: 10.1109/ROBOT.2004.1308745.

Naveau, M., M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères (Jan. 2017). "A Reactive Walking Pattern Generator Based on Nonlinear Model Predictive Control". In: *IEEE Robotics and Automation Letters* 2.1. Conference Name: IEEE Robotics and Automation Letters, pp. 10–17. ISSN: 2377-3766. DOI: 10.1109/LRA.2016.2518739.

Neunert, Michael, Cédric de Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli (May 2016). "Fast nonlinear Model Predictive Control for unified trajectory optimization and tracking". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1398–1404. DOI: 10.1109/ICRA.2016.7487274.

Neunert, Michael, Farbod Farshidian, Alexander Winkler, and Jonas Buchli (2017). "Trajectory Optimization Through Contacts and Automatic Gait Discovery for Quadrupeds". In: *IEEE Robotics and Automation Letters* 2.3, pp. 1502–1509. DOI: 10.1109/LRA.2017.2665685.

Neunert, Michael, Markus Stäuble, Markus Giftthaler, Carmine D. Bellicoso, Jan Carius, Christian Gehring, Marco Hutter, and Jonas Buchli (July 2018). "Whole-Body Nonlinear Model Predictive Control Through Contacts for Quadrupeds". In: *IEEE Robot. Autom. Lett.* 3.3. arXiv:1712.02889 [cs], pp. 1458–1465. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2018.2800124. URL: http://arxiv.org/abs/1712.02889.

Nguyen, Quan, Xingye Da, J. W. Grizzle, and Koushil Sreenath (2020). "Dynamic Walking on Stepping Stones with Gait Library and Control Barrier Functions". en. In: *Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics*. Ed. by Ken Goldberg, Pieter Abbeel, Kostas Bekris, and Lauren Miller. Springer Proceedings in Advanced Robotics. Cham: Springer International Publishing, pp. 384–399. ISBN: 978-3-030-43089-4. DOI: 10.1007/978-3-030-43089-4_25. URL: https://doi.org/10.1007/978-3-030-43089-4_25.

Nishiwaki, K. and S. Kagami (May 2006). "High frequency walking pattern generation based on preview control of ZMP". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* ISSN: 1050-4729, pp. 2667–2672. DOI: 10.1109/ROBOT.2006.1642104.

"Penalty and Augmented Lagrangian Methods" (2006a). en. In: *Numerical Optimization*. Ed. by Jorge Nocedal and Stephen J. Wright. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, pp. 497–528. ISBN: 978-0-387-40065-5. DOI: 10.1007/978-0-387-40065-5_17. URL: https://doi.org/10.1007/978-0-387-40065-5_17.

"Sequential Quadratic Programming" (2006b). en. In: *Numerical Optimization*. Ed. by Jorge Nocedal and Stephen J. Wright. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, pp. 529–562. ISBN: 978-0-387-40065-5. DOI: 10.1007/978-0-387-40065-5_18. URL: https://doi.org/10.1007/978-0-387-40065-5_18.

Orin, David E., Ambarish Goswami, and Sung-Hee Lee (Oct. 2013). "Centroidal dynamics of a humanoid robot". en. In: *Auton Robot* 35.2, pp. 161–176. ISSN: 1573-7527. DOI: 10.1007/s10514-013-9341-4. URL: https://doi.org/10.1007/s10514-013-9341-4.

Orthey, Andreas and Olivier Stasse (Oct. 2013). "Towards reactive whole-body motion planning in cluttered environments by precomputing feasible motion spaces". In: *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 274–279. DOI: 10.1109/HUMANOIDS.2013.7029987.

Ott, Christian, Maximo A. Roa, and Gerd Hirzinger (Oct. 2011). "Posture and balance control for biped robots based on contact force optimization". In: *2011 11th IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 26–33. DOI: 10.1109/Humanoids.2011.6100882.

Pan, Jia, Sachin Chitta, and Dinesh Manocha (May 2012). "FCL: A general purpose library for collision and proximity queries". In: *2012 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 3859–3866. DOI: 10.1109/ICRA.2012.6225337.

Pandala, Abhishek, Randall T. Fawcett, Ugo Rosolia, Aaron D. Ames, and Kaveh Akbari Hamed (July 2022). "Robust Predictive Control for Quadrupedal Locomotion: Learning to Close the Gap Between Reduced- and Full-Order Models". In: *IEEE Robotics and Automation Letters* 7.3. Conference Name: IEEE Robotics and Automation Letters, pp. 6622–6629. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3176105.

Papadopoulos, Evangelos G. (1993). "Nonholonomic Behavior in Free-floating Space Manipulators and its Utilization". In: ed. by Zexiang Li and J. F. Canny. Book Title: Nonholonomic Motion Planning. Boston, MA: Springer US, pp. 423–445. ISBN: 978-1-4613-6392-7 978-1-4615-3176-0. DOI: 10.1007/978-1-4615-3176-0_11. URL: http://link.springer.com/10.1007/978-1-4615-3176-0_11.

Pardo, Diego, Lukas Moller, Michael Neunert, Alexander W. Winkler, and Jonas Buchli (July 2016). "Evaluating Direct Transcription and Nonlinear Optimization

Methods for Robot Motion Planning". In: *IEEE Robot. Autom. Lett.* 1.2, pp. 946–953. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2016.2527062. URL: http://ieeexplore.ieee.org/document/7400956/.

Park, Jong Hyeon and Hyun Chul Cho (Apr. 2000). "An online trajectory modifier for the base link of biped robots to enhance locomotion stability". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 4. ISSN: 1050-4729, 3353–3358 vol.4. DOI: 10.1109/ROBOT.2000.845229.

Park, Juyong, Jaeyoung Haan, and F. C. Park (Aug. 2007). "Convex Optimization Algorithms for Active Balancing of Humanoid Robots". In: *IEEE Transactions on Robotics* 23.4. Conference Name: IEEE Transactions on Robotics, pp. 817–822. ISSN: 1941-0468. DOI: 10.1109/TRO.2007.900639.

Perez, Alejandro, Robert Platt, George Konidaris, Leslie Kaelbling, and Tomas Lozano-Perez (May 2012). "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics". In: *2012 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 2537–2542. DOI: 10.1109/ICRA.2012.6225177.

Perrin, Nicolas, Olivier Stasse, Léo Baudouin, Florent Lamiraux, and Eiichi Yoshida (Apr. 2012). "Fast Humanoid Robot Collision-Free Footstep Planning Using Swept Volume Approximations". In: *IEEE Transactions on Robotics* 28.2. Conference Name: IEEE Transactions on Robotics, pp. 427–439. ISSN: 1941-0468. DOI: 10.1109/TRO.2011.2172152.

Perrin, Nicolas, Darwin Lau, and Vincent Padois (2018). "Effective Generation of Dynamically Balanced Locomotion with Multiple Non-coplanar Contacts". en. In: *Robotics Research: Volume 2*. Ed. by Antonio Bicchi and Wolfram Burgard. Springer Proceedings in Advanced Robotics. Cham: Springer International Publishing, pp. 201–216. ISBN: 978-3-319-60916-4. DOI: 10.1007/978-3-319-60916-4_12. URL: https://doi.org/10.1007/978-3-319-60916-4_12.

Pfrommer, Samuel, Mathew Halm, and Michael Posa (2020). "ContactNets: Learning Discontinuous Contact Dynamics with Smooth, Implicit Representations". In: arXiv:2009.11193 [cs]. DOI: 10.48550/arXiv.2009.11193. URL: http://arxiv.org/abs/2009.11193.

Plamondon, Réjean and Wacef Guerfali (Nov. 1998). "The 2/3 power law: When and why?" en. In: *Acta Psychologica* 100.1, pp. 85–96. ISSN: 0001-6918. DOI: 10.1016/S0001-6918(98)00027-4. URL: https://www.sciencedirect.com/science/article/pii/S0001691898000274.

Ponton, Brahayam, Alexander Herzog, Stefan Schaal, and Ludovic Righetti (Nov. 2016). "A convex model of humanoid momentum dynamics for multi-contact motion generation". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. ISSN: 2164-0580, pp. 842–849. DOI: 10.1109/HUMANOIDS.2016.7803371.

Posa, Michael, Cecilia Cantu, and Russ Tedrake (Oct. 2014). "A direct method for trajectory optimization of rigid bodies through contact". In: *International Journal of Robotics Research* 33, pp. 69–81. DOI: 10.1177/0278364913506757.

Posa, Michael, Scott Kuindersma, and Russ Tedrake (May 2016). "Optimization and stabilization of trajectories for constrained dynamical systems". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1366–1373. DOI: 10.1109/ICRA.2016.7487270.

Pratt, Jerry, John Carff, Sergey Drakunov, and Ambarish Goswami (Dec. 2006). "Capture Point: A Step toward Humanoid Push Recovery". In: *2006 6th IEEE-RAS*

International Conference on Humanoid Robots. ISSN: 2164-0580, pp. 200–207. DOI: 10.1109/ICHR.2006.321385.

Quirynen, R., M. Vukov, M. Zanon, and M. Diehl (2015). "Autogenerating microsecond solvers for nonlinear MPC: A tutorial using ACADO integrators". en. In: *Optimal Control Applications and Methods* 36.5. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.10 pp. 685–704. ISSN: 1099-1514. DOI: 10.1002/oca.2152. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/oca.2152.

Ramos, Oscar E., Nicolas Mansard, Olivier Stasse, and Philippe Souères (Nov. 2012). "Walking on non-planar surfaces using an inverse dynamic stack of tasks". In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. ISSN: 2164-0580, pp. 829–834. DOI: 10.1109/HUMANOIDS.2012.6651616.

Ramos, Oscar E., Nicolas Mansard, and Philippe Souères (Nov. 2014). "Whole-body motion integrating the capture point in the operational space inverse dynamics control". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 707–712. DOI: 10.1109/HUMANOIDS.2014.7041440.

Rathod, Niraj, Angelo Bratta, Michele Focchi, Mario Zanon, Octavio Villarreal, Claudio Semini, and Alberto Bemporad (2021). "Model Predictive Control with Environment Adaptation for Legged Locomotion". In: *IEEE Access* 9. arXiv:2105.05998 [cs, eess], pp. 145710–145727. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3118957. URL: http://arxiv.org/abs/2105.05998.

Rawlings, J., D.Q. Mayne, and Moritz Diehl (Jan. 2017). *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing.

Rezazadeh, Siavash and Jonathan W Hurst (June 2020). "Control of ATRIAS in three dimensions: Walking as a forced-oscillation problem". en. In: *The International Journal of Robotics Research* 39.7. Publisher: SAGE Publications Ltd STM, pp. 774–796. ISSN: 0278-3649. DOI: 10.1177/0278364920916777. URL: https://doi.org/10.1177/0278364920916777.

Righetti, Ludovic, Jonas Buchli, Michael Mistry, and Stefan Schaal (Oct. 2011). "Control of legged robots with optimal distribution of contact forces". In: *2011 11th IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 318–324. DOI: 10.1109/Humanoids.2011.6100832.

Roussel, L., C. Canudas-De-Wit, and A. Goswami (May 1998). "Generation of energy optimal complete gait cycles for biped robots". In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. Vol. 3. ISSN: 1050-4729, pp. 2036–2041. DOI: 10.1109/ROBOT.1998.680615.

Saab, Layale, Oscar E. Ramos, François Keith, Nicolas Mansard, Philippe Souères, and Jean-Yves Fourquet (Apr. 2013). "Dynamic Whole-Body Motion Generation Under Rigid Contacts and Other Unilateral Constraints". In: *IEEE Transactions on Robotics* 29.2. Conference Name: IEEE Transactions on Robotics, pp. 346–362. ISSN: 1941-0468. DOI: 10.1109/TRO.2012.2234351.

Saibene, F. (1990). "The mechanisms for minimizing energy expenditure in human locomotion". eng. In: *Eur J Clin Nutr* 44 Suppl 1, pp. 65–71. ISSN: 0954-3007.

Salini, Joseph, Vincent Padois, and Philippe Bidaud (May 2011). "Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions". In: *2011 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 1283–1290. DOI: 10.1109/ICRA.2011.5980202.

Sano, A. and J. Furusho (May 1990). "Realization of natural dynamic walking using the angular momentum information". In: , *IEEE International Conference on Robotics and Automation Proceedings*, 1476–1481 vol.3. DOI: 10.1109/ROBOT.1990.126214.

Sardain, P. and G. Bessonnet (Sept. 2004). "Forces acting on a biped robot. Center of pressure-zero moment point". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 34.5. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, pp. 630–637. ISSN: 1558-2426. DOI: 10.1109/TSMCA.2004.832811.

Schultz, Gerrit and Katja Mombaur (Oct. 2010). "Modeling and Optimal Control of Human-Like Running". In: *IEEE/ASME Transactions on Mechatronics* 15.5. Conference Name: IEEE/ASME Transactions on Mechatronics, pp. 783–792. ISSN: 1941-014X. DOI: 10.1109/TMECH.2009.2035112.

Schwartz, Jacob T and Micha Sharir (Sept. 1983). "On the "piano movers" problem. II. General techniques for computing topological properties of real algebraic manifolds". en. In: *Advances in Applied Mathematics* 4.3, pp. 298–351. ISSN: 0196-8858. DOI: 10.1016/0196-8858(83)90014-3. URL: https://www.sciencedirect.com/science/article/pii/0196885883900143.

Selen, Luc P. J., David W. Franklin, and Daniel M. Wolpert (Oct. 2009). "Impedance Control Reduces Instability That Arises from Motor Noise". In: *J Neurosci* 29.40, pp. 12606–12616. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.2826-09.2009. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2784227/.

Sentis, L. and O. Khatib (Apr. 2005). "Control of Free-Floating Humanoid Robots Through Task Prioritization". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 1718–1723. DOI: 10.1109/ROBOT.2005.1570361.

Shapiro, Alexander (May 1988). "Sensitivity Analysis of Nonlinear Programs and Differentiability Properties of Metric Projections". In: *SIAM J. Control Optim.* 26.3. Publisher: Society for Industrial and Applied Mathematics, pp. 628–645. ISSN: 0363-0129. DOI: 10.1137/0326037. URL: https://epubs.siam.org/doi/10.1137/0326037.

Sideris, A. and J.E. Bobrow (June 2005). "An efficient sequential linear quadratic algorithm for solving nonlinear optimal control problems". In: *Proceedings of the 2005, American Control Conference, 2005.* ISSN: 2378-5861, 2275–2280 vol. 4. DOI: 10.1109/ACC.2005.1470308.

Siekmann, Jonah, Yesh Godse, Alan Fern, and Jonathan Hurst (May 2021). "Sim-to-Real Learning of All Common Bipedal Gaits via Periodic Reward Composition". In: pp. 7309–7315. DOI: 10.1109/ICRA48506.2021.9561814.

Singh, Rohan Pratap, Mehdi Benallegue, Mitsuharu Morisawa, Rafael Cisneros, and Fumio Kanehiro (Oct. 2022). *Learning Bipedal Walking On Planned Footsteps For Humanoid Robots*. arXiv:2207.12644 [cs]. DOI: 10.48550/arXiv.2207.12644. URL: http://arxiv.org/abs/2207.12644.

Smyth, Mary M. and Alan W. Wing (Oct. 2013). *Psychology of Human Movement*. en. Google-Books-ID: 42daBQAAQBAJ. Elsevier. ISBN: 978-1-4832-8915-1.

Sreenath, Koushil, Hae-Won Park, Ioannis Poulakakis, and J W Grizzle (Aug. 2011). "A Compliant Hybrid Zero Dynamics Controller for Stable, Efficient and Fast Bipedal Walking on MABEL". en. In: *The International Journal of Robotics Research* 30.9. Publisher: SAGE Publications Ltd STM, pp. 1170–1193. ISSN: 0278-3649. DOI: 10.1177/0278364910379882. URL: https://doi.org/10.1177/0278364910379882.

Stasse, O., T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, J. Mirabel, A. Del Prete, P. Souères, N. Mansard, F. Lamiraux, J.-P. Laumond, L. Marchionni, H. Tome, and F. Ferro (Nov. 2017). "TALOS: A new humanoid research platform targeted for industrial applications". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. ISSN: 2164-0580, pp. 689–695. DOI: 10.1109/HUMANOIDS.2017.8246947.

Stolle, M. and C.G. Atkeson (May 2006). "Policies based on trajectory libraries". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* ISSN: 1050-4729, pp. 3344–3349. DOI: 10.1109/ROBOT.2006.1642212.

Sugihara, T., Y. Nakamura, and H. Inoue (May 2002). "Real-time humanoid motion generation through ZMP manipulation based on inverted pendulum control". In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*. Vol. 2, 1404–1409 vol.2. DOI: 10.1109/ROBOT.2002.1014740.

Suh, H. J. Terry, Tao Pang, and Russ Tedrake (Jan. 2022). "Bundled Gradients through Contact via Randomized Smoothing". In: *IEEE Robotics and Automation Letters*. arXiv:2109.05143 [cs]. DOI: 10.48550/arXiv.2109.05143. URL: http://arxiv.org/abs/2109.05143.

Takanishi, A., M. Tochizawa, T. Takeya, H. Karaki, and I. Kato (1989). "Realization of Dynamic Biped Walking Stabilized with Trunk Motion Under Known External Force". en. In: *Advanced Robotics: 1989*. Ed. by Kenneth J. Waldron. Berlin, Heidelberg: Springer, pp. 299–310. ISBN: 978-3-642-83957-3. DOI: 10.1007/978-3-642-83957-3_21.

Tamar, Aviv, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel (2016). "Value Iteration Networks". In: arXiv:1602.02867 [cs, stat], pp. 2154–2162. DOI: 10.48550/arXiv.1602.02867. URL: http://arxiv.org/abs/1602.02867.

Tamar, Aviv, Garrett Thomas, Tianhao Zhang, Sergey Levine, and Pieter Abbeel (Mar. 2017). "Learning from the Hindsight Plan – Episodic MPC Improvement". In: arXiv:1609.09001 [cs]. arXiv. DOI: 10.48550/arXiv.1609.09001. URL: http://arxiv.org/abs/1609.09001.

Tassa, Yuval, Tom Erez, and Emanuel Todorov (Oct. 2012). "Synthesis and stabilization of complex behaviors through online trajectory optimization". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 4906–4913. DOI: 10.1109/IROS.2012.6386025.

Tassa, Yuval, Nicolas Mansard, and Emo Todorov (May 2014). "Control-limited differential dynamic programming". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729, pp. 1168–1175. DOI: 10.1109/ICRA.2014.6907001.

Todorov, E. and Weiwei Li (June 2005). "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems". In: *Proceedings of the 2005, American Control Conference, 2005.* ISSN: 2378-5861, 300–306 vol. 1. DOI: 10.1109/ACC.2005.1469949.

Todorov, Emanuel (May 2014). "Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in MuJoCo". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729, pp. 6054–6061. DOI: 10.1109/ICRA.2014.6907751.

Todorov, Emanuel and Michael I. Jordan (Nov. 2002). "Optimal feedback control as a theory of motor coordination". eng. In: *Nat Neurosci* 5.11, pp. 1226–1235. ISSN: 1097-6256. DOI: 10.1038/nn963.

Todorov, Emanuel, Tom Erez, and Yuval Tassa (Oct. 2012). "MuJoCo: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

Tonneau, Steve, Andrea Del Prete, Julien Pettré, Chonhyon Park, Dinesh Manocha, and Nicolas Mansard (June 2018). "An Efficient Acyclic Contact Planner for Multiped Robots". In: *IEEE Transactions on Robotics* 34.3. Conference Name: IEEE Transactions on Robotics, pp. 586–601. ISSN: 1941-0468. DOI: 10.1109/TRO.2018.2819658.

Tonneau, Steve, Daeun Song, Pierre Fernbach, Nicolas Mansard, Michel Taix, and Andrea Del Prete (Sept. 2019). *SL1M: Sparse L1-norm Minimization for contact planning on uneven terrain*. arXiv:1909.09044 [cs]. DOI: 10.48550/arXiv.1909.09044. URL: http://arxiv.org/abs/1909.09044.

Tsounis, Vassilios, Mitja Alge, Joonho Lee, Farbod Farshidian, and Marco Hutter (Mar. 2020). "DeepGait: Planning and Control of Quadrupedal Gaits Using Deep Reinforcement Learning". In: *IEEE Robotics and Automation Letters* PP, pp. 1–1. DOI: 10.1109/LRA.2020.2979660.

Vaillant, Joris, Abderrahmane Kheddar, Hervé Audren, François Keith, Stanislas Brossette, Kenji Kaneko, Mitsuharu Morisawa, Eiichi Yoshida, and Fumio Kanehiro (Nov. 2014). "Vertical ladder climbing by the HRP-2 humanoid robot". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 671–676. DOI: 10.1109/HUMANOIDS.2014.7041435.

Verschueren, Robin, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl (Nov. 2020). *acados: a modular open-source framework for fast embedded optimal control*. arXiv:1910.13753 [math]. DOI: 10.48550/arXiv.1910.13753. URL: http://arxiv.org/abs/1910.13753.

Villa, Nahuel Alejandro, Johannes Englsberger, and Pierre-Brice Wieber (Oct. 2019). "Sensitivity of Legged Balance Control to Uncertainties and Sampling Period". In: *IEEE Robotics and Automation Letters* 4.4. Conference Name: IEEE Robotics and Automation Letters, pp. 3665–3670. ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2927944.

Villarreal Magaña, Octavio, Victor Barasuol, Patrick Wensing, Darwin Caldwell, and Claudio Semini (May 2020). "MPC-based Controller with Terrain Insight for Dynamic Legged Locomotion". In: pp. 2436–2442. DOI: 10.1109/ICRA40945.2020.9197312.

Vukobratovic, Miomir and Davor Juricic (Jan. 1969). "Contribution to the Synthesis of Biped Gait". In: *IEEE Transactions on Biomedical Engineering* BME-16.1. Conference Name: IEEE Transactions on Biomedical Engineering, pp. 1–6. ISSN: 1558-2531. DOI: 10.1109/TBME.1969.4502596.

Vukov, Milan, Alexander Domahidi, Hans Joachim Ferreau, Manfred Morari, and Moritz Diehl (Dec. 2013). "Auto-generated algorithms for nonlinear model predictive control on long and on short horizons". In: *52nd IEEE Conference on Decision and Control*. ISSN: 0191-2216, pp. 5113–5118. DOI: 10.1109/CDC.2013.6760692.

Wall, James and Jacques Marescaux (Oct. 2013). "History of Telesurgery". In: pp. 15–18. ISBN: 978-2-8178-0390-6. DOI: 10.1007/978-2-8178-0391-3_2.

Wampler, Kevin and Zoran Popovic (July 2009). "Optimal Gait and Form for Animal Locomotion". In: *ACM Trans. Graph.* 28. DOI: 10.1145/1576246.1531366.

Webb, Dustin J. and Jur van den Berg (May 2013). "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics". In: *2013 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729, pp. 5054–5061. DOI: 10.1109/ICRA.2013.6631299.

Wensing, Patrick M. and David E. Orin (Nov. 2013). "High-speed humanoid running through control with a 3D-SLIP model". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISSN: 2153-0866, pp. 5134–5140. DOI: 10.1109/IROS.2013.6697099.

Wensing, Patrick M., Michael Posa, Yue Hu, Adrien Escande, Nicolas Mansard, and Andrea Del Prete (Nov. 2022). *Optimization-Based Control for Dynamic Legged Robots*.

arXiv:2211.11644 [cs]. DOI: 10.48550/arXiv.2211.11644. URL: http://arxiv.org/abs/2211.11644.

Westervelt, E. R., G. Buche, and J. W. Grizzle (June 2004). "Experimental Validation of a Framework for the Design of Controllers that Induce Stable Walking in Planar Bipeds". en. In: *The International Journal of Robotics Research* 23.6. Publisher: SAGE Publications Ltd STM, pp. 559–582. ISSN: 0278-3649. DOI: 10.1177/0278364904044410. URL: https://doi.org/10.1177/0278364904044410.

Westervelt, E.R., J.W. Grizzle, and D.E. Koditschek (Jan. 2003). "Hybrid zero dynamics of planar biped walkers". In: *IEEE Transactions on Automatic Control* 48.1. Conference Name: IEEE Transactions on Automatic Control, pp. 42–56. ISSN: 1558-2523. DOI: 10.1109/TAC.2002.806653.

Wieber, P.-B. (2006a). "Holonomy and Nonholonomy in the Dynamics of Articulated Motion". en. In: *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*. Ed. by Moritz Diehl and Katja Mombaur. Lecture Notes in Control and Information Sciences. Berlin, Heidelberg: Springer, pp. 411–425. ISBN: 978-3-540-36119-0. DOI: 10.1007/978-3-540-36119-0_20. URL: https://doi.org/10.1007/978-3-540-36119-0_20.

Wieber, Pierre-brice (Dec. 2006b). "Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations". In: *2006 6th IEEE-RAS International Conference on Humanoid Robots*. ISSN: 2164-0580, pp. 137–142. DOI: 10.1109/ICHR.2006.321375.

Wieber, Pierre-Brice and Christine Chevallereau (July 2006). "Online adaptation of reference trajectories for the control of walking systems". en. In: *Robotics and Autonomous Systems* 54.7, pp. 559–566. ISSN: 0921-8890. DOI: 10.1016/j.robot.2006.04.007. URL: https://www.sciencedirect.com/science/article/pii/S092188900600056X.

Winkler, Alexander W., C. Dario Bellicoso, Marco Hutter, and Jonas Buchli (July 2018). "Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization". In: *IEEE Robotics and Automation Letters* 3.3. Conference Name: IEEE Robotics and Automation Letters, pp. 1560–1567. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2798285.

Wisse, Martijn, Guillaume Feliksdal, Jan Van Frankenhuyzen, and Brian Moyer (June 2007). "Passive-based walking robot - Denise, a simple, efficient, and lightweight biped". In: *IEEE Robotics & Automation Magazine* 14.2. Conference Name: IEEE Robotics & Automation Magazine, pp. 52–62. ISSN: 1558-223X. DOI: 10.1109/MRA.2007.380639.

Wu, Xiaodong and Shugen Ma (Apr. 2010). "Adaptive creeping locomotion of a CPG-controlled snake-like robot to environment change". en. In: *Auton Robot* 28.3, pp. 283–294. ISSN: 1573-7527. DOI: 10.1007/s10514-009-9168-1. URL: https://doi.org/10.1007/s10514-009-9168-1.

Wächter, Andreas and Lorenz T. Biegler (Mar. 2006). "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". en. In: *Math. Program.* 106.1, pp. 25–57. ISSN: 1436-4646. DOI: 10.1007/s10107-004-0559-y. URL: https://doi.org/10.1007/s10107-004-0559-y.

Xiong, Xiaobin and Aaron D. Ames (Nov. 2019). "Motion Decoupling and Composition via Reduced Order Model optimization for Dynamic Humanoid Walking with CLF-QP based Active Force Control". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866, pp. 1018–1024. DOI: 10.1109/IROS40897.2019.8968215.

Yeganegi, Mohammad Hasan, Majid Khadiv, Andrea Del Prete, S. Ali A. Moosavian, and Ludovic Righetti (Aug. 2022). "Robust Walking Based on MPC With Viability Guarantees". In: *IEEE Transactions on Robotics* 38.4. Conference Name: IEEE Transactions on Robotics, pp. 2389–2404. ISSN: 1941-0468. DOI: 10.1109/TRO.2021.3127388.

Yokoyama, Hiroshi, Hiashi Saito, Rie Kurai, Isao Nambu, and Yasuhiro Wada (Oct. 2018). "Investigation of isochrony phenomenon based on the computational theory of human arm trajectory planning". en. In: *Human Movement Science* 61, pp. 52–62. ISSN: 0167-9457. DOI: 10.1016/j.humov.2018.07.001. URL: https://www.sciencedirect.com/science/article/pii/S0167945718300575.

Yu, Junzhi, Zhengxing Wu, Ming Wang, and Min Tan (Sept. 2016). "CPG Network Optimization for a Biomimetic Robotic Fish via PSO". In: *IEEE Transactions on Neural Networks and Learning Systems* 27.9. Conference Name: IEEE Transactions on Neural Networks and Learning Systems, pp. 1962–1968. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2015.2459913.

Zaytsev, Petr, S. Javad Hasaneini, and Andy Ruina (May 2015). "Two steps is enough: No need to plan far ahead for walking balance". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729, pp. 6295–6300. DOI: 10.1109/ICRA.2015.7140083.

Zheng, Y.F. and J. Shen (Feb. 1990). "Gait synthesis for the SD-2 biped robot to climb sloping surface". In: *IEEE Transactions on Robotics and Automation* 6.1. Conference Name: IEEE Transactions on Robotics and Automation, pp. 86–96. ISSN: 2374-958X. DOI: 10.1109/70.88120.

Zhong, Mingyuan, Mikala Johnson, Yuval Tassa, Tom Erez, and Emanuel Todorov (Apr. 2013). "Value function approximation and model predictive control". In: *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. ISSN: 2325-1867, pp. 100–107. DOI: 10.1109/ADPRL.2013.6614995.

Zhu, Huaijiang, Avadesh Meduri, and Ludovic Righetti (Sept. 2022). *Efficient Object Manipulation Planning with Monte Carlo Tree Search*. arXiv:2206.09023 [cs]. DOI: 10.48550/arXiv.2206.09023. URL: http://arxiv.org/abs/2206.09023.

Čaić, Martina, Gaby Odekerken, and Dominik Mahr (Apr. 2018). "Service robots: Value co-creation and co-destruction in elderly care networks". In: *Journal of Service Management* 29. DOI: 10.1108/JOSM-07-2017-0179.