

University of Duisburg-Essen  
Faculty of Engineering  
Chair of Mechatronics

# Highly-Dynamic Movements of a Humanoid Robot Using Whole-Body Trajectory Optimization

Master Thesis  
Maschinenbau (M.Sc.)

Julian Eßer  
Student ID: 3015459

<b>First examiner</b>	Prof. Dr. Dr. h.c. Frank Kirchner (DFKI)
<b>Second examiner</b>	Dr.-Ing. Tobias Bruckmann (UDE)
<b>Supervisor</b>	Dr. rer. nat. Shivesh Kumar (DFKI)
<b>Supervisor</b>	Dr. Carlos Mastalli (University of Edinburgh)
<b>Supervisor</b>	Dr. Olivier Stasse (LAAS-CNRS)

September 18, 2020



---

# Declaration

---

This study was carried out at the Robotics Innovation Center of the German Research Center for Artificial Intelligence in the Advanced AI Team on Mechanics & Control.

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

Bremen, September 18, 2020

---

Julian Eßer

---

# Abstract

---

In order to further close the gap between robots and their natural counterparts, current research is driving towards exploiting the natural dynamics of robots. This requires a rethinking about the way we control robots for moving in a more dynamic, efficient and natural way.

Dynamic bipedal locomotion is a challenging problem and remains an open area of research. A key characteristic is the decoupling between the center of mass and the multi-body dynamics. Particular difficulties arise from effective underactuation, the mechanism complexity, as well as nonlinear and hybrid dynamics.

A common approach is to decompose this problem into smaller sub-problems that are solved sequentially. Many state-of-the-art frameworks rely on trajectory optimization based on reduced centroidal dynamics, which is transferred to a whole-body trajectory via feedback linearization using Inverse Kinematics (IK) or Inverse Dynamics (ID). Recent research indicates that solving this mapping with a local optimal control solver, namely Differential Dynamic Programming (DDP), instead of IK/ID, produces more efficient motions, with lower forces and impacts.

This master thesis contributes to the research field of dynamic bipedal locomotion by applying, evaluating and extending DDP-based whole-body trajectory optimization, pursuing three objectives: First, we develop an approach to constrain the optimal control problem allowing DDP to produce inherently balanced motions. Second, we evaluate our approach for quasi-static and dynamic motions in a real-time physics simulation and in real-world experiments on the lightweight and biologically inspired RH5 humanoid robot. Third, we examine the limits of the derived whole-body planning approach and the system design via solving highly-dynamic movements.

**Keywords:** Humanoid Robots, Dynamic Bipedal Walking, Motion Planning, Multi-Contact Optimal Control, Differential Dynamic Programming, Whole-Body Trajectory Optimization

---

# Contents

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Related Work . . . . .	2
1.2.1	Traditional Legged Locomotion Planning . . . . .	2
1.2.2	Trajectory Optimization . . . . .	3
1.2.3	Crocoddyl Framework . . . . .	4
1.2.4	RH5 Humanoid Robot . . . . .	4
1.3	Contribution and Structure . . . . .	4
<b>2</b>	<b>Mathematical Background: Optimal Bipedal Locomotion</b>	<b>6</b>
2.1	Foundations of Bipedal Locomotion . . . . .	6
2.1.1	Terminology . . . . .	6
2.1.2	Dynamic Modeling of Legged Robots . . . . .	7
2.2	Stability Analysis: Not Falling Down . . . . .	9
2.2.1	Static Stability Criteria . . . . .	9
2.2.2	Dynamic Stability Criteria . . . . .	10
2.2.3	Stability Classification . . . . .	12
2.3	Differential Dynamic Programming (DDP) . . . . .	12
2.3.1	Finite Horizon Optimal Control . . . . .	12
2.3.2	Local Dynamic Programming . . . . .	13
2.3.3	Quadratic Approximation . . . . .	13
2.3.4	Backward Pass . . . . .	14
2.3.5	Forward Pass . . . . .	15
2.4	Handling Constraints with DDP . . . . .	15
2.4.1	DDP With Constrained Robot Dynamics . . . . .	15
2.4.2	KKT-Based DDP Algorithm . . . . .	16
2.4.3	Task-Related Constraints . . . . .	17

<b>3</b>	<b>Contact Stability Constrained DDP</b>	<b>18</b>
3.1	The Idea . . . . .	18
3.2	Center of Pressure Constraints . . . . .	18
3.2.1	Center of Pressure (CoP) Conditions . . . . .	18
3.2.2	CoP Computation . . . . .	19
3.2.3	CoP Constraints: General Case . . . . .	19
3.2.4	CoP Constraints: Special Case of Horizontal Floor . . . . .	19
3.3	Integration into the Crocoddyl Framework . . . . .	20
3.3.1	Inequality Constraints by Penalization . . . . .	20
3.3.2	Computing the Residual . . . . .	20
3.3.3	Unit-Testing Numerical Differentiation . . . . .	20
<b>4</b>	<b>Bipedal Walking Variants</b>	<b>21</b>
4.1	Formulation of the Optimization Problem . . . . .	21
4.1.1	Robot Tasks . . . . .	21
4.1.2	Contact and Impact Modeling . . . . .	21
4.1.3	Inequality Constraints for Physical Compliance . . . . .	21
4.2	Simulation Results for Increasing Gait Dynamics . . . . .	21
4.2.1	Static Walking . . . . .	21
4.2.2	Slow Dynamic Walking . . . . .	21
4.2.3	Fast Dynamic Walking . . . . .	21
4.3	Evaluation of Contact Stability . . . . .	21
4.3.1	Different Levels of CoP Restriction . . . . .	21
4.3.2	Comparison of CoP and ZMP Trajectories . . . . .	21
<b>5</b>	<b>Highly-Dynamic Movements</b>	<b>22</b>
5.1	Formulation of the Optimization Problem . . . . .	22
5.2	Simulation Results for Increasing Task Complexity . . . . .	22
5.2.1	A Simple Vertical Jump . . . . .	22
5.2.2	Forward Jumping Over Multiple Obstacles . . . . .	22
5.3	Identification of Limits in System Design . . . . .	22
5.3.1	Analysis of Joint Limits . . . . .	22
5.3.2	Analysis of Torque Limits . . . . .	22
<b>6</b>	<b>Validation in Real-Time Physics Simulation</b>	<b>23</b>
6.1	Simulation Setup . . . . .	23
6.1.1	PyBullet Simulation Environment . . . . .	23
6.1.2	Spline Interpolation of the Trajectories . . . . .	23
6.1.3	Control Approach . . . . .	23
6.2	Motion Stabilization with Joint Space Control . . . . .	23
6.2.1	Quasi-Static Movements . . . . .	23
6.2.2	Bipedal Walking Variants . . . . .	23
6.2.3	Highly-Dynamic Movements . . . . .	23

---

<b>7</b>	<b>Validation in Real-World Experiments</b>	<b>24</b>
7.1	Experimental Setup . . . . .	24
7.1.1	Global Overview . . . . .	24
7.1.2	Control Approach . . . . .	24
7.2	Verification of Consistency Between the Frameworks . . . . .	24
7.2.1	Recomputing the Contact Forces . . . . .	24
7.2.2	CoM and ZMP Trajectories . . . . .	24
7.3	Experiment I: Fast Squats . . . . .	24
7.4	Experiment II: One-Leg Balancing . . . . .	24
7.5	Experiment III: Static Walking . . . . .	24
7.6	Experiment IV: Dynamic Walking . . . . .	24
<b>8</b>	<b>Conclusion and Outlook</b>	<b>25</b>
8.1	Thesis Summary . . . . .	25
8.2	Future Directions . . . . .	25
	<b>Bibliography</b>	<b>26</b>

---

# Acronyms

---

**CoM** Center of Mass

**CoP** Center of Pressure

**DDP** Differential Dynamic Programming

**DoF** Degrees of Freedom

**EoM** Equations of Motion

**FCoM** Floor Projection of Center of Mass

**FD** Forward Dynamics

**FDDP** Feasibility-driven Differential Dynamic Programming

**ID** Inverse Dynamics

**IK** Inverse Kinematics

**KKT** Karush-Kuhn-Tucker

**LIP** Linear Inverted Pendulum

**OC** Optimal Control

**SP** Support Polygon

**TO** Trajectory Optimization

**ZMP** Zero-Moment Point

# CHAPTER 1

---

## Introduction

---

This introduction guides the reader to the aim of the master's thesis. Beginning with a motivation on humanoid robots a general problem statement is derived. Thereupon, two approaches are presented that are pursuing different ideas of how to solve this problem. Building upon these methods, the aim and specific objectives of the thesis are defined. Lastly, a brief overview of the thesis structure is provided.

### 1.1 Motivation

**Why Robots** Robotics research is highly motivated by the idea of creating machines with the ability to autonomously explore and interact with complex and dynamic environments. These intelligent agents can act in surroundings that are either inaccessible or dangerous to humans or support us in everyday life tasks.

#### Why Legged Robots

**Why Humanoids** Nature often plays a crucial rule and serves as source of inspiration in the design process of such systems. This is especially true for the research field of humanoid robots, which deals with robots that are generally inspired by human capabilities and share similar kinematics, sensing and behavior. Many of the objects that we interact with on a daily basis, are tailored to human form and human behavior, e.g. doors, stairs or tools. This is equally true for the environments that we move in. Humans make use of their legs to climb stairs, lean towards difficult postures or traverse rough terrain. [1, Ch. 67].



**Vision: Exploiting Natural Dynamics** We perform all these tasks seemingly effortless. But compared to current robots, the dynamic capabilities of humans and animals are still outstanding in terms of versatility, speed, efficiency and robustness [2]. In order to further close the gap between robots and their natural counterparts, current research is driving towards exploiting the natural dynamics of robots. This implies mutually dependent changes about how we think about the robots design [3] on the one hand and about ways of controlling them on the other hand, in order to move in a more dynamic, efficient and natural way [4, 5, 6].

**Problem: Challenges in Legged Locomotion** Legged locomotion is a challenging task since there is an ongoing need to avoid falling down, especially in contrast to wheeled robots [7]. While some humanoids are able to perform statically stable gaits, bipedal walking mostly introduces a new level of complexity in terms of *dynamic* balancing.[1, Ch. 67].

The difficulty of creating bipedal walking has several causes. The first difficulty is due to the mechanism complexity, i.e. dealing with high-dimensional degrees of freedom, leading to potentially expensive computations. Secondly, legged locomotion is subject to different contact and impact collisions, resulting in multi-phase models and hence hybrid dynamics. Finally, bipeds face the problem of effective underactuation, further restricting the applicable control approaches [8, Ch. 1].

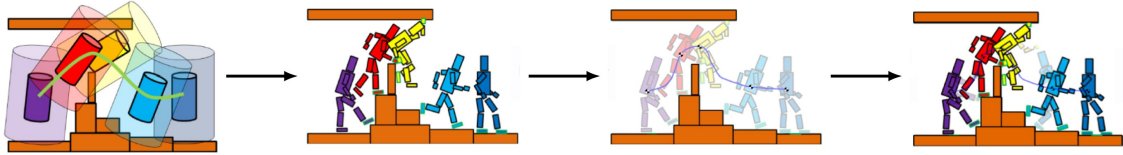
*Dynamic* bipedal locomotion adds additional complexity to this problem. A central characteristic of walking dynamically is that the center of mass (COM) partially leaves the biped's support polygon. Furthermore there is the need of generating feasible, controllable limit cycles. When considering running motions, difficulties are faced regarding the conservation of angular momentum, which implies restrictions on the controllability during flight phases [8, Ch. 1].

## 1.2 Related Work

There are existing numerous ways to generate feasible motion plans for robotic systems. Common approaches are relying on simplified dynamic models, while recent ones make use of numerical optimization for generating motion plans.

### 1.2.1 Traditional Legged Locomotion Planning

Previously we have already seen, why locomotion synthesis for legged robots is a challenging problem. Solving this global problem typically is approached by splitting it up into successive subproblems that are solved sequentially: (i) Contact planner, (ii) Centroidal pattern generator, (iii) Whole-body motion generator (fig. 1.1) [9]. The first stage of traditional legged locomotion planning is a contact planner that selects appropriate foothold position and step timings. In other words, this component predefines *where* and *when* external forces can act on the system. Once these force locations have been predetermined, the goal is to generate a body motion, i.e.



**Figure 1.1:** Tradional Legged Locomotion Planning [10].

a Center of Mass (CoM) and end-effector trajectories, which can be created with the available forces. A common approach is to model the robot as a Linear Inverted Pendulum (LIP) and find a motion where the Center of Pressure (CoP) remains inside the Support Polygon (SP). The LIP can be solved analytically and efficiently and consequently has been used in a variety of approaches [11, 12, 13, 14]. From the contact sequence and the centroidal trajectory, a dynamic-physical whole-body trajectory has to be found. This often is done by solving the Inverse Kinematics (IK) [15] or operational-space Inverse Dynamics (ID) [16] of the system and produces appropriate joint positions or torques, respectively that can be applied to a physical system.

This approach is beneficial in terms of computation time but comes at the cost of limited motion complexity and energy efficiency. The first limitation comes from the underlying assumptions of the LIP model. More complex motions and terrains might require to place feet at different heights, reorientation of the base or jump vertically [17]. Another limitation arises from the fact whole-body planning is proven to produce more efficient motions, with lower forces and impacts than a simple program such as an IK solver [18].

### 1.2.2 Trajectory Optimization

Generating motion plans for dynamic systems is subject to Trajectory Optimization (TO) algorithms that belong to the broad research field of Optimal Control (OC). TO can be described as the process of finding a state-control sequence, which locally minimizes a given cost function. This is appealing, since one can specify the behavior of a robot directly in the task space, e.g. a desired end-effector trajectory. There are existing different methods to formulate a TO, namely *direct* and *indirect* methods.

**Goal: Less hand-crafted components + Easy Formulation in Task Space**

**Direct vs Indirect Methods**

**TO based on reduced centroidal dynamics**

**Whole-body TO**

### 1.2.3 Crocoddyl Framework

Crocoddyl (Contact RObot COntrol by Differential DYnamic Library) is a recently presented open-source framework for efficient multi-contact optimal control [19].

Crocoddyl efficiently computes optimal robot trajectories with pre-defined contact phases. Its solver is based on various efficient Differential Dynamic Programming (DDP)-like algorithms, that are introduced in sections 2.3 to 2.4 in more detail. Along with the framework, a novel optimal control algorithm called Feasibility-driven Differential Dynamic Programming (FDDP) is introduced. The FDDP algorithm is an improved version of the classical DDP algorithm, which shows a greater globalization strategy due to two modifications. First, the backward pass also allows for infeasible state and control trajectories. Second, forward pass keeps the gaps open within the first iterations. The framework allows the computation of highly-dynamic movements (e.g. jumping, front-flip) within few milliseconds.

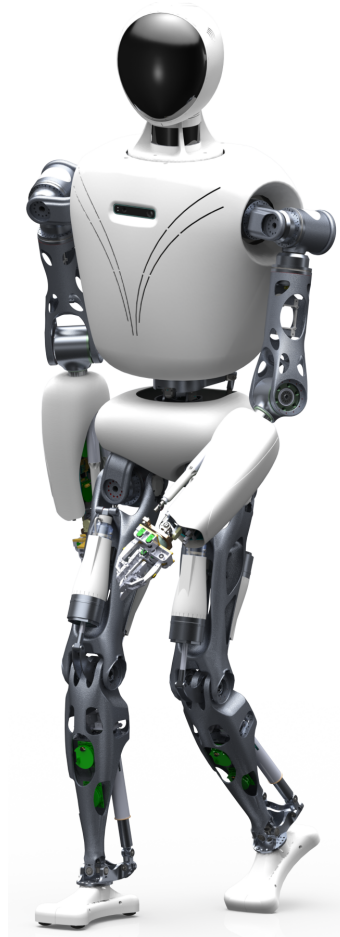
However, these exemplary case studies do not guarantee inherent balance of the motions, leading to trajectories that are hard to stabilize on a real system. To this end, we present a generic method for constraining DDP-like solvers in order to generate inherently balanced, dynamic motions that are applicable on real robots.

### 1.2.4 RH5 Humanoid Robot

The derived motion planning approach has been tested both in simulation and real-world experiments on a full-size humanoid robot. RH5 is a lightweight and biologically inspired humanoid that has recently been developed at DFKI Robotics Innovation Center[20].

The RH5 humanoid robot (see fig. 1.2) is designed to mimic the human anatomy with a total size of 200cm, a weight of 62kg and a total of 32 Degrees of Freedom (DoF). The two legs account for 12 DoF, the torso and neck kinematics each for three and the arms and grippers of the robot for 14 DoF. In order to achieve a high dynamic performance, the robot's design follows a series-parallel hybrid approach. Consequently, linkages and parallel mechanisms are utilized in most of the robots joints, e.g. the hip-flexion-extension, knee, ankle, torso and wrist. A comparison of RH5 with other state of the art humanoid robots revealed several advantages of this design approach, including better maximum velocity and torque of the ankle as well as an advantageous weight of the lower leg [21]. The interested reader can find a comprehensive introduction on series-parallel hybrid robots in [22, Ch.2].

## 1.3 Contribution and Structure



**Figure 1.2:** The recently presented RH5 is a lightweight and biologically inspired humanoid used as experimental platform within this thesis.

---

# Mathematical Background: Optimal Bipedal Locomotion

---

The second chapter provides the reader with fundamentals regarding terminology, modeling and stability analysis in the context of humanoid robotics, presents the class of used algorithms and relevant extensions and introduces the RH5 humanoid robot serving as experimental platform for this thesis.

## 2.1 Foundations of Bipedal Locomotion

### 2.1.1 Terminology

In order to describe the locomotion of a humanoid robot, specific terms are required that are introduced within this section. Vukobratović et al. provide an extensive introduction to the terminology related to bipedal walking [23], concisely summarized by Dekker [24].

#### **Walk**

Walk can be defined as: “*Movement by putting forward each foot in turn, not having both feet off the ground at once*”.

#### **Run**

Run in turn is characterized by a movement where partially both feet leaving the ground at the same time.

#### **Gait**

The way each human walks and runs is unique, hence gait can be defined as: “*Man-*

*ner of walking or running*".

### Periodic gait

If a gait is realized by repeating each locomotion phase in an identical <sup>1</sup> way, the gait is referred to as *periodic*.

### Symmetric gait

If the left and right leg move in an identical but time-shifted manner, the gait is referred to as *symmetric*.

### Double Support

A situation where the humanoid has two isolated contact surfaces with the ground.

### Single Support

A situation where the humanoid has only one contact surface with the ground.

### Support Polygon

The support polygon is formed by the *convex hull* about the ground contact points.

### Swing foot

This term refers to the leg that is performing a step, i.e. moving through the air.

### Supporting foot

This term refers to the leg that is in contact with the ground, supporting all the weight of the humanoid.

## 2.1.2 Dynamic Modeling of Legged Robots

In the following, the dynamic model for floating base systems, such as legged robots, is derived based on a general formulation. A concise introduction to dynamic modeling is presented with [25], comprehensive studies can be found in [26, 27, 28].

### General Formulation

Mathematical models of a robot's dynamics describe the motion as a function of time and control inputs. These models are the basis for both simulation and control of robotic systems. In an abstract form, the Equations of Motion (EoM) can be written as:

$$F(\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t), \mathbf{u}(t), t) = 0, \quad (2.1)$$

where

- $\mathbf{t}$  is the time variable,

---

<sup>1</sup>The locomotion phase can be identical w.r.t. the step size or the duration, depending on the index.

- $\mathbf{q}$  is the vector of generalized coordinates,
- $\dot{\mathbf{q}}$  is the first time derivative (velocity) of  $\mathbf{q}$ ,
- $\ddot{\mathbf{q}}$  is the second time derivative (acceleration) of  $\mathbf{q}$  and
- $\mathbf{u}$  is the vector of control inputs.

Consequently, the EoM provide a mapping between the control space on the one hand and the state space of robot on the other hand. Typical methods for computing the closed-form solution of the EoM are e.g. the classical *Newton-Euler* [29] method or the *Lagrange method* [30], where the former is based on principles for conservation of linear and angular momenta and the latter utilizes energy-based functions expressed in generalized coordinates.

### Fixed Base Systems

For applications with fixed-based robots, e.g. a robotic manipulator, the multi-body dynamics can be formulated as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}_g(\mathbf{q}), \quad (2.2)$$

where

- $\mathbf{M}(\mathbf{q})$  is the generalized inertia matrix,
- $\mathbf{C}(\mathbf{q})$  is the coriolis tensor,
- $\boldsymbol{\tau}$  is the vector of actuated joint torques and
- $\boldsymbol{\tau}_g(\mathbf{q})$  is the vector of external joint torques caused by gravity.

In contrast to the general formulation in eq. (2.1), this expression is time-invariant. Hence, eq. (2.2) can be used for computing the Forward Dynamics (FD), as well as the ID of a robotic system.

### Floating Base Systems

A floating base system is characterized by having a base that is free to move, rather than being fixed in space. Consequently, the vector of generalized coordinates  $\mathbf{q}$  not only contains the joints angles, but also accounts for the position and orientation of the floating base. Legged robots belong to this category of rigid-body systems as they make and break contacts with their environment in order to move. Contrary to manipulators, contacts need to be actively enforced by holonomic constraints for legged robots. There are namely two different types of contact constraints that can be applied: point contacts (3d) or surface contacts (6d).

For the case of point contacts, the dynamics of the floating base system become

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{S}^T \boldsymbol{\tau} + \boldsymbol{\tau}_g(\mathbf{q}) + \sum_{i=1}^k \mathbf{J}_{C_i}^T \mathbf{f}_i,$$

where

- $\mathbf{S}$  is the selection matrix of actuated joints,
- $\mathbf{J}_{C_i}$  is the Jacobian at the location of a contact point  $C_i$  and
- $\mathbf{f}_i$  is the contact force acting at the contact point  $C_i$ .

For the case of surface contacts, such as a flat foot on a flat floor, modeling a point contact is not sufficient since it only constrains the translation. In order to also account for the rotational constraints enforced by the geometry one could take into account multiple point contacts. A non-redundant alternative is to model more general frame contact constraints as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{S}^T \boldsymbol{\tau} + \boldsymbol{\tau}_g(\mathbf{q}) + \sum_{i=1}^k \mathbf{J}_{C_i}^T \mathbf{w}_i,$$

where  $\mathbf{w}_i$  is referred to as the *contact wrench* acting on the contact link  $i$ . This wrench stacks the resultant  $\mathbf{f}_i$  of contact forces and the moment  $\boldsymbol{\tau}_i$  exerted by these forces around the contact frame as

$$\mathbf{w}_i = (\mathbf{f}_i, \boldsymbol{\tau}_i)_{6 \times 1}.$$

For more details on contact wrenches and spatial vector algebra in general, the interested reader is referred to e.g. [28, Ch.2].

## 2.2 Stability Analysis: Not Falling Down

Humanoid robots are high-dimensional, constrained and nonlinear dynamical systems. In this section, the most common criteria for analyzing the long-term stability behavior of such complex systems are presented. Exhaustive studies on stability criteria and their relation can be found in [1, 24, 31].

### 2.2.1 Static Stability Criteria

#### Floor Projection of the Center of Mass (FCoM)

Consider the case of a robot that is not moving, i.e. a humanoid in static double support. In that case, the only forces acting on the humanoid are the ones caused



by gravity. These forces can be represented by a virtual force acting on the CoM of the robot. The position of the CoM w.r.t. the base frame can be described by

$$\mathbf{p}_{CoM} = \frac{\sum_{i=1}^n m_i \mathbf{p}_i}{\sum_{i=1}^n m_i},$$

where the robot has  $n$  links and  $\mathbf{p}_i$  indicate the according link distances of the individual CoMs. The Floor Projection of Center of Mass (FCoM) equals the first two components of the CoM position vector  $\mathbf{p}_{CoM}$  and the following relation holds:

$$\sum_{i=1}^n ((\mathbf{p}_{FCoM} - \mathbf{p}_i) \times m_i \mathbf{g}) = \mathbf{0}.$$

The FCoM can be used as a static stability margin, ensuring the motionless robot will not tip over or fall, if  $\mathbf{p}_{FCoM}$  always remains inside the SP. Note that this criteria is also applicable in so called *quasi-static* movements, where static forces are still dominating dynamic forces.

### 2.2.2 Dynamic Stability Criteria

In case of faster motions, dynamic forces will exceed the static forces and can not be neglected anymore. The acting forces can be divided into contact forces and gravity/inertial forces, where the so called Zero-Moment Point (ZMP) is based on the former, and the CoP on the latter. In the following, both concepts are introduced according to the description in [32] with a nomenclature equivalent to [25].

#### Center of Pressure (CoP)

The CoP is defined as the point, where the field of pressure forces acting on the sole is equivalent to a single resultant force where the resultant moment is zero. Hence the CoP is a local quantity that is derived from the interaction forces at the contact surface.

Considering the case of a foot contacting a plane surface, the resultant contact force  $\mathbf{f}^c$  is exerted by the environment onto the robot. This force consists of the resultant pressure force  $\mathbf{f}^p = (\mathbf{f}^c \cdot \mathbf{n})\mathbf{n}$ , as well as the resultant friction force  $\mathbf{f}^f = \mathbf{f}^c - \mathbf{f}^p$ . Hence, the following conditions hold:

$$\begin{aligned} \tau_O^p &= \mathbf{0} \\ \mathbf{p}_{CoP} \times (\mathbf{f}^p \cdot \mathbf{n})\mathbf{n} &= -\tau_O^p \\ (\mathbf{f}^p \cdot \mathbf{n})\mathbf{n} \times \mathbf{p}_{CoP} \times \mathbf{n} &= -\mathbf{n} \times \tau_O^p \end{aligned}$$

Since both the sole point  $O$  and  $\mathbf{p}_{CoP}$  belong to the same plane, we get:

$$\mathbf{p}_{CoP} = \frac{\mathbf{n} \times \tau_O^p}{\mathbf{f}^p \cdot \mathbf{n}}.$$

Finally, friction forces are tangent to the contact surface and their moment is aligned with  $\mathbf{n}$ , so we equivalently can write this relationship as:

$$\mathbf{p}_{CoP} = \frac{\mathbf{n} \times \boldsymbol{\tau}_O^c}{\mathbf{f}^c \cdot \mathbf{n}}. \quad (2.3)$$

Equation (2.3) can be used to compute the CoP expressed in the local contact frame.

### Zero-Moment Point (ZMP)

The ZMP is defined as a point on the ground where the *tipping moment* acting on the biped equals zero. This condition can be interpreted as a constraint on the contact moments, which contains *at least* the roll and pitch direction. Originally, the concept has been introduced in [33], it has been reviewed in [34] and made popular with [11].

The concept is build upon two key assumptions:

- There exists one planar contact surface (i.e. no multiple surfaces like on rough terrain)
- The friction is sufficiently high to prevent sliding of the feet

From the Newton-Euler equations, the motion of the biped can be written as

$$\begin{aligned} m\ddot{\mathbf{p}}_{CoM} &= m\mathbf{g} + \mathbf{f}^c \\ \dot{\mathbf{L}}_O &= \mathbf{p}_{CoM} \times m\mathbf{g} + \boldsymbol{\tau}_{CoM}^c, \end{aligned}$$

where  $m$  denotes the total mass of the robot,  $\mathbf{g}$  is the gravity vector,  $\ddot{\mathbf{p}}_{CoM}$  the centroidal acceleration,  $\dot{\mathbf{L}}_O$  the change of the angular momentum.  $\mathbf{w}_{CoM}^c = (\boldsymbol{\tau}_{CoM}^c, \mathbf{f}^c)_{6 \times 1}$  denotes the sum of all contact wrenches in the CoM frame. The gravito-inertial wrench of the robot can be defined as

$$\begin{aligned} \mathbf{f}^{gi} &= m(\mathbf{g} - \ddot{\mathbf{p}}_{CoM}) \\ \boldsymbol{\tau}_O^{gi} &= \mathbf{p}_{CoM} \times m\mathbf{g} - \dot{\mathbf{L}}_O. \end{aligned}$$

Using the wrench form of the Newton-Euler equations

$$\mathbf{w}^{gi} + \mathbf{w}^c = \mathbf{0}, \quad (2.4)$$

one can derive the ZMP, for the case of a planar surface, as

$$\mathbf{p}_{ZMP} = \frac{\mathbf{n} \times \boldsymbol{\tau}_O^{gi}}{\mathbf{f}^{gi} \cdot \mathbf{n}}. \quad (2.5)$$

In practice, one can use this formula to compute the ZMP from force sensors or from an inertial measurement unit.

### Coincidence of ZMP and CoP

As Sardain and Bessonnet outline, both the ZMP and the CoP yield the same point for the case of bipedal walking on a single plane surface. Comparing eq. (2.5) with eq. (2.3), we recognize the only difference is that the former is applied to the (global) gravito-inertial wrench, while the latter is applied to the (local) contact wrench. If we recall the Newton-Euler equations from eq. (2.4), it becomes clear why both points coincide when there is only one contact plane.

#### 2.2.3 Stability Classification

There are existing several classifications on stability, which will be defined in the following according to [8, Sec.1.2.1] and [31]. See Vukobratović et al. for more details on differentiating the terms dynamic stability and dynamic balance [23].

#### Statically Stable Motion

The gait or movement of a humanoid is classified as *statically stable*, if the FCoM does not leave the SP during the entire motion or gait. Consequently, the humanoid will remain in a stable position, whenever the movement is stopped. Typically, these kind of stability are only obtained with very low walking velocities or quasi-static motions, where the static forces dominate the dynamic forces.

#### Dynamically Stable Motion

If the FCoM partially leaves the SP at some point during the gait, but the CoP (or ZMP) always remains within the SP, the gait or movement is classified as *dynamically stable*. This stability margin is extremely useful for flat-foot dynamic walking since it prevents the foot from rotating around the boundary of the SP.

## 2.3 Differential Dynamic Programming (DDP)

This section describes the basics of DDP, which is an OC algorithm that belongs to the TO class. The algorithm was introduced in 1966 by Mayne [35]. A modern description of the algorithm using the same notations as below can be found in [36, 37].

### 2.3.1 Finite Horizon Optimal Control

We consider a system with discrete-time dynamics, which can be modeled as a generic function  $\mathbf{f}$

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \quad (2.6)$$

that describes the evolution of the state  $\mathbf{x} \in \mathbf{R}^n$  from time  $i$  to  $i+1$ , given the control  $\mathbf{u} \in \mathbf{R}^m$ . A complete trajectory  $\{\mathbf{X}, \mathbf{U}\}$  is a sequence of states  $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_N\}$  and control inputs  $\mathbf{U} = \{\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_N\}$  satisfying eq. (2.6). The *total cost*  $J$  of a

trajectory can be written as the sum of running costs  $l$  and a final cost  $l_f$  starting from the initial state  $\mathbf{x}_0$  and applying the control sequence  $\mathbf{U}$  along the finite time-horizon:

$$J(\mathbf{x}_0, \mathbf{U}) = l_f(\mathbf{x}_N) + \sum_{i=0}^{N-1} l(\mathbf{x}_i, \mathbf{u}_i). \quad (2.7)$$

As discussed in chapter 1, *indirect* methods such DDP represent the trajectory implicitly solely via the optimal controls  $\mathbf{U}$ . The states  $\mathbf{X}$  are obtained from forward simulation of the system dynamics, i.e. integration eq. (2.6). Consequently, the solution of the optimal control problem is the minimizing control sequence

$$\mathbf{U}^* = \underset{\mathbf{U}}{\operatorname{argmin}} J(\mathbf{x}_0, \mathbf{U}).$$

### 2.3.2 Local Dynamic Programming

Let  $\mathbf{U}_i \equiv \{\mathbf{u}_i, \mathbf{u}_{i+1}, \dots, \mathbf{u}_{N-1}\}$  be the partial control sequence, the *cost-to-go*  $J_i$  is the partial sum of costs from  $i$  to  $N$ :

$$J_i(\mathbf{x}, \mathbf{U}_i) = l_f(\mathbf{x}_N) + \sum_{j=i}^{N-1} l(\mathbf{x}_j, \mathbf{u}_j). \quad (2.8)$$

The *Value function* at time  $i$  is the optimal cost-to-go starting at  $\mathbf{x}$  given the minimizing control sequence

$$V_i(\mathbf{x}) = \min_{\mathbf{U}_i} J_i(\mathbf{x}, \mathbf{U}_i),$$

and the Value at the final time is defined as  $V_N(\mathbf{x}) \equiv l_f(\mathbf{x}_N)$ . The Dynamic Programming Principle [38] reduces the minimization over an entire sequence of controls to a sequence of minimizations over a single control, proceeding backwards in time:

$$V(\mathbf{x}) = \min_{\mathbf{u}} [l(\mathbf{x}, \mathbf{u}) + V'(\mathbf{f}(\mathbf{x}, \mathbf{u}))]. \quad (2.9)$$

Note that eq. (2.9) is referred to as the *Bellman equation* for *discrete-time* optimization problems [39]. For reasons of readability, the time index  $i$  is omitted and  $V'$  introduced to denote the Value at the next time step. The interested reader may note that the analogous equation for the case of *continuous-time* is a partial differential equation called the *Hamilton-Jacobi-Bellman equation* [40, 41].

### 2.3.3 Quadratic Approximation

DDP locally computes the optimal state and control sequences of the OC problem derived with eq. (2.9) by iteratively performing a forward and backward pass. The *backward pass* on the trajectory generates a new control sequence and is followed by a *forward pass* to compute and evaluate the new trajectory.

Let  $Q(\delta \mathbf{x}, \delta \mathbf{u})$  be the variation in the argument on the right-hand side of eq. (2.9) around the  $i^{th}(\mathbf{x}, \mathbf{u})$  pair

$$Q(\delta \mathbf{x}, \delta \mathbf{u}) = l(\mathbf{x} + \delta \mathbf{x}, \mathbf{u} + \delta \mathbf{u}) + V'(\mathbf{f}(\mathbf{x} + \delta \mathbf{x}, \mathbf{u} + \delta \mathbf{u})). \quad (2.10)$$

The DDP algorithm uses a quadratic approximation of this differential change. The quadratic Taylor expansion of  $Q(\delta \mathbf{x}, \delta \mathbf{u})$  leads to

$$Q(\delta \mathbf{x}, \delta \mathbf{u}) \approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta \mathbf{x} \\ \delta \mathbf{u} \end{bmatrix}^T \begin{bmatrix} 0 & \mathbf{Q}_x^T & \mathbf{Q}_u^T \\ \mathbf{Q}_x & \mathbf{Q}_{xx} & \mathbf{Q}_{xu} \\ \mathbf{Q}_u & \mathbf{Q}_{ux} & \mathbf{Q}_{uu} \end{bmatrix} \begin{bmatrix} 1 \\ \delta \mathbf{x} \\ \delta \mathbf{u} \end{bmatrix}. \quad (2.11)$$

The coefficients can be computed as

$$\mathbf{Q}_x = l_x + \mathbf{f}_x^T \mathbf{V}'_x, \quad (2.12a)$$

$$\mathbf{Q}_u = l_u + \mathbf{f}_u^T \mathbf{V}'_x, \quad (2.12b)$$

$$\mathbf{Q}_{xx} = l_{xx} + \mathbf{f}_x^T \mathbf{V}'_{xx} \mathbf{f}_x + \mathbf{V}'_x \cdot \mathbf{f}_{xx}, \quad (2.12c)$$

$$\mathbf{Q}_{ux} = l_{ux} + \mathbf{f}_u^T \mathbf{V}'_{xx} \mathbf{f}_x + \mathbf{V}'_x \cdot \mathbf{f}_{ux}, \quad (2.12d)$$

$$\mathbf{Q}_{uu} = l_{uu} + \mathbf{f}_u^T \mathbf{V}'_{xx} \mathbf{f}_u + \mathbf{V}'_x \cdot \mathbf{f}_{uu}. \quad (2.12e)$$

where the primes denote the values at the next time-step.

#### 2.3.4 Backward Pass

The first algorithmic step of DDP, namely the backward pass, involves computing a new control sequence on the given trajectory and consequently determining the search direction of a step in the numerical optimization. To this end, the quadratic approximation obtained from eq. (2.11), minimized with respect to  $\delta \mathbf{u}$  for some state perturbation  $\delta \mathbf{x}$ , results in

$$\delta \mathbf{u}^*(\delta \mathbf{x}) = \underset{\delta \mathbf{u}}{\operatorname{argmin}} Q(\delta \mathbf{x}, \delta \mathbf{u}) = -\mathbf{Q}_{uu}^{-1}(\mathbf{Q}_u + \mathbf{Q}_{ux} \delta \mathbf{x}),$$

giving us an open-loop term  $\mathbf{k}$  and a feedback gain term  $\mathbf{K}$ :

$$\mathbf{k} = -\mathbf{Q}_{uu}^{-1} \mathbf{Q}_u \quad \text{and} \quad \mathbf{K} = -\mathbf{Q}_{uu}^{-1} \mathbf{Q}_{ux}.$$

The resulting locally-linear feedback policy can be again inserted into eq. (2.11) leading to a quadratic model of the Value at time  $i$ :

$$\begin{aligned} \Delta V &= -\frac{1}{2} \mathbf{k}^T \mathbf{Q}_{uu} \mathbf{k} \\ \mathbf{V}_x &= \mathbf{Q}_x - \mathbf{K}^T \mathbf{Q}_{uu} \mathbf{k} \\ \mathbf{V}_{xx} &= \mathbf{Q}_{xx} - \mathbf{K}^T \mathbf{Q}_{uu} \mathbf{K}. \end{aligned}$$

### 2.3.5 Forward Pass

After computing the feedback policy in the backward pass, the forward pass computes a corresponding trajectory by integrating the dynamics via

$$\begin{aligned}\hat{\mathbf{x}}_0 &= \mathbf{x}_0 \\ \hat{\mathbf{u}}_i &= \mathbf{u}_i + \alpha \mathbf{k}_i + \mathbf{K}_i(\hat{\mathbf{x}}_i - \mathbf{x}_i) \\ \hat{\mathbf{x}}_{i+1} &= \mathbf{f}(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i),\end{aligned}$$

where  $\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i$  are the new state-control sequences. The step size of the numerical optimization is described by the backtracking line search parameter  $\alpha$ , which iteratively is reduced starting from 1. The backward and forward passes of the DDP algorithm are iterated until convergence to the (locally) optimal trajectory.

## 2.4 Handling Constraints with DDP

By nature, the DDP algorithm presented in ?? does not take into account constraints. Tassa et al. developed a control-limited DDP [37] that takes into account box inequality constraints on the controls allowing the consideration of torque limits on real robotic systems. Budhiraja et al. proposed a DDP version for the problem of multi-phase rigid contact dynamics by exploiting the Karush-Kuhn-Tucker constraint of the rigid contact model [18]. Since physically consistent bipedal locomotion is highly dependent on making contacts with the ground, this section provides details on the above mentioned approach.

### 2.4.1 DDP With Constrained Robot Dynamics

#### Contact Dynamics

In the case of rigid contact dynamics, DDP assumes a set of given contacts of the system with the environment. Then, an equality constrained dynamics can be incorporated by formulating rigid contacts as holonomic constraints to the robot dynamics. In other words, the contact points are assumed to have a fixed position on the ground.

The unconstrained robot dynamics can be represented as

$$\mathbf{M}\dot{\mathbf{v}}_{free} = \mathbf{S}\boldsymbol{\tau} - \mathbf{b} = \boldsymbol{\tau}_b, \quad (2.13)$$

with the joint-space inertia matrix  $\mathbf{M} \in \mathbf{R}^{n \times n}$  and the unconstrained acceleration vector  $\dot{\mathbf{v}}_{free}$ . The right-hand side of eq. (2.13) represents the n-dimensional force-bias vector accounting for the control  $\boldsymbol{\tau}$ , the Coriolis and gravitational effects  $\mathbf{b}$  and the selection matrix  $\mathbf{S}$  of actuated joints.

In order to incorporate the rigid contact constraints to the robot dynamics, one can apply the Gauss principle of least constraint [42]. The idea is to minimize the deviation in acceleration between the constrained and unconstrained motion:

$$\begin{aligned} \dot{\mathbf{v}} &= \arg \min_{\mathbf{a}} \frac{1}{2} \|\dot{\mathbf{v}} - \dot{\mathbf{v}}_{free}\|_{\mathbf{M}} \\ \text{subject to} \quad & \mathbf{J}_c \dot{\mathbf{v}} + \dot{\mathbf{J}}_c \mathbf{v} = \mathbf{0}, \end{aligned} \quad (2.14)$$

where  $\mathbf{M}$  formally represents the inertia tensor over the configuration manifold  $\mathbf{q}$ . In order to express the holonomic contact constraint  $\phi(\mathbf{q})$  in the acceleration space, it needs to be differentiated twice. Consequently, the contact condition can be seen as a second-order kinematic constraints on the contact surface position where  $\mathbf{J}_c = [\mathbf{J}_{c_1} \ \cdots \ \mathbf{J}_c]$  is a stack of  $f$  contact Jacobians.

### Karush-Kuhn-Tucker (KKT) Conditions

The Gauss minimization in eq. (2.14) corresponds to an equality-constrained quadratic optimization problem. The optimal solutions  $(\dot{\mathbf{v}}, \boldsymbol{\lambda})$  must satisfy the so-called Karush-Kuhn-Tucker (KKT) conditions given by

$$\begin{bmatrix} \mathbf{M} & \mathbf{J}_c^\top \\ \mathbf{J}_c & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_b \\ -\dot{\mathbf{J}}_c \mathbf{v} \end{bmatrix}. \quad (2.15)$$

These dual variables  $\boldsymbol{\lambda}^k$  represent external wrenches at the contact level. For a given robot state and applied torques, eq. (2.15) allows a direct computation of the contact forces. To this end, the contact constraints can be solved analytically at the level of dynamics instead of introducing additional constraints in the whole-body optimization [43].

#### 2.4.2 KKT-Based DDP Algorithm

The KKT dynamics from eq. (2.15) can be expressed as a function of the state  $\mathbf{x}_i$  and the control  $\mathbf{u}_i$ :

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \\ \boldsymbol{\lambda}_i &= \mathbf{g}(\mathbf{x}_i, \mathbf{u}_i), \end{aligned} \quad (2.16)$$

where the concatenation of the configuration vector and its tangent velocity forms the state  $\mathbf{x} = (\mathbf{q}, \mathbf{v})$ ,  $\mathbf{u}$  is the input torque vector and  $\mathbf{g}(\cdot)$  is the optimal solution of eq. (2.15).

Supposing a sequence of predefined contacts, the cost-to-go of the DDP backward-pass and its respective Hessians (compare eq. (2.8) and 2.12) turn into:

$$J_i(\mathbf{x}, \mathbf{U}_i) = l_f(\mathbf{x}_N) + \sum_{j=i}^{N-1} l(\mathbf{x}_j, \mathbf{u}_j, \boldsymbol{\lambda}_j)$$

with the control inputs  $U_i$  acting on the system dynamics at time  $i$ , and first-order approximation of  $g(\cdot)$  and  $f(\cdot)$  as

$$\begin{aligned} Q_x &= l_x + g_x^T l_\lambda + f_x^T V'_x, \\ Q_u &= l_u + g_u^T l_\lambda + f_u^T V'_u, \\ Q_{xx} &\approx l_{xx} + g_x^T l_{\lambda\lambda} g_x + f_x^T V'_{xx} f_x, \\ Q_{ux} &\approx l_{ux} + g_u^T l_{\lambda\lambda} g_x + f_u^T V'_{xx} f_x, \\ Q_{uu} &\approx l_{uu} + g_u^T l_{\lambda\lambda} g_u + f_u^T V'_{xx} f_u. \end{aligned} \quad (2.17)$$

Consequently, the KKT-based DDP algorithm utilizes the set of eq. (2.17) inside the backward-pass to incorporate the rigid contacts forces, while the updated system dynamics from eq. (2.16) is utilized during the forward-pass of the algorithm.

### 2.4.3 Task-Related Constraints

An important part of the motion generation is the execution of desired actions, e.g. grasping an object, moving the CoM or performing a robot step. For formulating these task-related constraints, we follow the notation used in [10].

An arbitrary task can be formulated as a regulator:

$$h_{task_k}(x_k, u_k) = s_{task}^d - s_{task}(x_k, u_k),$$

where the task is defined as the difference between the desired and current feature vectors  $s_{task}^d$  and  $s_{task}(x_k, u_k)$ , respectively. The task at each node can be added to the cost function via penalization as:

$$l_k(x_k, u_k) = \sum_{j \in tasks} w_{j_k} \| h_{j_k}(x_k, u_k) \|^2,$$

where  $w_{j_k}$  assigned to task  $j$  at corresponding time  $k$ . The DDP algorithm utilized the derivatives of the regulators functions, namely computing the Jacobians and Hessians of the cost functions. In the scope of this thesis, the following tasks are handled

$$tasks \subseteq \{CoM, LH_{SE(3)}, RH_{SE(3)}, LF_{SE(3)}, RF_{SE(3)}\} : \quad (2.18)$$

1) the CoM tracking ( $CoM$ ), 2) the tracking of the left- and right-hand pose ( $LH_{SE(3)}, RH_{SE(3)}$ ) and 3) the tracking of the left- and right-feet pose ( $LF_{SE(3)}, RF_{SE(3)}$ ).



---

# Contact Stability Constrained DDP

---

### 3.1 The Idea

### 3.2 Center of Pressure Constraints

#### 3.2.1 Center of Pressure (CoP) Conditions

Conditions for CoP inside convex hull:

$$\begin{aligned} \text{Pitch} : -X &\leq C_x \leq X \\ \text{Roll} : -Y &\leq C_y \leq Y \end{aligned} \tag{3.1}$$

Represented by the four inequality conditions:

$$\begin{aligned} -X &\leq C_x \\ C_x &\leq X \\ -Y &\leq C_y \\ C_y &\leq Y \end{aligned}$$

Rewritten for readability as

$$\begin{aligned} X + C_x &\geq 0 \\ X - C_x &\geq 0 \\ Y + C_y &\geq 0 \\ Y - C_y &\geq 0 \end{aligned} \tag{3.2}$$

### 3.2.2 CoP Computation

$$\mathbf{p}_{CoP} = \frac{\mathbf{n} \times \boldsymbol{\tau}_O^c}{\mathbf{f}^c \cdot \mathbf{n}} = \frac{\begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \times \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}}{\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \cdot \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}} = \begin{bmatrix} n_y t_z - n_z t_y \\ n_z t_x - n_x t_z \\ n_x t_y - n_y t_x \end{bmatrix} \cdot \frac{1}{f_x n_x + f_y n_y + f_z n_y} \quad (3.3)$$

$$C_x = \frac{n_y t_z - n_z t_y}{f_x n_x + f_y n_y + f_z n_y} \quad (3.4a)$$

$$C_y = \frac{n_z t_x - n_x t_z}{f_x n_x + f_y n_y + f_z n_y} \quad (3.4b)$$

### 3.2.3 CoP Constraints: General Case

$$\begin{aligned} X + \frac{n_y t_z - n_z t_y}{f_x n_x + f_y n_y + f_z n_y} &\geq 0 \\ X - \frac{n_y t_z - n_z t_y}{f_x n_x + f_y n_y + f_z n_y} &\geq 0 \\ Y + \frac{n_z t_x - n_x t_z}{f_x n_x + f_y n_y + f_z n_y} &\geq 0 \\ Y - \frac{n_z t_x - n_x t_z}{f_x n_x + f_y n_y + f_z n_y} &\geq 0 \end{aligned} \quad (3.5)$$

These conditions can be written in matrix form as:

$$\begin{bmatrix} X n_0 & X n_1 & X n_2 & 0 & -n_2 & n_1 \\ X n_0 & X n_1 & X n_2 & 0 & n_2 & -n_1 \\ Y n_0 & Y n_1 & Y n_2 & n_2 & 0 & -n_0 \\ Y n_0 & Y n_1 & Y n_2 & -n_2 & 0 & n_0 \end{bmatrix} \cdot \begin{bmatrix} f^x \\ f^y \\ f^z \\ \tau^x \\ \tau^y \\ \tau^z \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.6)$$

### 3.2.4 CoP Constraints: Special Case of Horizontal Floor

For the case of horizontal floor, with the according normal vector  $\mathbf{n} = [0, 0, 1]$ , the CoP can be computed to:

$$\mathbf{p}_{CoP} = \begin{bmatrix} -t_y/f_z \\ t_x/f_z \\ 0 \end{bmatrix} \quad (3.7)$$

Represented by the four inequality conditions:

$$\begin{aligned}
 X - \frac{\tau_y}{f_z} &\geq 0 \\
 X + \frac{\tau_y}{f_z} &\geq 0 \\
 Y + \frac{\tau_x}{f_z} &\geq 0 \\
 Y - \frac{\tau_x}{f_z} &\geq 0
 \end{aligned} \tag{3.8}$$

These conditions are implemented via:

$$\begin{bmatrix} 0 & 0 & X & 0 & -1 & 0 \\ 0 & 0 & X & 0 & 1 & 0 \\ 0 & 0 & Y & 1 & 0 & 0 \\ 0 & 0 & Y & -1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} f^x \\ f^y \\ f^z \\ \tau^x \\ \tau^y \\ \tau^z \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.9}$$

### 3.3 Integration into the Crocoddyl Framework

#### 3.3.1 Inequality Constraints by Penalization

#### 3.3.2 Computing the Residual

Computation of the cost residual:

$$\mathbf{r} = \mathbf{A} * \mathbf{w} \tag{3.10}$$

Computation of the cost value via a bounded, quadratic activation:

$$\begin{aligned}
 cost(r) &= 0.5 \cdot ||r||^2 \mid lb \geq r \geq ub \\
 cost(r) &= 0 \mid lb < r < ub
 \end{aligned} \tag{3.11}$$

Details:  $cost = 0.5 * \text{squaredNorm}(r - lb) + 0.5 * \text{squaredNorm}(r - ub)$  i.e. everything lower than lb and greater than ub will affect the cost.

#### 3.3.3 Unit-Testing Numerical Differentiation

---

## Bipedal Walking Variants

---

### 4.1 Formulation of the Optimization Problem

#### 4.1.1 Robot Tasks

#### 4.1.2 Contact and Impact Modeling

#### 4.1.3 Inequality Constraints for Physical Compliance

### 4.2 Simulation Results for Increasing Gait Dynamics

#### 4.2.1 Static Walking

#### 4.2.2 Slow Dynamic Walking

#### 4.2.3 Fast Dynamic Walking

### 4.3 Evaluation of Contact Stability

#### 4.3.1 Different Levels of CoP Restriction

#### 4.3.2 Comparison of CoP and ZMP Trajectories

---

## Highly-Dynamic Movements

---

### 5.1 Formulation of the Optimization Problem

### 5.2 Simulation Results for Increasing Task Complexity

#### 5.2.1 A Simple Vertical Jump

#### 5.2.2 Forward Jumping Over Multiple Obstacles

### 5.3 Identification of Limits in System Design

#### 5.3.1 Analysis of Joint Limits

#### 5.3.2 Analysis of Torque Limits

---

# Validation in Real-Time Physics Simulation

---

## 6.1 Simulation Setup

### 6.1.1 PyBullet Simulation Environment

### 6.1.2 Spline Interpolation of the Trajectories

### 6.1.3 Control Approach

## 6.2 Motion Stabilization with Joint Space Control

### 6.2.1 Quasi-Static Movements

### 6.2.2 Bipedal Walking Variants

### 6.2.3 Highly-Dynamic Movements

---

## Validation in Real-World Experiments

---

### 7.1 Experimental Setup

#### 7.1.1 Global Overview

#### 7.1.2 Control Approach

### 7.2 Verification of Consistency Between the Frameworks

#### 7.2.1 Recomputing the Contact Forces

#### 7.2.2 CoM and ZMP Trajectories

### 7.3 Experiment I: Fast Squats

### 7.4 Experiment II: One-Leg Balancing

### 7.5 Experiment III: Static Walking

### 7.6 Experiment IV: Dynamic Walking

## CHAPTER 8

---

# Conclusion and Outlook

---

### 8.1 Thesis Summary

### 8.2 Future Directions



---

## Bibliography

---

- [1] Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer, 2016.
- [2] Marco Hutter, Christian Gehring, Michael Bloesch, Mark A Hoepfflinger, C David Remy, and Roland Siegwart. Starleth: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In *Adaptive Mobile Robotics*, pages 483–490. World Scientific, 2012.
- [3] Jerry E Pratt and Benjamin T Krupp. Series elastic actuators for legged robots. In *Unmanned Ground Vehicle Technology Vi*, volume 5422, pages 135–144. International Society for Optics and Photonics, 2004.
- [4] Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712):1082–1085, 2005.
- [5] Sami Haddadin, Felix Huber, and Alin Albu-Schäffer. Optimal control for exploiting the natural dynamics of variable stiffness robots. In *2012 IEEE International Conference on Robotics and Automation*, pages 3347–3354. IEEE, 2012.
- [6] Jerry E Pratt. Exploiting inherent robustness and natural dynamics in the control of bipedal walking robots. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE DEPT OF ELECTRICAL ENGINEERING AND CS, 2000.
- [7] Marc H Raibert. *Legged robots that balance*. MIT press, 1986.
- [8] Eric R Westervelt, Jessy W Grizzle, Christine Chevallereau, Jun Ho Choi, and Benjamin Morris. *Feedback control of dynamic bipedal robot locomotion*. CRC press, 2018.
- [9] Justin Carpentier, Andrea Del Prete, Steve Tonneau, Thomas Flayols, Florent Forget, Alexis Mifsud, Kevin Giraud, Dinesh Atchuthan, Pierre Fernbach, Rohan Budhiraja, et al. Multi-contact locomotion of legged robots in complex

- environments—the loco3d project. In *RSS Workshop on Challenges in Dynamic Legged Locomotion*, page 3p, 2017.
- [10] Kevin Giraud, Pierre Fernbach, Gabriele Buondonno, Carlos Mastalli, Olivier Stasse, et al. Motion planning with multi-contact and visual servoing on humanoid robots. 2020.
- [11] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1620–1626. IEEE, 2003.
- [12] Mrinal Kalakrishnan, Jonas Buchli, Peter Pastor, Michael Mistry, and Stefan Schaal. Fast, robust quadruped locomotion over challenging terrain. In *2010 IEEE International Conference on Robotics and Automation*, pages 2665–2670. IEEE, 2010.
- [13] Alexander W Winkler, Carlos Mastalli, Ioannis Havoutis, Michele Focchi, Darwin G Caldwell, and Claudio Semini. Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5148–5154. IEEE, 2015.
- [14] C Dario Bellicoso, Fabian Jenelten, Péter Fankhauser, Christian Gehring, Jemin Hwangbo, and Marco Hutter. Dynamic locomotion and whole-body control for quadrupedal robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3359–3365. IEEE, 2017.
- [15] Bernard Espiau, François Chaumette, and Patrick Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, 1992.
- [16] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.
- [17] Alexander W Winkler. *Optimization-based motion planning for legged robots*. PhD thesis, ETH Zurich, 2018.
- [18] Rohan Budhiraja, Justin Carpentier, Carlos Mastalli, and Nicolas Mansard. Differential dynamic programming for multi-phase rigid contact dynamics. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–9. IEEE, 2018.
- [19] Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Righetti Ludovic Carpentier, Justin, Sethu

- Vijayakumar, and Nicolas Mansard. Crocoddyl: An Efficient and Versatile Framework for Multi-Contact Optimal Control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.
- [20] H Peters, P Kampmann, and M Simnofske. Konstruktion eines zweibeinigen humanoiden roboters. *Proceedings of the 2. VDI Fachkonferenz Humanoide Roboter, VDI Fachkonferenz Humanoide Roboter*, 2017.
- [21] Shivesh Kumar, Hendrik Wöhrle, José de Gea Fernández, Andreas Müller, and Frank Kirchner. A survey on modularity and distributivity in series-parallel hybrid robots. *Mechatronics*, 68:102367, 2020.
- [22] Shivesh Kumar. *Modular and Analytical Methods for Solving Kinematics and Dynamics of Series-Parallel Hybrid Robots*. PhD thesis, Universität Bremen, 2019.
- [23] M Vukobratović, Branislav Borovac, and Veljko Potkonjak. Towards a unified understanding of basic notions and terms in humanoid robotics. *Robotica*, 25(1):87–101, 2007.
- [24] MHP Dekker. Zero-moment point method for stable biped walking. *Eindhoven University of Technology*, 2009.
- [25] Stephane Caron. Legged locomotion teaching material. <https://scaron.info/category/teaching.html>. Accessed: 2020-06-22.
- [26] Friedrich Pfeiffer and Christoph Glocker. *Multibody dynamics with unilateral contacts*. John Wiley & Sons, 1996.
- [27] Abhinandan Jain. *Robot and multibody dynamics: analysis and algorithms*. Springer Science & Business Media, 2010.
- [28] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.
- [29] John YS Luh, Michael W Walker, and Richard PC Paul. On-line computational scheme for mechanical manipulators. 1980.
- [30] John M Hollerbach. A recursive lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. *IEEE Transactions on Systems, Man, and Cybernetics*, 10(11):730–736, 1980.
- [31] Elena Garcia, Joaquin Estremera, and Pablo Gonzalez-de Santos. A classification of stability margins for walking robots. *Robotica*, 20(6):595–606, 2002.
- [32] Philippe Sardain and Guy Bessonnet. Forces acting on a biped robot. center of pressure-zero moment point. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 34(5):630–637, 2004.

- 
- [33] Miomir Vukobratović and J Stepanenko. On the stability of anthropomorphic systems. *Mathematical biosciences*, 15(1-2):1–37, 1972.
  - [34] Miomir Vukobratović and Branislav Borovac. Zero-moment point—thirty five years of its life. *International journal of humanoid robotics*, 1(01):157–173, 2004.
  - [35] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, jan 1966. doi: 10.1080/00207176608921369.
  - [36] Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913. IEEE, 2012.
  - [37] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE, 2014.
  - [38] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.
  - [39] Donald E Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2004.
  - [40] Russ Tedrake. Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation (course notes for mit 6.832). <http://underactuated.mit.edu/>. Accessed: 2020-06-01.
  - [41] Morton I Kamien and Nancy Lou Schwartz. *Dynamic optimization: the calculus of variations and optimal control in economics and management*. Courier Corporation, 2012.
  - [42] Firdaus E Udwardia and Robert E Kalaba. A new perspective on constrained motion. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1906):407–410, 1992.
  - [43] Layale Saab, Oscar E Ramos, François Keith, Nicolas Mansard, Philippe Soueres, and Jean-Yves Fourquet. Dynamic whole-body motion generation under rigid contacts and other unilateral constraints. *IEEE Transactions on Robotics*, 29(2):346–362, 2013.