Title
Locomotion Analysis and Control of a Miniature Bipedal Robot

Author
Shen, Junjie

Publication Date
2022

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Locomotion Analysis and Control of a Miniature Bipedal Robot

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in Mechanical Engineering

by

Junjie Shen

2022

ABSTRACT OF THE DISSERTATION

Locomotion Analysis and Control of a Miniature Bipedal Robot

by

Junjie Shen

Doctor of Philosophy in Mechanical Engineering

University of California, Los Angeles, 2022

Professor Dennis Hong, Chair

From the dawn of time humans have been trying to recreate themselves with the technology of each age. What is impressive about humans is their capabilities of navigating the world and manipulating the world around them. It is these abilities that we wish to replicate in machines which can potentially overturn our daily lives. The recent advancements of robotics allow us to get closer than ever before to this realm of fantasy. For decades we have seen the incredible performance of automation robots in the factories, but we still have yet to see humanoid robots be utilized to aid humans in the everyday world. This is due in large part to the fact that the approaches to effectively navigating these two environments are different. The automation robots operate in the specifically structured environments. They are usually fully actuated with their bases being fixed, which grants them full control authority at all times. Simple control strategies are adequate under these conditions. Unfortunately, this is not the case with the humanoid robots in the real world.

Significantly faster development in quadruped robots than humanoid robots has been witnessed as recently there has been a large surge in the number of quadruped robots available

for commercial use. Compared to quadruped robots, humanoid robots are typically more mechanically complex and intrinsically unstable. This poses two critical challenges in the study of humanoid robots. First, accessibility to the physical hardware is limited as either it takes too much effort to develop a humanoid robot platform independently, or the commercially available ones, if any, are too expensive to afford. Second, as the humanoid robot system is considerably more challenging than the quadruped robot system, more advanced and efficient control algorithms are essential, especially when it comes to robust bipedal locomotion which is the fundamental capability of humanoid robots.

This dissertation aims at addressing some of the problems in these challenges. First, a next-generation miniature bipedal robot with proprioceptive actuation capable of dynamic behaviors is being under development. The robot is named BRUCE – Bipedal Robot Unit with Compliance Enhanced. The BRUCE robot is desired to serve as an accessible and reliable humanoid robot platform for general research purposes. It is expected to become open-source with an affordable price for the robotics community in the near future. Second, a state-of-the-art dynamic bipedal locomotion control framework is being studied. The approach is general and versatile as it is able to achieve a strong robustness of stabilizing a wide range of dynamic bipedal locomotion gaits including walking, running, and hopping. The effectiveness of the locomotion control framework was validated on the BRUCE robot both in simulation and with physical hardware.

The dissertation of Junjie Shen is approved.

Jason Speyer

Tetsuya Iwasaki

Jonathan Hopkins

Dennis Hong, Committee Chair

University of California, Los Angeles

2022

To my beloved family, friends, and ...

robots.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathbb{R}^n$	space of n-dimensional real vectors
$\mathbb{R}^{m\times n}$	space of $m \times n$ real matrices
$\mathsf{GL}(n, \mathbb{R})$	general linear group in dimension n
$\mathsf{SO}(n)$	special orthogonal group in dimension n
$\mathfrak{so}(3)$	space of 3×3 skew-symmetric matrices
a	scalar in general, i.e., $a \in \mathbb{R}$
$\boldsymbol{a}$	vector in general, i.e., $\boldsymbol{a} \in \mathbb{R}^n$
$\boldsymbol{A}$	matrix in general, i.e., $\boldsymbol{A} \in \mathbb{R}^{m\times n}$
$\boldsymbol{R}$	rotation matrix, i.e., $\boldsymbol{R} \in \mathsf{SO}(n)$
$\mathbb{I}$	identity matrix

LIST OF ACRONYMS

BEAR	Back-drivable Electromechanical Actuator for Robotics
BRUCE	Bipedal Robot Unit with Compliance Enhanced
B.S.	Bachelor of Science
CAD	Computer-Aided Design
CF	Complementary Filter
CMM	Centroidal Momentum Matrix
CoM	Center of Mass
CoP	Center of Pressure
CP	Convex Program
CPU	Central Processing Unit
CRBA	Composite-Rigid-Body Algorithm
DCM	Divergent Component of Motion
DH	Denavit Hartenberg
DoF	Degree of Freedom
JI	Joint Institute
FD	Forward Dynamics
FK	Forward Kinematics
HZD	Hybrid Zero Dynamics
ID	Inverse Dynamics
IK	Inverse Kinematics
IVP	Initial Value Problem
IMU	Inertial Measurement Unit
KBBQ	Korean Barbecue
KF	Kalman Filter
LIP	Linear Inverted Pendulum

LQR	Linear Quadratic Regulator
MPC	Model-Predictive Control
M.S.	Master of Science
NLP	Nonlinear Program
OCP	Optimal Control Problem
Ph.D.	Doctor of Philosophy
QP	Quadratic Program
RNEA	Recursive Newton-Euler Algorithm
RoMeLa	Robotics and Mechanisms Laboratory
SJTU	Shanghai Jiao Tong University
SLIP	Spring-Loaded Inverted Pendulum
SRB	Single Rigid Body
TO	Trajectory Optimization
UCLA	University of California, Los Angeles
UM	University of Michigan
WBC	Whole-Body Control
ZMP	Zero-Moment Point

ACKNOWLEDGMENTS

「Thank You」 to:

☺ My advisor, Prof. Dennis Hong, for his unconditional support during my Ph.D. journey at UCLA. To tell the truth, he was the main reason why I chose to come to UCLA and I am glad that I made the right decision back then. His guidance over the years has surpassed everything I could ever ask for. From him I saw the values of positive energy, enthusiasm, curiosity, imagination, open-mindedness and many more, not only in scientific research but also in general life. It was these pearls that helped me navigate through the many ups and downs of my Ph.D. life and most importantly, become an overall better person!

☺ My committee members, Prof. Jason Speyer, Prof. Tetsuya Iwasaki, and Prof. Jonathan Hopkins who I respect as researchers, teachers, and mentors, for taking the time to guide me throughout my Ph.D. adventure and provide thoughtful feedback on this dissertation. I truly have learned a lot from them!

☺ My second home, RoMeLa, where I met a group of remarkable people. I was fortunate to be a part of one of the greatest robotics research laboratories in the world. I would like to express my gratitude to everyone who was a part of RoMeLa for helping me not only as a colleague but also as a friend. Of course I will never forget our KBBQ nights!

☺ My friends and family, especially mom and dad for always being there for me during my peaks and valleys!

☺ My BRUCE team, Aaron, Nick, Hawkry, Tym, and Min, because this dissertation is impossible without their help. I cannot do it all by myself!

VITA

2016 – 2018 M.S. in Mechanical Engineering, UCLA, California, United States.

2012 – 2016 B.S. in Mechanical Engineering, UM-SJTU JI, SJTU, Shanghai, China.

2019 – 2022 Teaching Fellow, UCLA, California, United States.

2018 – 2019 Teaching Associate, UCLA, California, United States.

2017 – 2018 Teaching Assistant, UCLA, California, United States.

2015 – 2016 Teaching Assistant, UM-SJTU JI, SJTU, Shanghai, China.

PUBLICATIONS

11. **Junjie Shen** and Dennis Hong, "Divergent Component of Motion Based Analysis and Control of Dynamic Bipedal Locomotion Using Reduced-Order Models," 2022.

10. **Junjie Shen**, Jingwen Zhang, Yeting Liu, and Dennis Hong, "Implementation of a Robust Dynamic Walking Controller on a Miniature Bipedal Robot with Proprioceptive Actuation," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots*, 2022, pp. 39 – 46.

9. **Junjie Shen** and Dennis Hong, "Model Predictive Control Using Dynamic Model Decomposition Applied to Two-Wheeled Inverted Pendulum Mobile Robot," in *2022 19th International Conference on Ubiquitous Robots*, 2022, pp. 332 – 337.

8. **Junjie Shen** and Dennis Hong, "Convex Model Predictive Control of Single Rigid Body Model on SO(3) for Versatile Dynamic Legged Motions," in *2022 IEEE International Conference on Robotics and Automation*, 2022, pp. 6586 – 6592.

7. Yeting Liu, **Junjie Shen**, Jingwen Zhang, Xiaoguang Zhang, Taoyuanmin Zhu, and Dennis Hong, "Design and Control of a Miniature Bipedal Robot with Proprioceptive Actuation for Dynamic Behaviors," in *2022 IEEE International Conference on Robotics and Automation*, 2022, pp. 8547 – 8553.

6. **Junjie Shen** and Dennis Hong, "A Novel Model Predictive Control Framework Using Dynamic Model Decomposition Applied to Dynamic Legged Locomotion," in *2021 IEEE International Conference on Robotics and Automation*, 2021, pp. 4926 – 4932.

5. **Junjie Shen** and Dennis Hong, "Optimal Linearization via Quadratic Programming," in *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4572 – 4579, July 2020, and presented at *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

4. **Junjie Shen**, Yeting Liu, Xiaoguang Zhang, and Dennis Hong, "Optimized Jumping of an Articulated Robotic Leg," in *2020 17th International Conference on Ubiquitous Robots*, 2020, pp. 205 – 212.

3. Jingwen Zhang, **Junjie Shen**, and Dennis Hong, "Kinematic Analysis and Design Optimization for a Reduced-DoF Quadruped Robot with Minimal Torque Requirements," in *2020 17th International Conference on Ubiquitous Robots*, 2020, pp. 198 – 204.

2. **Junjie Shen** and Dennis Hong, "OmBURo: A Novel Unicycle Robot with Active Omnidirectional Wheel," in *2020 IEEE International Conference on Robotics and Automation*, 2020, pp. 8237 – 8243.

1. Xuan Lin, Jingwen Zhang, **Junjie Shen**, Gabriel Fernandez, and Dennis Hong, "Optimization Based Motion Planning for Multi-Limbed Vertical Climbing Robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1918 – 1925.

Chapter 1

Introduction

1.1 Motivation

Recently, the topic of humanoid robots has been placed under the spotlight. CyberOne [1], Xiaomi's first humanoid robot, debuted at the company's new product launch event in Beijing on August 11$^{\text{th}}$, 2022. Just a month later, Optimus [2] was unveiled at Tesla AI Day 2022 on September 30$^{\text{th}}$. There are many other remarkable humanoid robots which we are already familiar with, e.g., ASIMO [3] from Honda, Digit [4] from Agility Robotics, and Atlas [5] from Boston Dynamics. A lot of companies, domestic or foreign, giants or start-ups, high-tech or even low-tech, have been working on humanoid robots or putting it on the schedule. As humanoid robots are attracting more and more attention in industry (not to mention academia), debates on humanoid robots surge in the meantime. Some of the most typical ones are listed as follows with my personal thoughts:

- First of all, *why robots at first place?*

  In the past process of civilization, we human beings have clearly realized our physiological limits, e.g., humans cannot endure high or low temperature, hear ultrasound or infrasound, see infrared or ultraviolet, not to mention limited working intensity, efficiency, and duration. In order to make up for the deficiency, we have been building

a wide variety of tools, devices, and machines to accomplish the tasks which we do not want to or even cannot engage, e.g., repetitive, intensive, and dangerous tasks. For an easy task, a simple tool can suffice. However, for a complicated one, we might need an advanced machine to resolve and this is where robots come into play. A robot, in general, is an advanced machine that can sense, plan, and act. For decades we have seen robots have incredible success in industrial applications, e.g., health care, agriculture, food preparation, manufacturing, and military. They are capable of accomplishing these jobs with high endurance, speed, and precision. There is no doubt that the robotic technologies have impacted our lives in countless ways.

- Then *why humanoid robots?*

  There are many advantages of the human form:

  1. Although we humans may not be the best in every aspect, e.g., we cannot run as fast as cheetahs, climb as agile as monkeys, swim as fluid as fishes, we are general and versatile. We can pretty much go anywhere and do anything. Integration of different functions is always a great advantage for any products. Just like the Ultimate Orb in Dota whose attributes grant the same bonuses as those of ten Iron Branches, it is much more expensive but saves the item slots ☺. Just like the smartphone in everyday life, it combines tons of tools and devices into one unit. It is not only a mobile telephone but also a mini computer, a camera, a wallet, etc. Human life has completely changed ever since the creation of smartphone and I believe humanoid robots have the potential to overturn it again.

  2. If we want to have robot operate around/for us in human environments, it would be convenient to have the human form. There is no denying that the entire world is specifically designed for us humans, e.g., the door size, the stair height, all the tools we use, the devices, the facilities. If it is of the human form, we just need to have the robot adapt to it. If not, however, we might need to recreate all of

these for robot use only, which can take too much effort.

3. Humanoid robots are expected to have a close relationship with human beings in the everyday world, and psychologically, the human shape looks more approachable as living/working partners.

- Given the large advancements in robotics, *why is it still difficult to develop humanoid robots?*

  There are many difficult problems in developing humanoid robots as it is inherently cross-disciplinary, e.g., machine vision, perception, learning, artificial intelligence, energy source, just to name a few, which are beyond the scope of this dissertation. Mobility is another tough challenge and locomotion is the fundamental. The robot must integrate the sensor inputs, make an environmental model, interact with the environment, plan its movements, and then send commands to its actuators.

  The robots can perform greatly in the factories but not in the everyday world. This is due in large part to the fact that the approaches to effectively navigating the two environments are different. In the factories:

  1. The environment is structured and controlled.
  2. The robots are typically fully actuated with fixed bases, e.g., robotic manipulators, which grants them full control authority at all times.
  3. The robot states can be directly measured with high accuracy.
  4. Compliant interaction with the environment is usually not required and instead, high precision, stiffness, and strength are preferred.
  5. Simple control strategies can be used under these conditions.

  Unfortunately, this is not the case with the legged/humanoid robots in the real world:

  1. The environment is unstructured and can be changing.

2. By removing the fixed contact to the ground the system can become underactuated with a floating base, which limits the control authority.

3. With the additional floating-base joint, the robot states now can no longer be directly accessed, but instead sensor fusion and advanced filtering techniques are required.

4. For legged/humanoid robots, when a foot makes contact with the ground a collision occurs imparting an impulse to the system. These collisions must be effectively mitigated for the sake of system stability. However, hardware-wise, the common actuators for the robots in the factories do not incorporate compliance and thus are not capable of handling the impacts.

5. For legged/humanoid robots, the control of the robot motion is now done mainly through regulating the contact forces at changing contact locations. These contact forces are strongly restricted and thus need to be carefully planned to achieve the desired behavior. In addition, the system is now no longer continuous since as the robot travels different feet are in contact with the environment at different times. Under these circumstances, simple control approaches are no longer feasible and instead advanced motion planning and control techniques are required in order to account for the complex system dynamics.

Significantly faster development in quadruped robots than humanoid robots has been witnessed recently in terms of locomotion capability. Compared to quadruped robots, humanoid robots are typically more mechanically complex, requiring more powerful actuators and more degrees of freedom, as well as being more intrinsically unstable. This poses two critical challenges in the study of humanoid robots:

1. Accessibility to the hardware platform is limited as either it takes too much effort to develop a humanoid robot platform independently or the commercially available ones, if any, are too expensive to afford. As we know, one of the biggest issues in developing

robotics is the difference between the simulation and the real world, i.e., what works in the simulation will likely not be reflected exactly in the real world. It is therefore necessary to perform meticulous testing on the physical robot in order to adjust the system to function as desired. After all, a perfect robot in the simulated world is no use for us in the real world.

2. As the humanoid robot system is highly nonlinear and complex, nominally underactuated and unstable, multi-input and multi-output, as well as time-variant and hybrid (considerably more challenging than the quadruped robot system), more advanced and efficient control algorithms are essential especially for robust bipedal locomotion.

This dissertation aims at addressing some of the problems in these challenges. Particularly, first, a next-generation miniature bipedal robot capable of dynamic behaviors is being under development for general research purposes. The robot is named BRUCE – Bipedal Robot Unit with Compliance Enhanced. It is expected to become open-source with an affordable price for the robotics community in the near future. Second, a state-of-the-art dynamic bipedal locomotion control framework is being studied. The approach is general and versatile as it intends to achieve a strong robustness of stabilizing a wide range of bipedal locomotion gaits including walking, running, and hopping.

1.2 Background

A humanoid robot is a robot resembling the human body in shape. In general, humanoid robots have a torso, a head, two arms, and two legs, though some humanoid robots may replicate only part of the human body for different purposes, e.g., bipedal robots with only a torso and two legs for the study of bipedal locomotion.

1.2.1 A Brief History of Humanoid Robots

Automata

Automata are known as the early ancestors of robots but technically they are not robots, as they can only act but cannot sense and plan. The word "automaton" is the latinization of the Ancient Greek "αὐτόματον", which means acting of one's own will. It is more often used to describe relatively self-operating machines, especially those made to resemble human or animal actions, which can automatically follow a sequence of operations or respond to predetermined instructions.

The idea of humanoid automata originated in many ancient civilizations including Ancient Greece and Ancient China. In various ancient Greek mythologies, the Greek god of blacksmiths, *Hephaestus*, created several different humanoid automata which can speak and work for him [6]. In the 3rd century B.C., the Liezi, a Daoist text attributed to Chinese philosopher *Lie Yukou*, includes mention of an engineer named *Yan Shi* who created a life-size, human-like automaton for the emperor *King Mu of Zhou* [7]. It was primarily made of leather and wood, capable of singing and dancing. In the 13th century, a Muslim engineer named *Ismail al-Jazari* designed various humanoid automata [8]. One of them was a waitress that could serve drinks. In the 15th century, *Leonardo da Vinci* designed a humanoid automaton clad in a suit of armor, capable of sitting, standing, raising its visor, and independently manoeuvring its arms [9]. The entire system was operated by a series of pulleys and cables. In the 18th century, French inventor *Jacques de Vaucanson* built The Flute Player,

a wooden, human-sized automaton capable of playing various melodies with the flute [10]. It consisted of a system of bellows, pipes, weights, and other mechanical components to simulate the muscles necessary to play the flute. In the 19th century, Japanese craftsman *Hisashige Tanaka* was developing perhaps the first mobile humanoid automaton, Karakuri puppet, a tea carrying doll [11]. When the spring is wound, it "walks" forward with a cup of tea while bowing its head.

These examples are just the tip of the iceberg, not to mention the unimaginable undocumented ones. It is interesting to see that from the dawn of time humans have been interested in recreating themselves with the technology of each age.

Early Robots

The term "robot" was first used to denote a fictional humanoid robot in the play R.U.R. (Rossum's Universal Robots) by Czech playwright *Karl Čapek* in 1920. In the play, artificial people were built to work on factory assembly lines but eventually rebelled against their human masters. According to *Karl*, the word was actually created by his brother *Josef Čapek* from the Czech word "robota" which means forced labor.

In 1927, Japan's first humanoid "robot", *Gakutensoku*, was designed and constructed by biologist *Makoto Nishimura* [12]. It was actuated by compressed air, could write fluidly and raise its eyelids. *Gakutensoku* means "learning from the laws of nature" and was not designed as a laborer but to think, write, and entertain.

In 1928, the first British humanoid "robot", *Eric*, was exhibited at the annual exhibition of the Model Engineers Society in London [13]. At the event's opening, *Eric* rose to his feet, bowed, and gave a four-minute opening address. Invented by *William Richards* and *Alan Reffell*, the robot's frame consisted of an aluminium body of armor with eleven electromagnets and one motor powered by a 12 V power source. The robot could move its hands and head, and could be controlled through remote control or voice control.

In 1939, the humanoid "robot" known as *Elektro*, built by the Westinghouse Electric

Corporation, appeared at the World's Fair in New York [14]. Standing 2.1 meters tall and weighing 120 kilograms, it could walk by voice command, speak about 700 words, smoke cigarettes, blow up balloons, and move its head and arms. The body consisted of a steel gear, cam, and motor skeleton covered by an aluminium skin. Its photoelectric "eyes" could distinguish red and green light.

Though these nearly modern robotic devices are equipped with more advanced hardware structures and control mechanisms, capable of more complicated task behaviors, they are still considered more as automata than as robots because at its most basic, they lack the sense of autonomy.

Modern Autonomous Robots

It was not until 1942 that science fiction writer *Isaac Asimov* coined the word "robotics" in his short story "Runaround" and formulated the Three Laws of Robotics [15]:

- First Law - A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- Second Law - A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
- Third Law - A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

In 1954, the world's first digitally operated and programmable robot, *Unimate*, was invented by *George Devol*, which represents the foundation of the modern robotics industry [16]. The machine undertook the job of transporting die castings from an assembly line and welding these parts on auto bodies, a dangerous task for workers who might be poisoned by toxic fumes or lose a limb if they were not careful. Since then, industrial robots have been refined and expanded, capable of performing a variety of other tasks. Additionally,

robots have made their way out of manufacturing and into other industries such as medicine, agriculture, entertainment, and space exploration.

In 1966, Professor *Ichiro Kato* carried out the first studies concerning humanoid robots at Waseda University in Japan, with the artificial lower limb WL-1 and upper limb WAM-1. This work later led to WAP-1, 2, 3, a series of biped walking robots. In 1973, he introduced the first full-scale anthropomorphic robot developed in the world, WABOT-1, which walked with its lower limbs, and was able to grip and transport objects with hands which used tactile-sensors [17]. A number of other humanoid and bipedal robots capable of stable walking were built afterwards, e.g., WABIAN series [18].

Since the advent of WABOT-1, robot fever has broken out all over the world. In particular, this work pioneered the field of early anthropomorphic robots, the legs of which have a minimum of six degrees of freedom (DoF), i.e., a 3-DoF hip to simulate a spherical joint, one DoF for knee bending, and two DoFs for an ankle ball joint, in order for the feet to be able to achieve arbitrary 3D configurations in terms of both position and orientation. Prominent examples of such robots are ASIMO [3], HRP [19], iCub [20], NAO [21], DARwIn-OP [22], and TALOS [23]. These robots usually utilize rotary joints with servomotors in a serial configuration. Its strength lies in the simplified structure and analysis, which, however, comes at the drawback of limited capabilities, e.g., accumulation of backlash or elasticity from each joint, increased leg weight and inertia [24].

As the study of humanoid robots matures, i.e., they become capable of more dynamic motions, the effect of decreased joint rigidity and increased leg inertia can be substantial, which at the time was not necessarily appreciated because the robots moved so slow that the dynamics could be fairly neglected. There is no denying that legs play a critical role in humanoid dynamic motions. For example, when it comes to robust locomotion, even slower, less dynamic robots still need to move their legs quickly and accurately enough to reposition their feet in order to maintain balance. And in the meantime, hopefully, the fast acceleration of the leg motion will not influence the overall dynamics too much,

which can possibly deteriorate the operating condition. Therefore, nowadays, the popular trend on humanoid robot platform is lightweight legs with small feet, which lowers the leg inertia and, in the process, reduces the control efforts. To achieve this paradigm in practice, numerous approaches have been studied. The most prevalent option is to place the actuators off-axis, as close to the torso as possible, as opposed to collocating them at each joint. However, off-axis actuation requires the introduction of power transmission mechanisms, e.g., belts and linkages, which typically trade off between range of motion, joint backlash, rigidity, and complexity [25]. Placing the actuators off-axis also leads to interesting joint designs where parallel configurations can be involved, which exhibit higher stiffness due to the mechanical drive coupling but can suffer the common issues of limited workspace and increased complexity [24]. As a result, several compact and elegant solutions have been produced, including Lola [26], CHARLI [27], SAFFiR [28], TORO [29], Valkyrie [30], ATRIAS [31], WALK-MAN [32], LARMbot [33], RHP2 [34], NimbRo [35], Hybrid Leg [36], and CogIMon [37]. Apart from joint-specific solutions, other main approaches to leg weight reduction include topology-optimized structural components with high strength-to-weight ratios (e.g., Atlas [5], BRUCE [38]), biomimetic humanoid design (e.g., Cassie [4], Digit [4], BioBiped [39], Kenshiro [40], Kengoro [41], CARL [42], eLeg [43]), utilizing light and strong materials such as carbon fiber, aluminum, and titanium (e.g., NABiRoS [44], NABi-V2 [45]).

Humanoid robots have the potential to exert a much larger influence on humanity in the future. While the world has seen large advancements in the field of humanoid robots, humanoid robots are still only beginning to fulfill this great potential. One final phenomenon that has just begun in the past couple of years is the commercial humanoid industry. With the growing robustness, maturity, and ubiquity of humanoid robotics technology, more and more full-size humanoid robots becomes commercially available, e.g., ASIMO [3], Digit [4], Valkyrie [30], NimbRo [35], TALOS [23]. Though the price of a full humanoid robot platform becomes more and more affordable, it is still enormous for personal use, e.g., Digit [4] costs

over 250,000 USD per unit. Currently, the main use of commercial humanoid robots is still for research, entertainment, and demonstrative purposes, same as the old days. We still have yet to see humanoid robots be utilized to aid humans in any industries. It will be interesting to see the further development in the field of humanoid robots and its impact on human society in the coming years! ☺

1.2.2 Sensing

Similar to humans, humanoid robots require the sense of self-movement and environment awareness as the first of the three primitives of robots (in advance of plan and act). Over the years, multiple sensing technologies have been developed for modern autonomous robots, which can be categorized into two major types as follows.

Proprioceptive Sensors

Proprioceptive sensors measure values internal to the robot system [46]. For example, an inertial measurement unit (IMU), which is composed of an accelerometer, gyroscope, and magnetometer, is typically used for estimation of the robot body position and orientation. The IMU information is often fused with additional information coming from the joint sensors for improved accuracy. Joint position measurement is usually done with high-resolution, hall-effect-based, magnetic absolute encoders, from which joint velocity can be calculated by straightforward differentiation. Joint force or torque can be sensed by measuring the motor current or fluid pressure for hydraulic/pneumatic actuation. In combination with a Jacobian, end-effector wrenches can be estimated roughly. However, fine manipulation relies on direct force/torque (F/T) sensor feedback.

Exteroceptive Sensors

Exteroceptive sensors acquire information from the robot's environment [46]. The information is then interpreted by the robot in order to extract meaningful environmental fea-

tures. Typical examples are vision sensors (cameras as the robot "eyes"), sound sensors (for speaking and hearing), tactile sensors (robot "skin" for meticulous interaction with the environment [47, 48], e.g., through skin humans can sense various physical quantities such as temperature, humidity, and pressure), range sensors (for obstacle detection without physical contact), etc.

1.2.3 Actuation

The structure of a humanoid robot is often co-dependent on the selected actuation strategy which is one of the most crucial aspects of the entire system (as it is responsible for the motions of the robot). While some new types of soft actuators mimicking biological muscles show great potentials [41, 49], the technology is not mature enough at this point. Modern electric, hydraulic, and pneumatic actuators are still the top choices for most humanoid systems given their robust performance.

Electric Actuators

Electric actuators have long been the default choice of actuation in many applications including robotics, as they offer a good trade-off between torque, speed, and size. Using electric actuators also simplifies the control aspect due to the nearly linear input-to-output (i.e., current-to-torque) relationship. In addition, power storage and distribution can be convenient and clean.

Conventional humanoid robots used brushless direct-current (BLDC) motors with a large gear reduction ratio in order to achieve high torque density at the cost of low speed. Nevertheless, these heavily geared motors, specifically designed to perform accurate, position-controlled tasks in fairly structured environments, are quite vulnerable when faced with significant impacts (e.g., foot strike during locomotion), due to the increased reflected inertia and gear friction from the gearbox [50]. The layout is a relic of classic industrial manipulators and is not amenable to modern humanoid robots as they are required to interact more

and more with their environments.

The issue can be addressed with the help of series elastic actuators (SEA) by intentionally adding controlled variable mechanical impedance in series with the motor [51]. The elastic element enables SEA to absorb impacts, store energy, and control the output torque accurately through deflection using Hooke's law. However, the force bandwidth of SEA is limited, i.e., limited actuation speed and operating bandwidth as determined by the stiffness of the elastic element [52]. In addition, these robotic systems with added mechanical impedance usually result in complex dynamics, making them difficult to control at best and restricted in their capabilities at worst.

Lately, impressive advances in electromagnetic technology lead to a new approach, proprioceptive actuators, which are essentially high-torque-density (due to a large gap radius) BLDC motors with a small amount of gearing. Based on the amount of gear reduction, proprioceptive actuators can be classified as direct-drive motors (with no gear reduction) [50] and quasi-direct-drive motors (with little gear reduction) [53, 54, 55]. The main advantage of these actuators, besides impact mitigation, is high transparency (in terms of power transmission) and mechanical performance [50], i.e., it is possible to sense and control the output torque directly through the current. The main drawback of proprioceptive actuators is their high heat production caused by running the large BLDC motors at low speed, meaning additional design considerations need to be made for systems requiring high continuous torque output, e.g., a cooling system [55, 56].

Hydraulic Actuators

Electric actuators are still limited in high-power applications due to magnetic saturation and thermal requirement. Hydraulic actuators, in that aspect, can offer a much larger power density. They are usually produced as pistons, where linear motion is performed by controlling the pressure difference of incompressible liquid (usually oil) between two chambers separated by a movable cylinder. The price to pay is mainly the leakage problem, bulky size

and loud noise of the pump. Recently, these limitations have been resolved by miniaturizing the pump size and integrating the hydraulic system in the robot at the design level as it is the case for Atlas [5] and BigDog [57]. In spite of these achievements, only a few groups other than Boston Dynamics have been successful in implementing mobile hydraulic actuation on humanoid robots due to its complexity, e.g., Hydra [58], TaeMu [59].

Pneumatic Actuators

Pneumatic actuators operate on the basis of gas compressibility (usually air). Several types of pneumatic actuators, e.g. cylinders, bellows, pneumatic engines, pneumatic stepper motors, and pneumatic artificial muscles (in fact inverse bellows), are commonly used to date [60]. The major attractions about pneumatics are the low weight and the inherent compliant behavior. Pneumatic actuation has some important drawbacks making it less popular though, mainly with regard to its control (due to system nonlinearity), inflating speed (i.e., slow response), embeddability, and service life [60, 61]. Some examples of robots with pneumatic actuation are available, e.g., Lucy [62], Mowgli [63].

1.2.4 Locomotion

Locomotion is the fundamental of any types of mobile robots. As for humanoid robots, bipedal locomotion has been studied for decades and yet it remains an active research field. The first control paradigm for bipedal locomotion used the notion of static walking [18], where the center of mass (CoM) projection on the ground is always contained within the support polygon of the feet. Ever since locomotion on bipedal robots has become more and more dynamic, which results from continued progress in three key aspects: the mathematical understanding of locomotion, the computational ability to encode this mathematics, and the hardware capable of realizing this understanding in practice [64]. The hardware perspective has been contextualized previously. In terms of control, mathematical modeling of locomotion captures the essential walking behaviors at first place, which forms the basis for gait

generation. These generated motions are then tracked by real-time feedback control.

Mathematical Modeling

The mathematical modeling of the bipedal system can be divided into two categories roughly, full-order model and reduced-order model. The full-order model is concrete but sophisticated as it exploits every single detail of the robot. The reduced-order model is abstract but simple as it only focuses on the most salient aspect of the system dynamics.

- Floating-Base System

  Bipedal robots can be modeled as a "tree" of rigid bodies, one of which can serve as the root/base (usually pelvis or torso). The structure lends itself well to the generalization of efficient kinematics and dynamics algorithms for different configurations [65]. Because interactions with the environment are always changing during locomotion, a convenient way of modeling the system is to construct a general representation of the robot floating in the air, and then enforce ground contacts through forces arising from the associated holonomic constraints which are imposed at the stance feet. This is thus often referred to as the floating-base model of the robot. The configuration space involves the position and orientation of the floating base, as well as the local coordinates dictating the shape of the robot (e.g., revolute joint angles and prismatic joint displacements). The resultant floating-base dynamics is complicated as it is nonlinear and high-dimensional, nominally underactuated and unstable, multi-input and multi-output, as well as time-variant and hybrid [66].

- Centroidal Momentum Model

  For the sake of computational efficiency, the full floating-base dynamics can be simplified. The first working assumption is sufficient torque authority and kinematic reachability, i.e, the robot always has enough joint torques to realize the actuated part of the dynamics as long as the motion is kinematically feasible. As a result, we can just

focus on the six unactuated coordinates of the floating base, the dynamics of which is equivalent to the Newton-Euler equations. This model is known as centroidal momentum model as it relates the rate of change in the linear and angular momentum to the net external wrench acting on the robot.

- Linear Inverted Pendulum Model

  Simple as centroidal momentum model, it is still nonlinear (due to the bilinear terms, e.g, the resultant moment of the external force is the cross product between its associated moment arm and the force itself). If we further assume constant angular momentum about the CoM (typically zero) as well as a linear CoM trajectory (typically at a fixed height) as observed in human walking on a flat ground [67], a linear expression for the robot dynamics can be derived. This linearized model is called linear inverted pendulum (LIP) model as a bipedal robot supporting itself on one leg during walking resembles an inverted pendulum [68]. The LIP can also be viewed as a cart–table system, where the cart–table lies on a base with a geometry corresponding to the support polygon [69].

- Variations on Inverted Pendulum Models

  In an attempt to overcome issues associated with the strict assumptions of the LIP, more complex pendulum models have been introduced. For example, inverted pendulum with a variable height has been studied to address the main drawback of a constant CoM height [70, 71]. To account for the inertia of a swing leg, its mass concentrated at its foot has been added to the LIP [72]. To make use of nontrivial angular momenta from the torso and limbs, a flywheel with centroidal moment of inertia and rotational angle limits has been investigated based on the LIP [73].

- Zero-Moment Point

  Zero-moment point (ZMP) represents the point on the ground at which the net moment of the inertial and gravity forces has no component tangential to the ground surface

[74]. If the ZMP approaches the edge of the support polygon, the robot would start to rotate about the foot edge. This led to perhaps the most commonly used dynamic stability margin, referred to as the ZMP criterion, which states that locomotion is stable as long as the ZMP remains within the support polygon. Though this notion is effective, it is conservative, producing unnatural walking, and controlling these motions typically requires the robot to remain fully actuated. The ZMP also coincides with the center of pressure (CoP) when the robot is in contact with a single surface [75], e.g., walking on a flat ground.

- Capturability

  When the robot starts to tip over, the ZMP does not exist and therefore the ZMP criterion becomes pointless. Capturability can serve as a more general stability condition. The idea of capture point (CP) has been introduced at the first place [73, 76]. It describes the point on the ground onto which the CoP has to reach for the robot to come to a complete rest, e.g., if the CP is within the support polygon, the robot can modulate its CoP (to the CP) to balance without taking a step; if the CP and the support polygon are disjoint, the robot must take a step (to cover the CP) to come to a stop. To account for the step length and step time constraints, the N-step capturability has been analyzed based on the CP – the ability of a robot to come to a stop without falling by taking N or fewer steps [77]. Capture regions can be generated accordingly which define appropriate foot placement to avoid falling.

- Divergent Component of Motion

  The dynamics of the LIP can be decomposed into two parts, one is stable and the other is unstable. This implies that controlling only the unstable part is enough to generate stable locomotion. The unstable part of the dynamics has been referred to as the divergent component of motion (DCM) [78], which acts as the CP of the LIP. Accordingly, the capturability-based analysis can be applied to the DCM directly. The

name of "DCM" is more commonly used when it comes to locomotion as the CP is originally proposed for push recovery (CP literally means the "point" where the robot can "capture" itself).

- Spring-Loaded Inverted Pendulum

  The spring-loaded inverted pendulum (SLIP) is a conceptual model for locomotion on compliant legs, originating from biological studies of animal locomotion (especially running) [79]. In the simplest form, it reduces the body to a point mass that rebounds on a massless spring leg in stance and moves on a ballistic trajectory in flight [80]. The SLIP is attractive due to its inherent energy efficiency and robustness to ground height variation, which serves as a natural starting point for many running robots in design and control [31, 81, 82, 83, 84, 85, 86]. Simple as the SLIP, its dynamics do not admit a closed-form solution, which limits its applications to some extent.

- Single Rigid Body Model

  If the leg of the robot is light enough, i.e., the mass/inertia ratio of the torso to the leg is sufficiently large, the leg dynamics will not significantly affect those of the torso and thus they can be neglected fairly. The resultant model is called single rigid body (SRB) model as it only considers a single rigid body (i.e., torso) subject to the contact forces at the feet [87]. Though the SRB model is popular for quadrupedal robots with lightweight leg design (it efficiently captures the effect of the net external wrench on the evolution of both robot torso position and orientation) [88, 89, 90, 91, 92, 93], it does not work well for most bipedal robots as their leg dynamics can hardly be neglected due to more DoFs and thus increased weight/inertia [94].

- Hybrid Zero Dynamics

  Instead of using approximated models, some researchers have adopted more complete paradigms for locomotion which consider nontrivial impacts at touchdown with the

robot full-order model. To deal formally with the nonlinearity, tools in nonlinear control theory have been introduced, zero dynamics in particular, which are the internal dynamics (if exist) of any system compatible with the output being identically equal to zero [95]. Hybrid zero dynamics (HZD) is the extension of zero dynamics to hybrid systems, e.g., walking includes both continuous (leg swing) and discrete (velocity jump due to foot strike) dynamics. The idea of HZD is to encode desired locomotion behaviors via a set of suitably parametrized virtual constraints, which effectively coordinate the higher-dimensional robot plant into a lower-dimensional hybrid subsystem that governs the stability properties of the robot's locomotion behavior [96].

Gait Generation

Throughout the mathematical modeling of locomotion, we can see how the problem is fundamentally different from the conventional textbook examples. Due to the inherent complexity, e.g., hybrid dynamics, high nonlinearity, strong restrictions, and considerable DoFs, almost all approaches to producing stable dynamic locomotion gaits (i.e., gait generation) must tackle the problem by means of motion planning. In order to properly handle the various constraints naturally imposed on the problem, it is typically transcribed into an optimal control problem (OCP), a control design process using mathematical optimization (see Appendix A). However, it is not possible to derive the analytical solution for general nonlinear systems and therefore the technique of trajectory optimization (TO) is widely adopted as a numerical approach, which computes an open-loop solution to the OCP (see Appendix B).

The TO of bipedal locomotion in its most complete form, i.e, optimizing over contact schedule and contact force simultaneously with the robot full-order model, is extremely challenging even offline:

1. Trajectories are infinite-dimensional, the first challenge of TO is to suitably discretize the space of trajectories in order for numerical optimization.

2. A second challenge is that the discretized optimization problem is usually also fairly

large depending on the system dimension and granularity of the discretization, and hence optimization may be computationally inefficient.

3. The optimization problem is highly nonlinear due mainly to the involvement of the sophisticated robot full-order model including both kinematics and dynamics. In particular, a nonlinear optimization problem is often formulated as a nonlinear program (NLP) which suffers the issues of computational complexity, initial guess, local minima, feasibility, etc. Therefore, it is imperative that the TO problem is conditioned appropriately to be used.

Due to the very limited computing power of early devices, this approach was still stuck with simple planar models by the end of 1990s [97, 98, 99, 100, 101]. It was not until the year of 2000 that the optimal walking motions for complete 3D models could be solved [102, 103, 104]. Fortunately, numerical methods have continued to improve, recently allowing the problem to be solved in real time [105, 106]. Nevertheless, the curse of the NLP still limits its performance, i.e., often at best a locally optimal solution to the OCP is provided (not even that any global optimum exists [107]) depending on the initial guess. Not being able to efficiently generate gait online limits the robots to a predefined set of precomputed actions, potentially ruining their versatility and reactivity in the presence of even small perturbations. One way to alleviate this serious limitation is to generate a database of trajectories (i.e., gait library) which can be queried online (or serve as the initial guess of the NLP), possibly conditioned on the commands sent to the robot and the current states of the robot in order to improve its locomotion stability [108, 109, 110, 111].

To reduce the computational burden of planning with robot full-order model, there exist a variety of methods using the reduced-order models. The introduction of the reduced-order models naturally breaks down the problem into multiple stages (typically the higher the level, the simpler the model and the longer the horizon) as they only focus on the most salient aspect of the system, e.g., CoM and ZMP, and therefore requires the low-level motion controller to track the high-level plan while handling other neglected behaviors, e.g., leg

swing and torso orientation. For the most typical model of walking, i.e., LIP, its linear dynamics of the restricted system often yield efficient motion planning approaches. One of the earliest and most representative method is the ZMP preview control [69], i.e, considering the jerk of the COM as an input, CoM and ZMP trajectories can be solved optimally and analytically for predefined footsteps using a preview control based on the linear quadratic regulator (LQR).

The significantly enhanced computational efficiency thereof leads to a marvelous idea: if the algorithms for motion planning could work sufficiently fast to be applied online, we can use them as a feedback control policy which is able to continuously adapt the motions to the current states of the robot and its environment [112]. This is what we called model-predictive control (MPC), also known as receding horizon control as the prediction horizon keeps being shifted forward. Using the idea of MPC, an improvement of the original ZMP preview control scheme [69] was proposed by simply running the algorithm online continuously, focusing on the problem of compensating real-time perturbations [113].

However, predefining the footholds is not desirable under strong disturbances and uncertainties. Fortunately, without introduing too much extra computation, the simplicity of the LIP additionally allows to plan over a horizon of multiple steps through a quadratic program (QP) while restricting the CoP within the support polygon [114, 115, 116]. In this vein, more progresses have been achieved to make full use of the LIP in the MPC. It has been noted that accurate modulation of the CoP is often limited [117], especially when the moving speed is fast, not to mention robots with a small footprint, e.g., point [118] or line foot [111], or limited ankle authority, e.g., deficient DoF [38] or passive joint [119, 120]. In this regard, we do not need to specifically sample in time to take into account variable CoP, at least for the high-level motion planning. Accordingly, with invariant CoP associated with the footstep location, the LIP dynamics can be integrated analytically so that only an order of magnitude less decision variables are involved [121, 122]. Note that a closed-form solution also exists for a linear CoP trajectory [123]. Nevertheless, in all these strategies,

phase duration is always fixed to maintain the convexity of the problem, even though optimizing both footstep location and phase duration can greatly improve the stability margin, as shown by some nonlinear approaches [124, 125]. Notably, the LIP dynamics can be decomposed into two parts, one is stable and the other is unstable, and controlling only the unstable part (i.e., DCM) is enough to generate stable walking [78, 126]. Unlike the full CoM states including both position and velocity, the DCM follows a first-order dynamics, which helps simplify other aspects of locomotion analysis and control. Based on the DCM, it was reported to successfully adapt both location and timing of the next step while keeping a convex TO problem thanks to a suitable change of variable, which leads to even less decision variables [119]. The promising idea has also been utilized and verified in later works, e.g., [127, 128, 129, 130, 131, 132].

Despite the ability of these algorithms to be implemented online, they do not consider the discrete dynamics associated with foot strike, ruling out the nontrivial impacts which are naturally associated with dynamic locomotion [133]. In addition, naively executing the reduced-order plan would limit the motion capability of the robot and additional details need to be separately designed on the low-level stage, e.g., swing-leg motion, which increase the overall system complexity. This could also possibly fight against the reduced-order plan as it is difficult to provide a priori guarantees on whether any given reduced-order plan is feasible to execute on the full-order model either in the near term or in the long horizon [134]. Recently, a new approach called model hierarchy predictive control (MHPC) has been introduced, which plans with the full-order model in the near horizon and the reduced-order model in the long horizon to benefit from both model accuracy and computational efficiency [135]. Nevertheless, nonlinear MHPC is still of considerable computational complexity and suffer the issues of the NLP due to the involvement of the intricate robot full-order model [135, 136]. In this regard, a convex formulation for MHPC was proposed, which can be solved efficiently and thus implemented in real time for feedback control at the cost of slightly reduced model accuracy [38].

Motion Control

Properly executing the locomotion gait generated previously requires the deployment of the real-time motion control. When the locomotion has been planned using the reduced-order models, the spatial geometry of the robot must be translated into joint space that can be controlled. In addition, the locomotion behaviors which are neglected by the reduced-order models must be taken into account. Even with the full-order hybrid model, controllers must be synthesized in order to track these desired motions in practice due to the inevitable disturbances and uncertainties.

The simplest control scheme for determining joint torques is proportional-derivative (PD) control on joint positions and velocities. The desired references can be obtained either from inverse kinematics (IK) for the reduced-order model or directly from the outputs of a TO using the full-order model. The strongest advantage of this method is the sheer simplicity of its implementation as well as its intuitive physical meaning with respect to tuning. For full-order TO, simply tracking the outputs with a well-tuned joint PD controller is sometimes sufficient to achieve stable walking [110, 111, 137, 138, 139, 140] and even running [141] on hardware. This is because the output trajectories implicitly encode the stability constraints with the robot full-order model information, even if achieving these behaviors requires slightly different torques on the actual robot.

Unfortunately, this is not the case with the reduced-order model as some aspects of the system are approximated or simply omitted in return for computational efficiency. For example, most reduced-order models typically consider a point-mass representation of the robot. As a result, other locomotion behaviors such as torso orientation, whole-body momentum, contact force regulation, and leg swing, become important concerns when developing feedback controllers for implementation of a low-dimensional plan.

For simultaneous execution of multiple tasks in the low level, whole-body control (WBC) is also a good candidate which exploits the full capabilities of the entire body of redundant, floating-based robots interacting with the environment. In addition, it is able to realize

fast, agile, and compliant motions yet without sacrificing accuracy. The WBC scheme typically uses inverse dynamics (ID) to handle contact force interactions with the environment [142, 143]. However, ID-based WBCs heavily depend on the high quality of the dynamic model which is often difficult to obtain in practice. On the contrary, the robot kinematic model is much easier to get with high accuracy. As a result, examples of the WBC also exist which are based instead on IK [144, 145] or the combination of both [120, 146]. In addition, early approaches usually considered trajectories in joint space, resulting in an incredible amount of required motion details [147, 148]. By contrast, task-space control eases this burden by designing trajectories in the intuitive task space [149, 150, 151]. However, for all these methods, contact stability and torque limits are not properly handled, the violation of which can easily cause poor performance or even control failure. Lately, an elegant way to implement WBC is using optimization, mostly QP (to exploit the fact that the instantaneous system dynamics can be expressed linearly with respect to a certain choice of decision variables), which is able to account for system dynamics, map between the task space and joint space, as well as satisfy various constraints concurrently. Depending on how the task hierarchy is managed, QP-based WBC can be mainly categorized into two types. A weighted WBC scheme sets all operational tasks as the objectives of a single QP with priorities implicitly being enforced with different weights [146, 152, 153]. A strictly hierarchical framework solves cascaded QPs from the highest priority to the lowest, where the subsequent QP is carried out with additional constraints to preserve the optimality of the previous one [154, 155, 156].

1.3 Organization

This dissertation contributes to the advancement of humanoid robots, with a focus on the humanoid robot system design, joint mechanisms, kinematics, bipedal locomotion, optimization and optimal control, motion planning and control. The remainder of this dissertation is organized as follows.

Chapter 2 introduces the *Bipedal Robot Unit with Compliance Enhanced* (BRUCE), a cost-effective miniature bipedal robot platform capable of dynamic behaviors. Details of the hardware design are first presented. Complete modeling and analysis of the robot system, e.g., kinematics and dynamics, are also carried out. Preliminary experiments are then conducted to verify the basic functionalities of the robot design as well as to explore its dynamic capabilities. The BRUCE project is aimed to provide accessible and reliable humanoid robot platforms for general research purposes.

Chapter 3 goes through the *Divergent Component of Motion Based Analysis and Control of Dynamic Bipedal Locomotion Using Reduced-Order Models*. The nominal gait pattern is first derived and the viability condition is also investigated based on the reduced-order models. The locomotion control is solving a small-scale QP with multiple steps being considered in advance, which optimally plans the step location and timing in real time. In a numerical simulation environment, the theoretical capabilities of the approach is verified independent of the actual robot.

Chapter 4 emphasizes some other key aspects of the entire robot system to properly apply the locomotion control strategy presented in Chapter 3 to a full humanoid robot, e.g., software architecture, state estimation, and low-level control. A full discussion on practical *Implementation Details* is also considered valuable to promote the research on humanoid robots as they are likely to be skipped in the literature. The effectiveness of the overall locomotion control framework is finally validated on the BRUCE robot introduced in Chapter 2 with an extensive series of successful simulation and hardware experiments. The experimental results demonstrate the strong robustness of stabilizing various dynamic

bipedal locomotion gaits including walking, running, and hopping.

Chapter 5 summarizes the work presented in this dissertation and discusses the potential future directions.

Some fundamental knowledges and tools are also provided in Appendix A, B, and C which can help better comprehend this dissertation.

Chapter 2

Bipedal Robot Unit with Compliance Enhanced

2.1 Introduction

2.1.1 Background

Ever since the creation of the first humanoid robot, WABOT-1 [17], humanoid robots have been developed in various aspects such as joint design with high precision and torque, integration of sensors and computer vision to sense the environment, as well as software and control algorithms. However, dynamic behaviors such as running and jumping remain challenging for humanoid robots. One of the main problems is the lack of capability to properly handle contact impact due to the high gear ratio of traditional servo motors. Series elastic actuators (SEA) provide a great potential solution to this, but they can still suffer from the low force control bandwidth [52]. Recently, proprioceptive actuation has exerted huge influence on robotics due to its impact mitigation and wide-bandwidth force control capabilities [54]. With proprioceptive actuators, highly dynamic motions have become feasible for quadruped robots [92, 157], but few examples exist for humanoid robots.

2.1.2 Motivation and Contribution

The main reason for the slower development in humanoid robots than quadruped robots in terms of dynamic behaviors lies not only on more complex control algorithms but also on the limited accessibility to hardware. Atlas [5] from Boston Dynamics is proved to be the most dynamic full-size humanoid robot with advanced control algorithms and state-of-the-art hardware. However, its accessibility is highly restricted. Digit [4] from Agility Robotics is also a great humanoid robot platform that addresses the mobility limitations of conventional humanoids, but it costs over $250,000$ USD per unit. In terms of accessibility, the small-size humanoid robots seem to be a more preferable option for research purposes. DarwIn-OP [22] by the Robotics and Mechanisms Laboratory (RoMeLa) has been a reliable open-platform humanoid robot due to its great performance and affordable price. Although traditional servo motors are used in DarwIn-OP for actuation, it still presents better dynamic performance than regular full-size humanoid robots thanks to its reduced size and moment of inertia. In order to promote highly dynamic motion development for humanoid robots, a miniature bipedal robot with proprioceptive actuation is thus desired.

As the study of humanoid robots becomes a worldwide interdisciplinary research field, the demand for a cost-effective bipedal robot system capable of dynamic behaviors is growing exponentially. Inspired by the accessibility and reliability of current small-size humanoid robot platforms, as well as the rising technology of proprioceptive actuation, the next-generation miniature Bipedal Robot Unit with Compliance Enhanced (BRUCE) has been under development recently, as shown in Figure 2.1. In order to perform human-like dynamic motions, the joint configuration and range of motion are designed to be as close as possible to those of human beings as seen in Figure 2.2. In addition, unlike the traditional humanoid robots whose actuators are directly located at each joint, a 2-DoF cable-driven differential pulley system and a 4-bar linkage mechanism are applied to the hip and ankle joints, respectively. By doing so, the moment of inertia of each leg is significantly reduced in favor of highly dynamic leg motions. Meanwhile, the choice of cable drive for the differential transmission

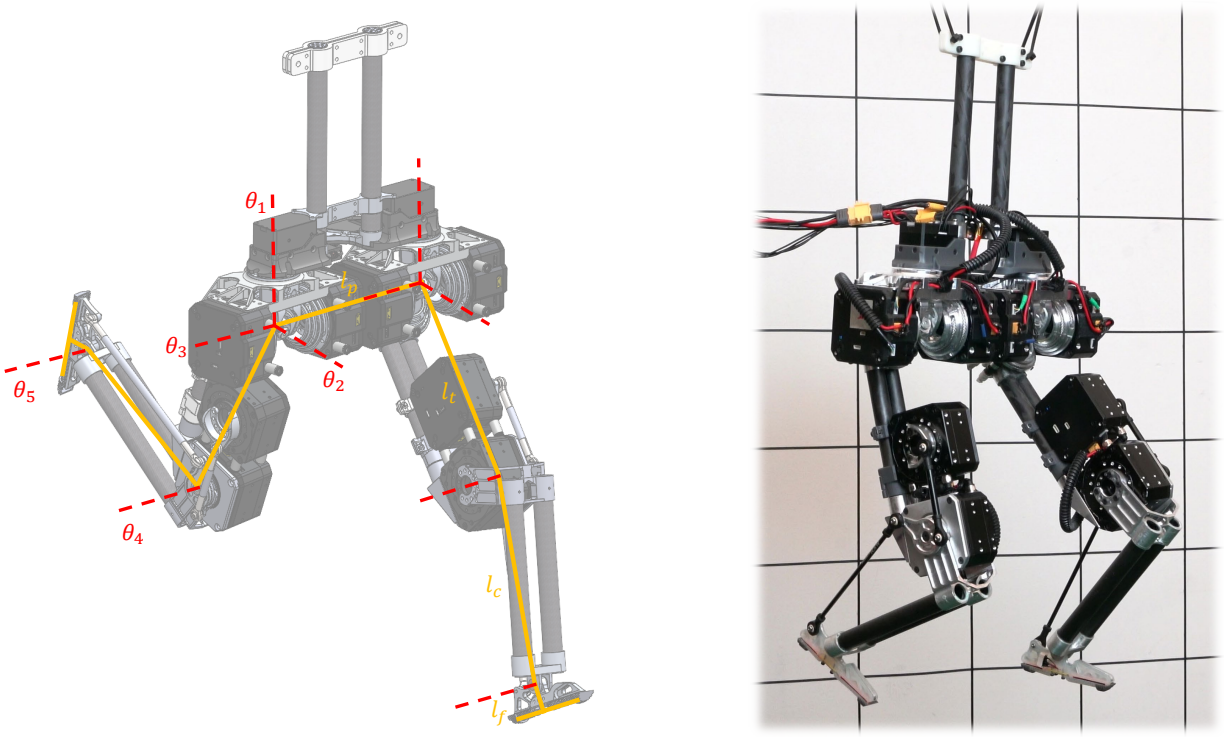


Figure 2.1: Bipedal Robot Unit with Compliance Enhanced (BRUCE). On the left is the CAD model and on the right is its hardware platform.

also brings less backlash than conventional bevel gears. The delicate design also brings about interesting problems in modeling and analysis.

2.2 Design of BRUCE

This section discusses the detailed hardware design of BRUCE. It elaborates on how and why BRUCE is designed in its own way.

2.2.1 Mechanical Configuration

Joint Configuration

To ensure BRUCE has an adequate range of motion while keeping the overall platform simple and lightweight, each leg is composed of a spherical hip joint, a single DoF knee joint, and a single DoF ankle joint, as shown in Figure 2.1. Moreover, each foot has a line contact with

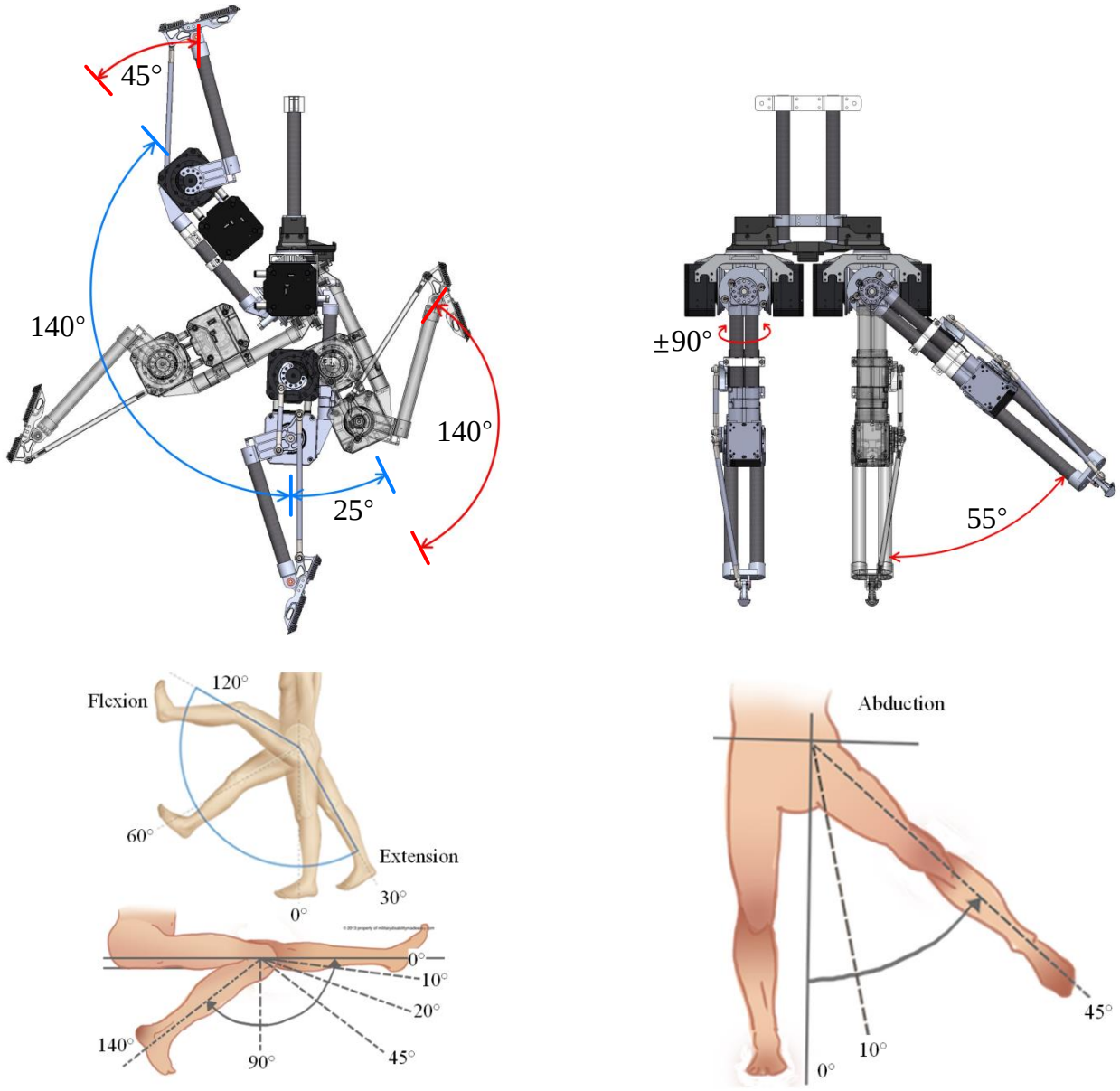


Figure 2.2: Lower body comparison of joint configuration and range of motion between BRUCE and human [158].

the ground so that the actuation in the foot roll direction can be omitted. Unlike regular full-size humanoid robots with fully actuated ankles, the single DoF ankles of BRUCE could lose some functionality on the foot since it only rotates in the pitch direction. Nevertheless, the benefit from the lightweight design outweighs this drawback distinctly when it comes to highly dynamic leg motions.

Table 2.1: BRUCE mechanical parameters.

Parameter	Symbol	Value	Unit
Torso mass	m_b	315	g
Hip mass	m_h	667	g
Thigh mass	m_t	839	g
Calf mass	m_c	96	g
Foot mass	m_f	24	g
Total mass	M	3,567	g
Torso length	l_b	291.5	mm
Pelvis width	l_p	150	mm
Thigh length	l_t	175	mm
Calf length	l_c	169.5	mm
Foot length	l_f	24	mm
Total height	L	660	mm

Link Length

BRUCE is designed to be a miniature bipedal robot with a similar range of motion to that of a human's lower body. Therefore, the size of BRUCE is scaled down from a full-size human. The total height of BRUCE is designed to be 660 mm which is approximately 1/3 of an adult male's height. The resultant link lengths of its torso, thigh, calf, and foot are respectively 291.5 mm, 175 mm, 169.5 mm, and 24 mm, which results in a similar length ratio to that of an average adult [159]. The distance between the two legs is chosen to be 150 mm to prevent possible collision between the hip actuators when they are rotating in the yaw direction. Table 2.1 lists the key mechanical parameters of BRUCE.

Table 2.2: Koala BEAR specifications.

Parameter	Value	Unit
Weight	250	g
Speed Constant	27.3	RPM/V
Torque Constant	0.035	Nm/A
Gear Ratio	9	/
Stall Torque (15 sec)	3.5	Nm
Stall Torque (1.5 sec)	10.5	Nm

2.2.2 Actuation Scheme

In order to have better actuation transparency and compliance to unstructured environments, Koala BEAR proprioceptive actuators [55] from Westwood Robotics [160] are adopted. The module can provide real-time joint states including joint position, velocity, and torque while running an internal control loop at 2 kHz with the embedded microcontroller. The actuator specifications can be found in Table 2.2.

With this proprioceptive actuation scheme, BRUCE's legs are desired to have low inertia in favor of dynamic behaviors. The distribution of the actuators needs to be reconsidered so that they can be kept close to the torso instead of being directly located at each joint like traditional humanoid robots. In the past, researchers have proposed some reliable solutions by using linkage mechanisms [4] and cable-driven systems [161, 162] to reduce the inertia and mass of the limbs. Inspired by previous work, a 2-DoF cable-driven differential pulley system is designed for the hip pitch and roll motions, and two pairs of 4-bar linkage mechanisms are used to actuate the ankle joint.

2.2.3 Hip Design with Cable-Driven System

Instead of connecting two actuators in serial for the pitch and roll directions of the hip joint, a 2-DoF parallel actuation configuration is preferable for BRUCE. First, this design could mount the two actuators on the hip to reduce the mass and inertia of the femur link. Moreover, the available hip pitch torque is doubled as the two actuators are powering the same joint, which could benefit BRUCE during dynamic motions in the sagittal plane, as is usually the case.

Previously, the prototype of BRUCE [158] was using bevel gears to realize the parallel actuation scheme of the hip. However, the leg wobbled easily, and hip joint accuracy was low due to the backlash in gears. To improve the joint accuracy and stability, the compact cable transmission with cable differential [163] is novelly applied on the hip joint to form a cable-driven differential pulley system, which has already been successfully implemented on other robotic joints such as the torso [164] and shoulder [165]. Despite the extra complexity in installation, the cable-driven differential pulley system appears to be a suitable replacement for traditional bevel gears due to its zero-backlash feature. In addition, unlike gears in which grease is necessary for lubrication, no lubrication is needed between the cable and pulley, which could benefit the cleanliness of the hip assembly.

The proposed 2-DoF cable-driven differential pulley system is illustrated in Figure 2.3(a)–(c). To effectively actuate the hip joint in both pitch and roll directions without slip, at least two cables are needed for each pulley attached to the hip actuator, and it results in a total of four cables for the pulley connected to the femur link. As shown in Figure 2.3(b), the blue cable is active when Actuator 2 rotates along positive z_3 direction, while the reverse rotation will make the red cable active. To avoid any broken cables due to the excessive load, the minimum radius R_{min} of wrapping the cable around the pulley needs to be carefully determined, which is related to the maximum torque of the actuator T_{max} and the material

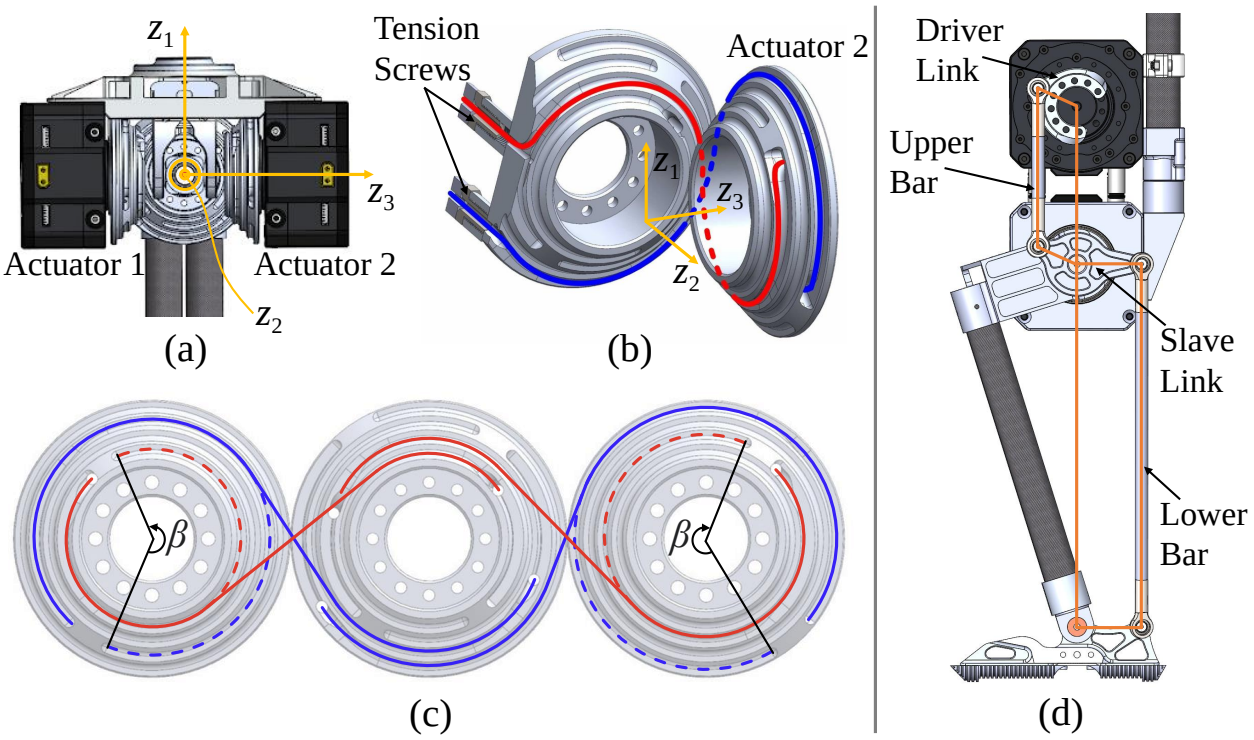


Figure 2.3: BRUCE leg design highlights. (a) Spherical hip joint. (b) Assembly of pulleys and cables, where the cables are pretensioned by screws. (c) Wiring schematic of 2-DoF cable-driven differential pulley system, where β represents the effective rotation region for the pulley and also corresponds to the range of hip roll motion. (d) Linkage mechanism for ankle joint actuation.

property for the cable as follows:

$$R_{\min} = \frac{T}{F_y} \leq \frac{T_{\max}}{F_y} = \frac{T_{\max}}{\sigma_y \cdot \pi r_c^2}, \tag{2.1}$$

where F_y is the axial force at yield, σ_y is the yield stress of the cable material, and r_c is the radius of the cable. In our case, with $T_{\max} = 10.5$ Nm for the Koala BEAR actuator, $\sigma_y = 215$ MPa and $r_c = 2.4$ mm for a 304 stainless steel cable, the maximum of $R_{\min}$ is determined to be 16.2 mm for a safety factor of 1.5. On the real hardware, it is adjusted to 19 mm to properly fit into the assembly.

The parallel configuration of the hip joint in the pitch and roll directions leads to the coupling of the two actuators. As illustrated in Figure 2.3(b) and (c), pure hip pitch motion

will be achieved when the two side pulleys rotate for the same angle in the opposite directions, while rotating in the same direction leads to pure roll motion. Any other combinations of the two pulley rotations will result in both pitch and roll motions simultaneously. The detailed kinematics relationship will be derived in the next section.

Hip Joint Backlash Comparison

To show the cable-driven differential pulley system has far less backlash than the traditional differential bevel gear system, a comparison experiment was conducted to visualize the two backlash conditions. For test setup, two hip assemblies with different designs were mounted to a fixed location respectively, and a background paper was placed behind with reference points on it. For data collection, the thigh link in each assembly was manually aligned to the reference points, and the angle readings from the two hip actuators were recorded for comparison. The result is shown in Figure 2.4 and we can see that the measured angles from the cable-driven hip joint almost perfectly fit the reference while the bevel gear hip module has a poorer performance.

Hip Joint Stiffness Analysis

Furthermore, as the cable-driven transmission is applied to the hip joint, the joint stiffness needs to be analyzed since it might be affected by the cable elongation. The cable axial stiffness k_c can be first determined to be

$$k_c = \frac{AE_c}{L_c} = \frac{\pi r_c^2 \cdot E_c}{2\pi R} = \frac{r_c^2 E_c}{2R}, \tag{2.2}$$

where A, L_c are the cross-sectional area and the length of the cable, and E_c is the Young's Modulus for the cable material. The cable axial stiffness k_c relates the axial force F and the cable elongation δ by Hooke's law:

$$F = k_c \delta. \tag{2.3}$$

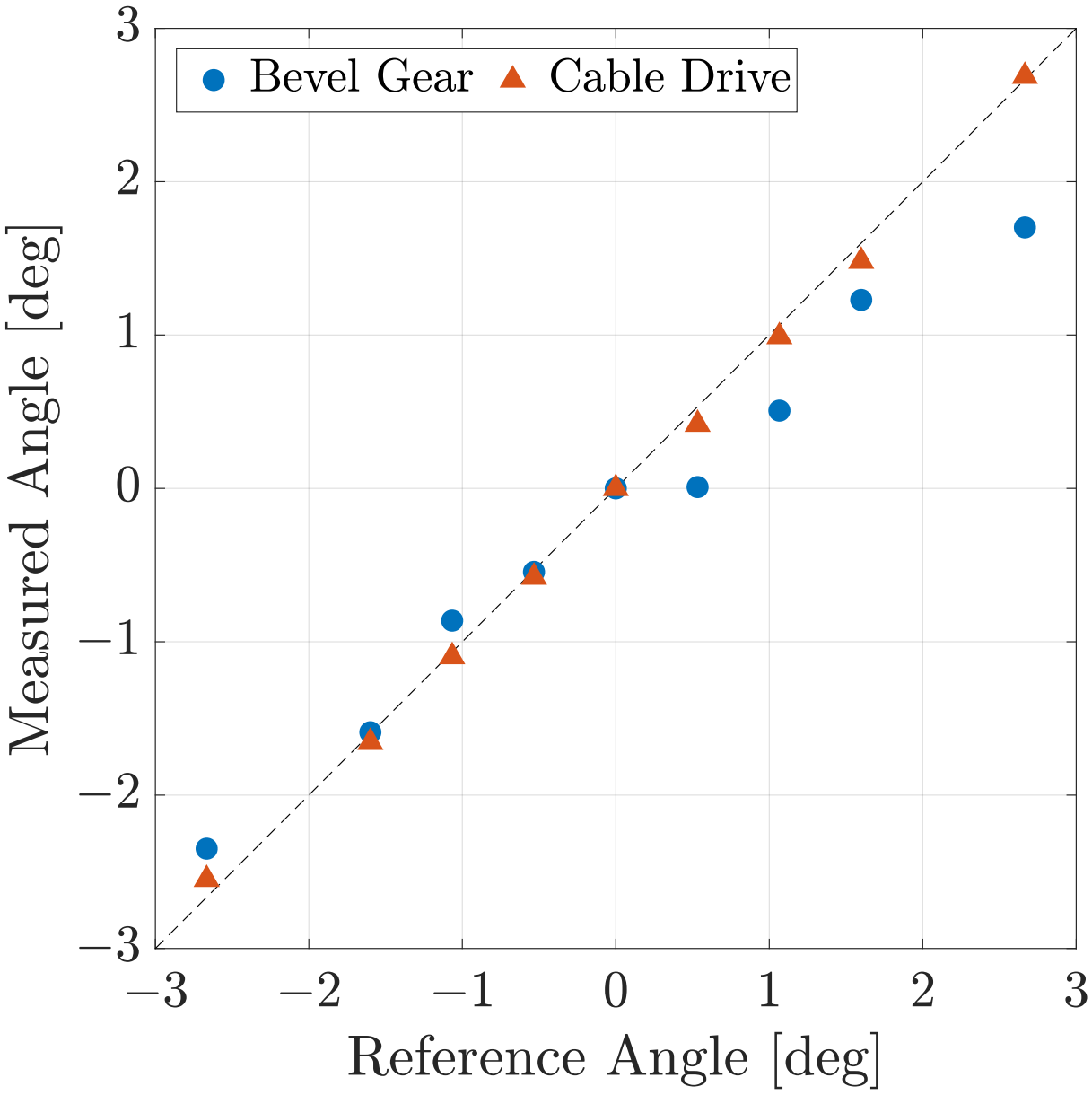


Figure 2.4: BRUCE hip joint backlash comparison between the cable-driven differential pulley system and the differential bevel gear system.

The joint stiffness k_j can then be determined to be

$$k_j = \frac{T}{\Delta\theta} = \frac{FR}{\delta/R} = \underbrace{(F/\delta)}_{:=k_c} R^2 = \frac{1}{2} r_c^2 E_c R, \tag{2.4}$$

where $\Delta\theta$ is the resultant joint rotation angle. With $R = 19$ mm for the pulley, $r_c = 2.4$ mm and $E_c = 1.9 \times 10^{11}$ N/m^2 for the 304 stainless steel cable, the hip joint stiffness is 10,397 Nm/rad, which is sufficiently stiff as the actuator only has a maximum torque of 10.5 Nm. The resultant joint rotation due to cable elongation is 0.058° at worst, which is negligible. In addition, each cable is properly pretensioned by adjusting the screws as shown in Figure 2.3(b) to ensure reliable power transmission.

2.2.4 Leg Design with Linkage Mechanism

The femur and tibia linkages of BRUCE are composed of carbon fiber tubes and topology-optimized aluminium parts. The two actuators for the knee and ankle joints are mounted in the femur link to keep the tibia link as light as possible. Since the ankle actuator is relocated to the femur link, a mechanism to transmit the torque from the actuator to the ankle joint is needed. Generally, timing belt is a good option for power transmission due to its simplicity and the ability to transmit continuous rotational motions. However, due to its low stiffness, there will be unwanted compliance between the belt teeth and pulley. As a result, the rotor-belt resonant frequency will be low, which could limit the torque control bandwidth of the ankle joint [166]. To overcome this problem, a reliable torque transmission with high stiffness is used instead, e.g., linkage mechanism.

As shown in Figure 2.3(d), BRUCE utilizes two pairs of 4-bar linkage mechanisms both of parallelogram configuration to properly transmit the torque from the actuator to the ankle joint with a 1:1 transmission ratio. Since the lower bar is such a thin and long link, the buckling load F_b verification must be done to determine its radius r_l so that it will not buckle under extreme scenarios:

$$F_b = \frac{\pi^2 E_l I}{L_l^2} \geq \frac{T_{\max}}{l} \Rightarrow I \geq \frac{L_l^2 T_{\max}}{\pi^2 E_l l}, \tag{2.5}$$

where E_l, I, L_l, l are the Young's Modulus for the lower bar material, moment of inertia of

Table 2.3: BRUCE ankle joint range of motion.

Knee Angle	**Ankle Angle Range**
$-30°$	$-25° \sim 77°$
$0°$	$-60° \sim 72°$
$30°$	$-58° \sim 50°$
$60°$	$-58° \sim 28°$
$90°$	$-58° \sim -3°$
$120°$	$-58° \sim -32°$

the cross section of the lower bar, the length of the lower bar, and the moment arm of the slave link. To match with the tibia link length, we need to use a 169.5 mm long aluminum rod with $E_l = 6.9 \times 10^{10}$ N/m^2. Substitute $I = \pi r_l^2/4$ into (2.5) and we have

$$r_l \geq \frac{2L_l}{\pi}\sqrt{\frac{T_{\max}}{\pi E_l l}}. \tag{2.6}$$

With $T_{\max} = 10.5$ Nm for the Koala BEAR actuator and $l = 30$ mm, (2.6) will lead to the minimum radius of 3 mm for the lower bar with a safety factor of 1.5. Moreover, due to the complexity of the combination of the two linkage mechanisms, the ankle joint motion depends on the knee configuration. Table 2.3 lists the ankle joint range with varying knee angles. Even though its range of motion is restricted when the knee angle is large, e.g., squatting, it still meets the needs for our applications. Note that we define the positive direction of the knee angle as the knee flexes. The ankle angle is $0°$ when the foot is perpendicular to the tibia and its positive direction is defined as the foot flexes.

Similar to the hip joint, the two actuators for the knee and ankle joints are coupled. The detailed kinematics relationship will be derived in the next section.

Figure 2.5: BRUCE contact sensing foot V1. On the left is the exploded view of the CAD model and on the right is the hardware assembly. For the assembly, the two copper foils are glued to the plastic contact layer, which is then attached to the aluminum base using screws.

2.2.5 Contact Sensing Foot

For BRUCE being able to detect when the contact between the foot and the ground is created or broken for state estimation purpose in an unstructured environment, a contact sensor is needed. A reliable yet lightweight contact sensing foot module is thus proposed for BRUCE, as shown in Figure 2.5. The sensing foot is designed based on the working principle of an electronic switch. When the foot is touching the ground and the contact force is sufficiently large, the plastic contact layer will bend and make the copper foils touch the aluminum base to close the circuit. As for the trigger force, a simple experiment was conducted where the proposed sensing foot was pressed downwards gradually on top of a precise scale by a custom 2-DoF testbed. The critical value of the contact force is recorded as the trigger force when the contact is detected by the sensor. From the experimental results, the trigger forces for the toe and heel both almost remain constant under different contact angles, as 1.18 N for the toe and 2.45 N for the heel on average. The constant trigger force provides BRUCE a reliable contact sensing ability for ground touching status.

Meanwhile, there is a potential false positive contact detection when the leg is swinging in the air with a large acceleration since the plastic contact layer can possibly bend due to inertia force. However, while the contact layer weighs only 1.5 g which is extremely

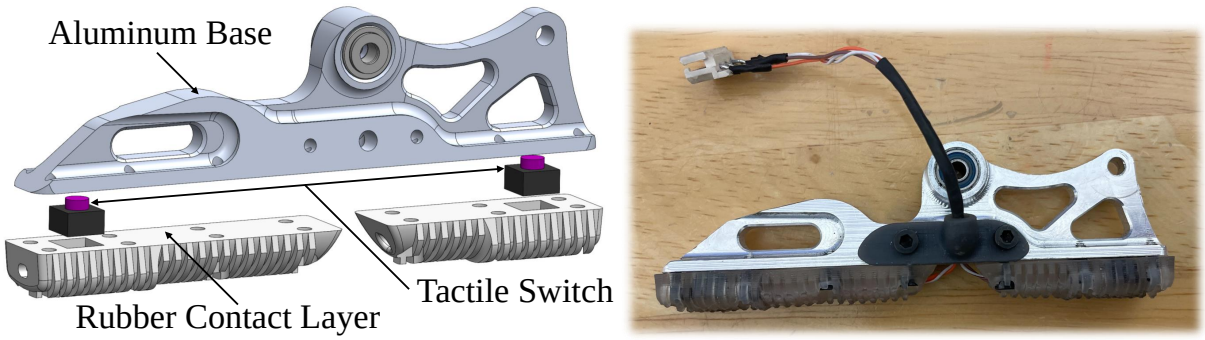


Figure 2.6: BRUCE contact sensing foot V2. On the left is the CAD model and on the right is the hardware assembly.

lightweight, the required acceleration to trigger that false positive detection is computed to be about 80 and 166 times gravitational acceleration for the toe and heel, respectively, which far exceeds the normal operating condition for BRUCE. In addition, the sensing foot is easy for maintenance given its simple structure.

During the hardware experiments, the design of the sensing foot is further modified to ensure more reliable ground contact information for walking purpose. Previously, the plastic contact layer can break from time to time due to the touchdown impact. In addition, dust can easily stick to the copper foils, which can affect their contact quality. The new design of the sensing foot has tactile switches directly inserted into the rubber contact layer, as shown in Figure 2.6. In this way, the foot touchdown is more compliant and the contact detection mechanism is fully isolated from the outside environment, which greatly improves the contact sensing.

2.2.6 Upper Body

The upper body of BRUCE is designed to make BRUCE fully untethered. Meanwhile, the overall system will also benefit from the upper body with the additional DoFs and more lumped inertia at the hip. In the plan, each arm of BRUCE will have three DoFs, which includes a 2-DoF shoulder joint and a 1-DoF elbow joint. Note that the two arms are not

actively used for the locomotion purpose at this point and therefore the entire upper body is considered as one SRB for simplicity from now on.

2.3 Kinematics

Kinematics is the study of the motion of the subject without regard to the forces that cause it. One usually focuses on the position and the orientation, their velocities, and their accelerations while any order derivatives of the variables (not necessarily with respect to time) can be involved as well. Therefore, kinematics can refer to all the space and time properties of the motion. When it comes to the motions of robots, especially robots with articulated limbs, e.g., manipulators and legged robots, kinematic analysis is usually to find out the relationship of the motions between the joint space and operational (or task-oriented) space. The joint space refers to the space of the joint variables which define the spacial relationship (including both position and orientation) between the intermediate frames. The operational space refers to the space of the variables which define the target (or task) frame, e.g., the end effector of a manipulator, the foot and CoM of a legged robot.

While the kinematic analysis can be addressed numerically, it is always better to derive its closed-form solution if possible. First, one can better understand the kinematic relationship with the analytical equations. Second, it can speed up the computation in practice. And third, most importantly, it is just fun to derive them in an elegant manner ☺, right? This section discusses the detailed kinematic analysis of BRUCE. Most of the symbolic calculations are done in *Mathematica* [167]. We will first solve the leg kinematics within the base (or body, or torso) frame and from there the floating base joint will be added to derive the overall whole-body kinematics with respect to the world (or inertia, or global) frame.

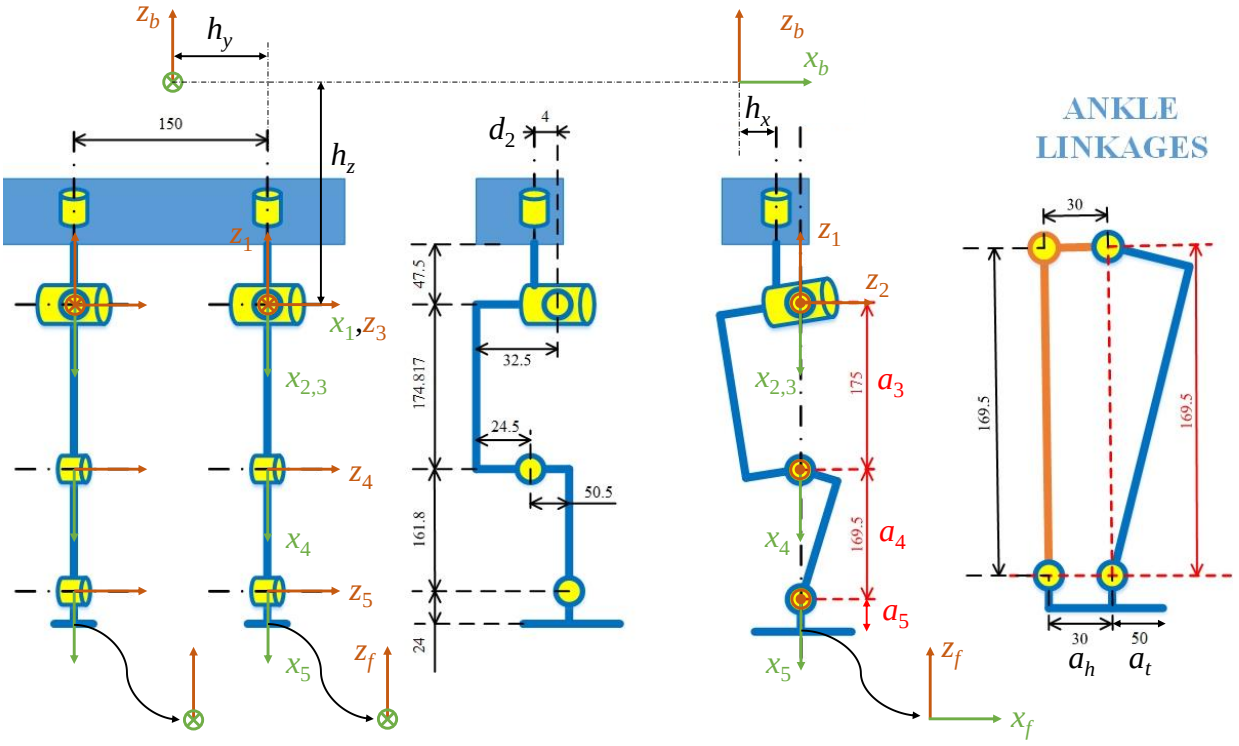


Figure 2.7: BRUCE frame attachment (units in mm).

2.3.1 Leg Kinematics

Consider the leg kinematics of BRUCE as a manipulator problem, it can be solved using the general Denavit-Hartenberg (DH) approach. Figure 2.7 shows the frame attachment and Table 2.4 lists the corresponding modified DH parameters. Note that some of the intermediate frames are intentionally removed for simplicity and as a result, the corresponding spacial relationships cannot be summarized using the DH parameters. In addition, the original version of BRUCE has the link offset d_2 and since it does not add too much computational complexity, we keep it here.

The leg kinematics is almost the same for both legs. There are only two differences. The first one is the sign of h_y (+ for the left leg and − for the right leg). From now on, we will use the postsuperscript *leg* to emphasize the variables are leg dependent and if necessary, we will use the postsuperscript l or r to specifically select the left leg or the right leg respectively. Note that a lot of the variables in this subsection should come with the postsuperscript *leg*

Table 2.4: Modified Denavit-Hartenberg parameters.

i	α_{i-1}	a_{i-1}	d_i	θ_i
2	$-\frac{\pi}{2}$	0	d_2	θ_2
3	$+\frac{\pi}{2}$	0	0	θ_3
4	0	a_3	0	θ_4
5	0	a_4	0	θ_5

to be exact but it is removed most of the time for better readability, e.g., the DH joint positions should always be denoted as

$$\boldsymbol{\theta}^{leg} := \begin{bmatrix} \theta_1^{leg} \\ \theta_2^{leg} \\ \theta_3^{leg} \\ \theta_4^{leg} \\ \theta_5^{leg} \end{bmatrix}. \tag{2.7}$$

The second difference is the conversion between the DH joint space and the actuator space (since the joint directions are the same but the legs are symmetric and the actuators are actually facing to each other), as we will see soon.

Forward Kinematics

Forward Kinematics (FK) is the problem of computing the position and orientation of the target frame given the joint variables. We first derive the individual (homogeneous) transformation matrices which relate the frames attached to neighboring links. We then concatenate these individual transformations to solve for the target frame relative to the base frame.

The individual transformation matrices of BRUCE can be determined to be

$$
{}^{b}\boldsymbol{T}_1 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & h_x \\ \sin\theta_1 & \cos\theta_1 & 0 & h_y^{leg} \\ 0 & 0 & 1 & -h_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{1}\boldsymbol{T}_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ -\sin\theta_2 & -\cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.8}
$$

$$
{}^{2}\boldsymbol{T}_3 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{3}\boldsymbol{T}_4 = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & a_3 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.9}
$$

$$
{}^{4}\boldsymbol{T}_5 = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & a_4 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{5}\boldsymbol{T}_f = \begin{bmatrix} 0 & 0 & -1 & a_5 \\ 1 & 0 & 0 & a_6 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.10}
$$

where $a_6 = a_t > 0$ for the toe, $a_6 = a_m = 0$ for the middle, and $a_6 = -a_h < 0$ for the heel. The overall transformation matrix is thus determined to be

$$
{}^{b}\boldsymbol{T}_f = {}^{b}\boldsymbol{T}_1\,{}^{1}\boldsymbol{T}_2\,{}^{2}\boldsymbol{T}_3\,{}^{3}\boldsymbol{T}_4\,{}^{4}\boldsymbol{T}_5\,{}^{5}\boldsymbol{T}_f = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.11}
$$

where the elements

$$
r_{11} = -c_1 c_2 s_{345} - s_1 c_{345}, \tag{2.12}
$$

$$
r_{21} = -s_1 c_2 s_{345} + c_1 c_{345}, \tag{2.13}
$$

$$
r_{31} = s_2 s_{345}, \tag{2.14}
$$

$$r_{12} = -c_1 s_2, \tag{2.15}$$

$$r_{22} = -s_1 s_2, \tag{2.16}$$

$$r_{32} = -c_2, \tag{2.17}$$

$$r_{13} = -c_1 c_2 c_{345} + s_1 s_{345}, \tag{2.18}$$

$$r_{23} = -s_1 c_2 c_{345} - c_1 s_{345}, \tag{2.19}$$

$$r_{33} = s_2 c_{345}, \tag{2.20}$$

$$p_x = c_1 c_2 a^c_{3456} - s_1 \left(d_2 + a^s_{3456}\right) + h_x, \tag{2.21}$$

$$p_y = s_1 c_2 a^c_{3456} + c_1 \left(d_2 + a^s_{3456}\right) + h^{leg}_y, \tag{2.22}$$

$$p_z = -s_2 a^c_{3456} - h_z. \tag{2.23}$$

$$a^c_{3456} = a_3 c_3 + a_4 c_{34} + a_5 c_{345} - a_6 s_{345}, \tag{2.24}$$

$$a^s_{3456} = a_3 s_3 + a_4 s_{34} + a_5 s_{345} + a_6 c_{345}. \tag{2.25}$$

Note that we denote $c_{ijk} = \cos\left(\theta_i + \theta_j + \theta_k\right)$ and $s_{ijk} = \sin\left(\theta_i + \theta_j + \theta_k\right)$ for simplicity.

With ${}^b\boldsymbol{T}_f$, we can know both the position ${}^b\boldsymbol{p}_f\left(\boldsymbol{\theta}\right)$ and the orientation ${}^b\boldsymbol{R}_f\left(\boldsymbol{\theta}\right)$ of the foot frame $\{f\}$ relative to the base frame $\{b\}$ (both as functions of the joint configuration $\boldsymbol{\theta}$):

$${}^b\boldsymbol{p}_f = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}, \quad {}^b\boldsymbol{R}_f = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{2.26}$$

Now we just need to take the time derivatives to get their velocities and accelerations:

$${}^b\dot{\boldsymbol{p}}_f = \frac{\mathrm{d}\,{}^b\boldsymbol{p}_f}{\mathrm{d}t} = \frac{\partial\,{}^b\boldsymbol{p}_f}{\partial\boldsymbol{\theta}}\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}t} = {}^b_v\boldsymbol{J}_f\dot{\boldsymbol{\theta}}, \tag{2.27}$$

$${}^b\ddot{\boldsymbol{p}}_f = \frac{\mathrm{d}^2\,{}^b\boldsymbol{p}_f}{\mathrm{d}t^2} = \frac{\mathrm{d}\,{}^b\dot{\boldsymbol{p}}_f}{\mathrm{d}t} = {}^b_v\boldsymbol{J}_f\ddot{\boldsymbol{\theta}} + {}^b_v\dot{\boldsymbol{J}}_f\dot{\boldsymbol{\theta}}, \tag{2.28}$$

$$ {}^{b}\dot{\boldsymbol{R}}_f = \frac{\mathrm{d}^{b}\boldsymbol{R}_f}{\mathrm{d}t} = \widehat{{}^{b}\boldsymbol{\omega}_f}\,{}^{b}\boldsymbol{R}_f \Rightarrow {}^{b}\boldsymbol{\omega}_f = \left({}^{b}\dot{\boldsymbol{R}}_f\,{}^{b}\boldsymbol{R}_f^{\top}\right)^{\vee} = {}^{b}_{\omega}\boldsymbol{J}_f\dot{\boldsymbol{\theta}}, \tag{2.29} $$

$$ {}^{b}\dot{\boldsymbol{\omega}}_f = \frac{\mathrm{d}^{b}\boldsymbol{\omega}_f}{\mathrm{d}t} = {}^{b}_{\omega}\boldsymbol{J}_f\ddot{\boldsymbol{q}} + {}^{b}_{\omega}\dot{\boldsymbol{J}}_f\dot{\boldsymbol{\theta}}. \tag{2.30} $$

Based on that, the linear velocity Jacobian ${}^{b}_{v}\boldsymbol{J}_f(\boldsymbol{\theta})$ can be determined to be

$$ {}^{b}_{v}\boldsymbol{J}_f = \frac{\partial {}^{b}\boldsymbol{p}_f}{\partial \boldsymbol{\theta}} = \begin{bmatrix} j_{11} & j_{12} & j_{13} & j_{14} & j_{15} \\ j_{21} & j_{22} & j_{23} & j_{24} & j_{25} \\ j_{31} & j_{32} & j_{33} & j_{34} & j_{35} \end{bmatrix}, \tag{2.31} $$

where the elements

$$ j_{11} = \frac{\mathrm{d}p_x}{\mathrm{d}\theta_1} = -s_1c_2a^c_{3456} - c_1\left(d_2 + a^s_{3456}\right), \tag{2.32} $$

$$ j_{21} = \frac{\mathrm{d}p_y}{\mathrm{d}\theta_1} = +c_1c_2a^c_{3456} - s_1\left(d_2 + a^s_{3456}\right), \tag{2.33} $$

$$ j_{31} = \frac{\mathrm{d}p_z}{\mathrm{d}\theta_1} = 0, \tag{2.34} $$

$$ j_{12} = \frac{\mathrm{d}p_x}{\mathrm{d}\theta_2} = -c_1s_2a^c_{3456}, \tag{2.35} $$

$$ j_{22} = \frac{\mathrm{d}p_y}{\mathrm{d}\theta_2} = -s_1s_2a^c_{3456}, \tag{2.36} $$

$$ j_{32} = \frac{\mathrm{d}p_z}{\mathrm{d}\theta_2} = -c_2a^c_{3456}, \tag{2.37} $$

$$ j_{13} = \frac{\mathrm{d}p_x}{\mathrm{d}\theta_3} = -s_1a^c_{3456} - c_1c_2a^s_{3456}, \tag{2.38} $$

$$ j_{23} = \frac{\mathrm{d}p_y}{\mathrm{d}\theta_3} = +c_1a^c_{3456} - s_1c_2a^s_{3456}, \tag{2.39} $$

$$ j_{33} = \frac{\mathrm{d}p_z}{\mathrm{d}\theta_3} = +s_2a^s_{3456}, \tag{2.40} $$

$$ j_{14} = \frac{\mathrm{d}p_x}{\mathrm{d}\theta_4} = -s_1a^c_{456} - c_1c_2a^s_{456}, \tag{2.41} $$

$$j_{24} = \frac{\mathrm{d}p_y}{\mathrm{d}\theta_4} = +c_1 a^c_{456} - s_1 c_2 a^s_{456}, \tag{2.42}$$

$$j_{34} = \frac{\mathrm{d}p_z}{\mathrm{d}\theta_4} = +s_2 a^s_{456}, \tag{2.43}$$

$$j_{15} = \frac{\mathrm{d}p_x}{\mathrm{d}\theta_5} = -s_1 a^c_{56} - c_1 c_2 a^s_{56}, \tag{2.44}$$

$$j_{25} = \frac{\mathrm{d}p_y}{\mathrm{d}\theta_5} = +c_1 a^c_{56} - s_1 c_2 a^s_{56}, \tag{2.45}$$

$$j_{35} = \frac{\mathrm{d}p_z}{\mathrm{d}\theta_5} = +s_2 a^s_{56}, \tag{2.46}$$

$$a^c_{456} = a_4 c_{34} + a_5 c_{345} - a_6 s_{345}, \tag{2.47}$$

$$a^s_{456} = a_4 s_{34} + a_5 s_{345} + a_6 c_{345}, \tag{2.48}$$

$$a^c_{56} = a_5 c_{345} - a_6 s_{345}, \tag{2.49}$$

$$a^s_{56} = a_5 s_{345} + a_6 c_{345}. \tag{2.50}$$

Its time derivative is then determined to be

$$\overset{b}{_v}\dot{\boldsymbol{J}}_f = \frac{\mathrm{d}\,\overset{b}{_v}\boldsymbol{J}_f}{\mathrm{d}t} = \begin{bmatrix} \dot{j}_{11} & \dot{j}_{12} & \dot{j}_{13} & \dot{j}_{14} & \dot{j}_{15} \\ \dot{j}_{21} & \dot{j}_{22} & \dot{j}_{23} & \dot{j}_{24} & \dot{j}_{25} \\ \dot{j}_{31} & \dot{j}_{32} & \dot{j}_{33} & \dot{j}_{34} & \dot{j}_{35} \end{bmatrix}, \tag{2.51}$$

where the elements

$$\dot{j}_{11} = -j_{21}\dot{\theta}_1 - j_{22}\dot{\theta}_2 - j_{23}\dot{\theta}_3 - j_{24}\dot{\theta}_4 - j_{25}\dot{\theta}_5, \tag{2.52}$$

$$\dot{j}_{21} = +j_{11}\dot{\theta}_1 + j_{12}\dot{\theta}_2 + j_{13}\dot{\theta}_3 + j_{14}\dot{\theta}_4 + j_{15}\dot{\theta}_5, \tag{2.53}$$

$$\dot{j}_{31} = 0, \tag{2.54}$$

$$\dot{j}_{12} = -j_{22}\dot{\theta}_1 - c_1 c_2 a^c_{3456}\dot{\theta}_2 + c_1 s_2 \left(a^s_{3456}\dot{\theta}_3 + a^s_{456}\dot{\theta}_4 + a^s_{56}\dot{\theta}_5 \right), \tag{2.55}$$

$$\dot{j}_{22} = +j_{12}\dot{\theta}_1 - s_1 c_2 a^c_{3456}\dot{\theta}_2 + s_1 s_2 \left(a^s_{3456}\dot{\theta}_3 + a^s_{456}\dot{\theta}_4 + a^s_{56}\dot{\theta}_5 \right), \tag{2.56}$$

$$\dot{j}_{32} = s_2 a^c_{3456}\dot{\theta}_2 + c_2 \left(a^s_{3456}\dot{\theta}_3 + a^s_{456}\dot{\theta}_4 + a^s_{56}\dot{\theta}_5 \right), \tag{2.57}$$

$$\begin{aligned}\dot{j}_{13} = &-j_{23}\dot{\theta}_1 + c_1 s_2 a^s_{3456}\dot{\theta}_2 - (j_{21} + s_1 d_2)\,\dot{\theta}_3 + (s_1 a^s_{456} - c_1 c_2 a^c_{456})\,\dot{\theta}_4 + \\ &(s_1 a^s_{56} - c_1 c_2 a^c_{56})\,\dot{\theta}_5,\end{aligned} \tag{2.58}$$

$$\begin{aligned}\dot{j}_{23} = &+j_{13}\dot{\theta}_1 + s_1 s_2 a^s_{3456}\dot{\theta}_2 + (j_{11} + c_1 d_2)\,\dot{\theta}_3 - (c_1 a^s_{456} + s_1 c_2 a^c_{456})\,\dot{\theta}_4 - \\ &(c_1 a^s_{56} + s_1 c_2 a^c_{56})\,\dot{\theta}_5,\end{aligned} \tag{2.59}$$

$$\dot{j}_{33} = c_2 a^s_{3456}\dot{\theta}_2 + s_2 \left(a^c_{3456}\dot{\theta}_3 + a^c_{456}\dot{\theta}_4 + a^c_{56}\dot{\theta}_5 \right), \tag{2.60}$$

$$\dot{j}_{14} = -j_{24}\dot{\theta}_1 + c_1 s_2 a^s_{456}\dot{\theta}_2 + (s_1 a^s_{456} - c_1 c_2 a^c_{456}) \left(\dot{\theta}_3 + \dot{\theta}_4 \right) + (s_1 a^s_{56} - c_1 c_2 a^c_{56})\,\dot{\theta}_5, \tag{2.61}$$

$$\dot{j}_{24} = +j_{14}\dot{\theta}_1 + s_1 s_2 a^s_{456}\dot{\theta}_2 - (c_1 a^s_{456} + s_1 c_2 a^c_{456}) \left(\dot{\theta}_3 + \dot{\theta}_4 \right) - (c_1 a^s_{56} + s_1 c_2 a^c_{56})\,\dot{\theta}_5, \tag{2.62}$$

$$\dot{j}_{34} = c_2 a^s_{456}\dot{\theta}_2 + s_2 a^c_{456} \left(\dot{\theta}_3 + \dot{\theta}_4 \right) + s_2 a^c_{56}\dot{\theta}_5, \tag{2.63}$$

$$\dot{j}_{15} = -j_{25}\dot{\theta}_1 + c_1 s_2 a^s_{56}\dot{\theta}_2 + (s_1 a^s_{56} - c_1 c_2 a^c_{56}) \left(\dot{\theta}_3 + \dot{\theta}_4 + \dot{\theta}_5 \right), \tag{2.64}$$

$$\dot{j}_{25} = +j_{15}\dot{\theta}_1 + s_1 s_2 a^s_{56}\dot{\theta}_2 - (c_1 a^s_{56} + s_1 c_2 a^c_{56}) \left(\dot{\theta}_3 + \dot{\theta}_4 + \dot{\theta}_5 \right), \tag{2.65}$$

$$\dot{j}_{35} = c_2 a^s_{56}\dot{\theta}_2 + s_2 a^c_{56} \left(\dot{\theta}_3 + \dot{\theta}_4 + \dot{\theta}_5 \right). \tag{2.66}$$

The angular velocity Jacobian ${}^b_{\omega}\boldsymbol{J}_f(\boldsymbol{\theta})$ can be alternatively determined by the following simple construction:

$${}^b_{\omega}\boldsymbol{J}_f = \begin{bmatrix} {}^b\boldsymbol{z}_1 & {}^b\boldsymbol{z}_2 & {}^b\boldsymbol{z}_3 & {}^b\boldsymbol{z}_4 & {}^b\boldsymbol{z}_5 \end{bmatrix}, \tag{2.67}$$

where the rotation (or $\boldsymbol{z}$) axes

$$ {}^b\boldsymbol{z}_1 = {}^b\boldsymbol{R}_1{}^1\boldsymbol{z}_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad {}^b\boldsymbol{z}_2 = {}^b\boldsymbol{R}_2{}^2\boldsymbol{z}_2 = \begin{bmatrix} -s_1 \\ c_1 \\ 0 \end{bmatrix}, \tag{2.68} $$

$$ {}^b\boldsymbol{z}_3 = {}^b\boldsymbol{R}_3{}^3\boldsymbol{z}_3 = \begin{bmatrix} c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix}, \qquad {}^b\boldsymbol{z}_4 = {}^b\boldsymbol{R}_4{}^4\boldsymbol{z}_4 = \begin{bmatrix} c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix}, \tag{2.69} $$

$$ {}^b\boldsymbol{z}_5 = {}^b\boldsymbol{R}_5{}^5\boldsymbol{z}_5 = \begin{bmatrix} c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix}. \tag{2.70} $$

We can see that the last three columns are the same since their rotation axes are parallel to each other. Note that ${}^b_\omega\boldsymbol{J}_f\dot{\boldsymbol{\theta}}$ will actually give us the angular velocity of frame $\{5\}$ described in the base frame $\{b\}$ and frame $\{f\}$ shares the same angular velocity since they are relatively stationary. Its time derivative is then determined to be

$$ {}^b_\omega\dot{\boldsymbol{J}}_f = \frac{\mathrm{d}\,{}^b_\omega\boldsymbol{J}_f}{\mathrm{d}t} = \begin{bmatrix} g_{11} & g_{12} & g_{13} & g_{14} & g_{15} \\ g_{21} & g_{22} & g_{23} & g_{24} & g_{25} \\ g_{31} & g_{32} & g_{33} & g_{34} & g_{35} \end{bmatrix}, \tag{2.71} $$

where the elements

$$ g_{11} = 0, \tag{2.72} $$

$$ g_{21} = 0, \tag{2.73} $$

$$ g_{31} = 0, \tag{2.74} $$

$$ g_{12} = -c_1\dot{\theta}_1, \tag{2.75} $$

$$g_{22} = -s_1\dot{\theta}_1, \tag{2.76}$$

$$g_{32} = 0, \tag{2.77}$$

$$g_{13} = -s_1 s_2\dot{\theta}_1 + c_1 c_2\dot{\theta}_2, \tag{2.78}$$

$$g_{23} = +c_1 s_2\dot{\theta}_1 + s_1 c_2\dot{\theta}_2, \tag{2.79}$$

$$g_{33} = -s_2\dot{\theta}_2, \tag{2.80}$$

$$g_{14} = -s_1 s_2\dot{\theta}_1 + c_1 c_2\dot{\theta}_2, \tag{2.81}$$

$$g_{24} = +c_1 s_2\dot{\theta}_1 + s_1 c_2\dot{\theta}_2, \tag{2.82}$$

$$g_{34} = -s_2\dot{\theta}_2, \tag{2.83}$$

$$g_{15} = -s_1 s_2\dot{\theta}_1 + c_1 c_2\dot{\theta}_2, \tag{2.84}$$

$$g_{25} = +c_1 s_2\dot{\theta}_1 + s_1 c_2\dot{\theta}_2, \tag{2.85}$$

$$g_{35} = -s_2\dot{\theta}_2. \tag{2.86}$$

Inverse Kinematics

Inverse Kinematics (IK) is the converse of FK – given the desired position and orientation of the target frame relative to the base frame, how do we compute the set of joint variables which will achieve it? It usually refers to the configuration part only since the inversion of the velocity and acceleration can be simply done by the "inverse" of the Jacobians.

Before we even start, note that each leg of BRUCE is deficient since it only has five DoFs, which is less than six. As a result, we cannot achieve any arbitrary special configurations of both position and orientation. For our application, we are most interested in the position configuration with three DoFs as well as the direction of the x_f–axis (foot pointing direction) with two DoFs. Therefore, the IK problem is that given (2.12) (2.13) (2.14) (2.21) (2.22)

(2.23), solve θ_1, θ_2, θ_3, θ_4, θ_5. By OBSERVATION ☺, we can obtain

$$\left(\left(p_x - h_x\right) c_1 + \left(p_y - h_y^{leg}\right) s_1\right) s_2 + \left(p_z + h_z\right) c_2 = 0, \tag{2.87}$$

$$\left(r_{11} c_1 + r_{21} s_1\right) s_2 + r_{31} c_2 = 0. \tag{2.88}$$

Further eliminating θ_2 yields

$$\underbrace{\left(\left(p_x - h_x\right) r_{31} - \left(p_z + h_z\right) r_{11}\right)}_{k_1} c_1 + \underbrace{\left(\left(p_y - h_y^{leg}\right) r_{31} - \left(p_z + h_z\right) r_{21}\right)}_{k_2} s_1 = 0 \tag{2.89}$$

$$\Rightarrow \theta_1 = \operatorname{atan2}\left(-k_1, k_2\right). \tag{2.90}$$

Note that θ_1 is nominally around $-90°$ and thus $s_1 < 0$ for our case, i.e., the other solution is ignored. To be exact, we also need to discuss the situations when k_1 and k_2 are both zero, e.g., $p_x = h_x$, $p_y = h_y^{leg}$, $p_z = -h_z$ or $p_x = h_x$, $p_y = h_y^{leg}$, $r_{11} = r_{21} = 0$, which lead to infinite numbers of solutions of θ_1. However, they will hardly happen during nominal operation. With θ_1 solved, θ_2 can be determined to be

$$\theta_2 = \operatorname{atan2}\left(-p_z - h_z, \left(p_x - h_x\right) c_1 + \left(p_y - h_y^{leg}\right) s_1\right). \tag{2.91}$$

Note that θ_2 is nominally around $90°$ and thus $s_2 > 0$ for our case, i.e., the other solution is ignored as well. Substituting θ_1 and θ_2 into (2.12), (2.13), and (2.14) we can get

$$c_{345} = r_{21} c_1 - r_{11} s_1, \tag{2.92}$$

$$s_{345} = \begin{cases} \dfrac{r_{31}}{s_2} & \text{if } s_2 \neq 0, \\ -\dfrac{r_{11} c_1 + r_{21} s_1}{c_2} & \text{if } s_2 = 0. \end{cases} \tag{2.93}$$

Substituting c_{345} and s_{345} into (2.21), (2.22), and (2.23) we can obtain

$$k_3 = a_3c_3 + a_4c_{34} = -a_5c_{345} + a_6s_{345} + \begin{cases} \dfrac{p_z + h_z}{-s_2} & \text{if } s_2 \neq 0, \\ \dfrac{(p_x - h_x)\, c_1 + \left(p_y - h_y^{leg}\right) s_1}{c_2} & \text{if } s_2 = 0, \end{cases} \tag{2.94}$$

$$k_4 = a_3s_3 + a_4s_{34} = \left(p_y - h_y^{leg}\right) c_1 - (p_x - h_x)\, s_1 - a_5s_{345} - a_6c_{345} - d_2. \tag{2.95}$$

Sum of squares of them above yields

$$k_3^2 + k_4^2 = a_3^2 + a_4^2 + 2a_3a_4c_4 \Rightarrow c_4 = \frac{k_3^2 + k_4^2 - a_3^2 - a_4^2}{2a_3a_4} \tag{2.96}$$

$$\Rightarrow \theta_4 = \operatorname{atan2}\left(-\sqrt{1 - c_4^2}, c_4\right). \tag{2.97}$$

Note that θ_4 is nominally nonpositive and thus $s_4 \leq 0$ for our case, i.e., the other solution is ignored. Substituting θ_4 into (2.94) and (2.95) we can obtain

$$\begin{cases} (a_3 + a_4c_4)\, c_3 - a_4s_4s_3 = k_3 \\ a_4s_4c_3 + (a_3 + a_4c_4)\, s_3 = k_4 \end{cases} \tag{2.98}$$

$$\Rightarrow \theta_3 = \operatorname{atan2}\left((a_3 + a_4c_4)\, k_4 - a_4s_4k_3, (a_3 + a_4c_4)\, k_3 + a_4s_4k_4\right), \tag{2.99}$$

since the determinant $(a_3 + a_4c_4)^2 + (a_4s_4)^2 = k_3^2 + k_4^2 > 0$. Note that θ_3 will have infinitely many solutions if $k_3^2 + k_4^2 = 0 \Rightarrow a_3 = a_4, c_4 = -1$, which will never happen ☺. Finally, substituting θ_3 and θ_4 into (2.92) and (2.93) and we can solve

$$\theta_5 = \operatorname{atan2}\left(s_{345}, c_{345}\right) - \theta_3 - \theta_4. \tag{2.100}$$

To sum up, the IK solution of the legs is given by

$$\theta_1 = \operatorname{atan2}\left(-k_1, k_2\right), \tag{2.101}$$

$$\theta_2 = \text{atan2}\left(-p_z - h_z, (p_x - h_x)\, c_1 + \left(p_y - h_y^{leg}\right) s_1\right), \tag{2.102}$$

$$\theta_4 = \text{atan2}\left(-\sqrt{1 - c_4^2}, c_4\right), \tag{2.103}$$

$$\theta_3 = \text{atan2}\left(\left(a_3 + a_4 c_4\right) k_4 - a_4 s_4 k_3, \left(a_3 + a_4 c_4\right) k_3 + a_4 s_4 k_4\right), \tag{2.104}$$

$$\theta_5 = \text{atan2}\left(s_{345}, c_{345}\right) - \theta_3 - \theta_4, \tag{2.105}$$

where the variables

$$k_1 = \left(p_x - h_x\right) r_{31} - \left(p_z + h_z\right) r_{11}, \tag{2.106}$$

$$k_2 = \left(p_y - h_y^{leg}\right) r_{31} - \left(p_z + h_z\right) r_{21}, \tag{2.107}$$

$$c_{345} = r_{21} c_1 - r_{11} s_1, \tag{2.108}$$

$$s_{345} = \begin{cases} \dfrac{r_{31}}{s_2} & \text{if } s_2 \neq 0, \\ -\dfrac{r_{11} c_1 + r_{21} s_1}{c_2} & \text{if } s_2 = 0, \end{cases} \tag{2.109}$$

$$k_3 = -a_5 c_{345} + a_6 s_{345} + \begin{cases} \dfrac{p_z + h_z}{-s_2} & \text{if } s_2 \neq 0, \\ \dfrac{\left(p_x - h_x\right) c_1 + \left(p_y - h_y^{leg}\right) s_1}{c_2} & \text{if } s_2 = 0, \end{cases} \tag{2.110}$$

$$k_4 = \left(p_y - h_y^{leg}\right) c_1 - \left(p_x - h_x\right) s_1 - a_5 s_{345} - a_6 c_{345} - d_2, \tag{2.111}$$

$$c_4 = \frac{k_3^2 + k_4^2 - a_3^2 - a_4^2}{2 a_3 a_4}. \tag{2.112}$$

Note that $\theta_1 \sim -\pi/2$, $\theta_2 \sim \pi/2$, $\theta_3 \sim 0$, $\theta_4 \leq 0$, and $\theta_5 \sim 0$. With BRUCE leg IK derived in closed form, we can easily achieve any desired leg configuration by simply specifying the foot position ${}^b\boldsymbol{p}_f^{leg}$ and direction ${}^b\boldsymbol{x}_f^{leg}$, denoted as

$$\boldsymbol{\theta}^{leg} = \mathbf{IK}\left({}^b\boldsymbol{p}_f^{leg}, {}^b\boldsymbol{x}_f^{leg}\right) \quad \text{or} \quad \boldsymbol{\theta}^{leg} = \mathbf{IK}\left({}^b\boldsymbol{T}_f^{leg}\right) \tag{2.113}$$

for convenience although the other two axes are not used at all.

Conversion Between DH Joint Space and Actuator Space

To get the present DH joint information, e.g., present position, velocity, and torque (or force), as well as to realize the joint command, e.g., goal position, velocity, and torque, we need to know the conversion between the DH joint space and the actuator space. The conversion is not trivial since the actuators are coupled, e.g., for the right leg, we have

$$\begin{cases} \theta_1^r = -\varphi_1^r \\ \theta_2^r = -\frac{1}{2}\left(\varphi_2^r + \varphi_3^r\right) + \frac{\pi}{2} \\ \theta_3^r = +\frac{1}{2}\left(\varphi_2^r - \varphi_3^r\right) \\ \theta_4^r = -\varphi_4^r \\ \theta_5^r = +\varphi_4^r + \varphi_5^r \end{cases} \Rightarrow \begin{cases} \varphi_1^r = -\theta_1^r \\ \varphi_2^r = -\theta_2^r + \theta_3^r + \frac{\pi}{2} \\ \varphi_3^r = -\theta_2^r - \theta_3^r + \frac{\pi}{2} \\ \varphi_4^r = -\theta_4^r \\ \varphi_5^r = +\theta_4^r + \theta_5^r \end{cases}, \tag{2.114}$$

where φ_i^{leg} denotes the rotor rotational position of the ith Koala BEAR actuator. Similarly, for the left leg, we have

$$\begin{cases} \theta_1^l = -\varphi_1^l \\ \theta_2^l = -\frac{1}{2}\left(\varphi_2^l + \varphi_3^l\right) + \frac{\pi}{2} \\ \theta_3^l = +\frac{1}{2}\left(\varphi_2^l - \varphi_3^l\right) \\ \theta_4^l = +\varphi_4^l \\ \theta_5^l = -\varphi_4^l - \varphi_5^l \end{cases} \Rightarrow \begin{cases} \varphi_1^l = -\theta_1^l \\ \varphi_2^l = -\theta_2^l + \theta_3^l + \frac{\pi}{2} \\ \varphi_3^l = -\theta_2^l - \theta_3^l + \frac{\pi}{2} \\ \varphi_4^l = +\theta_4^l \\ \varphi_5^l = -\theta_4^l - \theta_5^l \end{cases}. \tag{2.115}$$

We can see that the hip configurations for both legs are the same while the knee and ankle configurations are not. This is because the corresponding actuators are facing to each other, which reverses the actuator rotor rotation direction. Note that the rotor zero position should also coincide with the nominal stance configuration in Figure 2.7, where the frames are all

aligned. Now taking the time derivatives of them gives the velocity mapping:

$$\begin{bmatrix} \dot{\theta}_1^r \\ \dot{\theta}_2^r \\ \dot{\theta}_3^r \\ \dot{\theta}_4^r \\ \dot{\theta}_5^r \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -0.5 & -0.5 & 0 & 0 \\ 0 & +0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & +1 & +1 \end{bmatrix} \begin{bmatrix} \dot{\varphi}_1^r \\ \dot{\varphi}_2^r \\ \dot{\varphi}_3^r \\ \dot{\varphi}_4^r \\ \dot{\varphi}_5^r \end{bmatrix}, \tag{2.116}$$

$$\begin{bmatrix} \dot{\theta}_1^l \\ \dot{\theta}_2^l \\ \dot{\theta}_3^l \\ \dot{\theta}_4^l \\ \dot{\theta}_5^l \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -0.5 & -0.5 & 0 & 0 \\ 0 & +0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} \dot{\varphi}_1^l \\ \dot{\varphi}_2^l \\ \dot{\varphi}_3^l \\ \dot{\varphi}_4^l \\ \dot{\varphi}_5^l \end{bmatrix}. \tag{2.117}$$

We can easily verify that these two Jacobians are invertible. Finally, the transpose of these two Jacobians yields the static torque mapping:

$$\begin{bmatrix} \kappa_1^r \\ \kappa_2^r \\ \kappa_3^r \\ \kappa_4^r \\ \kappa_5^r \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -0.5 & +0.5 & 0 & 0 \\ 0 & -0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0 & -1 & +1 \\ 0 & 0 & 0 & 0 & +1 \end{bmatrix} \begin{bmatrix} \tau_1^r \\ \tau_2^r \\ \tau_3^r \\ \tau_4^r \\ \tau_5^r \end{bmatrix}, \tag{2.118}$$

$$\begin{bmatrix} \kappa_1^l \\ \kappa_2^l \\ \kappa_3^l \\ \kappa_4^l \\ \kappa_5^l \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -0.5 & +0.5 & 0 & 0 \\ 0 & -0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0 & +1 & -1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \tau_1^l \\ \tau_2^l \\ \tau_3^l \\ \tau_4^l \\ \tau_5^l \end{bmatrix}, \tag{2.119}$$

where κ_i^{leg} denotes the actuator output torque and τ_i^{leg} denotes the joint torque. It is tricky to fully understand the static torque relationship and one easy way is to compare BRUCE's parallel (or coupled) leg configuration with its series counterpart. For example, for the serial leg with the same configuration (and the same frame attachment), the net torque acting on the left foot link would be τ_5^l. However, for the parallel leg of BRUCE, the net torque acting on the left foot link is $-\kappa_5^l$. We need to set them equal in order to have the same effect and thus we have $\kappa_5^l = -\tau_5^l$. Similarly, for the serial leg, the net torque acting on the left shank link would be $\tau_4^l - \tau_5^l$ and for the BRUCE leg, it is just κ_4^l. We need to set them equal as well for the same effect. Things get complicated for the thigh link. For the serial leg, the net torque would be $\tau_3^l - \tau_4^l$ in the pitch direction and τ_2^l in the roll direction. For the BRUCE leg, it is $-\kappa_4^l + \kappa_5^l + \kappa_2^l - \kappa_3^l$ in the pitch direction and $-\kappa_2^l - \kappa_3^l$ in the roll direction. We need to set them equal in each direction respectively. The same logic applies to the left hip link in the yaw direction and the right leg as well.

2.3.2 Whole-Body Kinematics

We now solve the robot whole-body kinematics relative to the world frame $\{w\}$ in consideration of the floating base motion (the presuperscript w will always be omitted for readability). Before we start, let us define the generalized coordinates, velocities, and accelerations:

$$\boldsymbol{q} = \begin{bmatrix} \log\left(\boldsymbol{R}_b\right)^\vee \\ \boldsymbol{p}_b \\ \boldsymbol{\theta}^r \\ \boldsymbol{\theta}^l \end{bmatrix}, \quad \dot{\boldsymbol{q}} = \begin{bmatrix} {}^b\boldsymbol{\omega}_b \\ {}^b\boldsymbol{v}_b \\ \dot{\boldsymbol{\theta}}^r \\ \dot{\boldsymbol{\theta}}^l \end{bmatrix}, \quad \ddot{\boldsymbol{q}} = \begin{bmatrix} {}^b\boldsymbol{\alpha}_b \\ {}^b\boldsymbol{a}_b \\ \ddot{\boldsymbol{\theta}}^r \\ \ddot{\boldsymbol{\theta}}^l \end{bmatrix}, \tag{2.120}$$

where ${}^b\boldsymbol{\omega}_b$ and ${}^b\boldsymbol{v}_b$ are the angular and linear velocity of the floating base described in its own frame, ${}^b\boldsymbol{\alpha}_b$ and ${}^b\boldsymbol{a}_b$ are thus the corresponding angular and linear acceleration. Note that the generalized velocities and accelerations do not need to be the strict time derivatives of the generalized coordinates for the floating base part.

Given the base position $\boldsymbol{p}_b$ and orientation $\boldsymbol{R}_b$ (which can be estimated in Section 4.3), we can compute the foot position in the world frame first:

$$\boldsymbol{p}_f^{leg} = \boldsymbol{p}_b + \boldsymbol{R}_b {}^b\boldsymbol{p}_f^{leg}. \tag{2.121}$$

Its velocity can be derived by taking the time derivative:

$$\begin{aligned}
\dot{\boldsymbol{p}}_f^{leg} &= \dot{\boldsymbol{p}}_b + \dot{\boldsymbol{R}}_b {}^b\boldsymbol{p}_f^{leg} + \boldsymbol{R}_b {}^b\dot{\boldsymbol{p}}_f^{leg} && (2.122)\\
&= \boldsymbol{v}_b + \boldsymbol{R}_b \widehat{{}^b\boldsymbol{\omega}_b} {}^b\boldsymbol{p}_f^{leg} + \boldsymbol{R}_b {}^b_v\boldsymbol{J}_f^{leg}\dot{\boldsymbol{\theta}}^{leg} && (2.123)\\
&= \boldsymbol{R}_b \left({}^b\boldsymbol{v}_b - \widehat{{}^b\boldsymbol{p}_f^{leg}} {}^b\boldsymbol{\omega}_b + {}^b_v\boldsymbol{J}_f^{leg}\dot{\boldsymbol{\theta}}^{leg} \right) && (2.124)\\
\Rightarrow \dot{\boldsymbol{p}}_f^r &= \underbrace{\boldsymbol{R}_b \begin{bmatrix} -\widehat{{}^b\boldsymbol{p}_f^r} & \mathbb{I} & {}^b_v\boldsymbol{J}_f^r & \boldsymbol{0} \end{bmatrix}}_{{}_v\boldsymbol{J}_f^r} \dot{\boldsymbol{q}} = {}_v\boldsymbol{J}_f^r\dot{\boldsymbol{q}}, && (2.125)\\
\dot{\boldsymbol{p}}_f^l &= \underbrace{\boldsymbol{R}_b \begin{bmatrix} -\widehat{{}^b\boldsymbol{p}_f^l} & \mathbb{I} & \boldsymbol{0} & {}^b_v\boldsymbol{J}_f^l \end{bmatrix}}_{{}_v\boldsymbol{J}_f^l} \dot{\boldsymbol{q}} = {}_v\boldsymbol{J}_f^l\dot{\boldsymbol{q}}. && (2.126)
\end{aligned}$$

A second time derivative gives the acceleration:

$$\begin{aligned}
\ddot{\boldsymbol{p}}_f^r &= {}_v\boldsymbol{J}_f^r\ddot{\boldsymbol{q}} + {}_v\dot{\boldsymbol{J}}_f^r\dot{\boldsymbol{q}}, && (2.127)\\
\ddot{\boldsymbol{p}}_f^l &= {}_v\boldsymbol{J}_f^l\ddot{\boldsymbol{q}} + {}_v\dot{\boldsymbol{J}}_f^l\dot{\boldsymbol{q}}, && (2.128)
\end{aligned}$$

where

$$\begin{aligned}
{}_v\dot{\boldsymbol{J}}_f^r &= \dot{\boldsymbol{R}}_b \begin{bmatrix} -\widehat{{}^b\boldsymbol{p}_f^r} & \mathbb{I} & {}^b_v\boldsymbol{J}_f^r & \boldsymbol{0} \end{bmatrix} + \boldsymbol{R}_b \begin{bmatrix} -\dot{\widehat{{}^b\boldsymbol{p}_f^r}} & \boldsymbol{0} & {}^b_v\dot{\boldsymbol{J}}_f^r & \boldsymbol{0} \end{bmatrix} && (2.129)\\
&= \boldsymbol{R}_b \left(\widehat{{}^b\boldsymbol{\omega}_b} \begin{bmatrix} -\widehat{{}^b\boldsymbol{p}_f^r} & \mathbb{I} & {}^b_v\boldsymbol{J}_f^r & \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} -\widehat{{}^b_v\boldsymbol{J}_f^r\dot{\boldsymbol{\theta}}^r} & \boldsymbol{0} & {}^b_v\dot{\boldsymbol{J}}_f^r & \boldsymbol{0} \end{bmatrix} \right) && (2.130)\\
&= \boldsymbol{R}_b \begin{bmatrix} -\widehat{{}^b\boldsymbol{\omega}_b}\widehat{{}^b\boldsymbol{p}_f^r} - \widehat{{}^b_v\boldsymbol{J}_f^r\dot{\boldsymbol{\theta}}^r} & \widehat{{}^b\boldsymbol{\omega}_b} & \widehat{{}^b\boldsymbol{\omega}_b} {}^b_v\boldsymbol{J}_f^r + {}^b_v\dot{\boldsymbol{J}}_f^r & \boldsymbol{0} \end{bmatrix}, && (2.131)\\
{}_v\dot{\boldsymbol{J}}_f^l &= \boldsymbol{R}_b \begin{bmatrix} -\widehat{{}^b\boldsymbol{\omega}_b}\widehat{{}^b\boldsymbol{p}_f^l} - \widehat{{}^b_v\boldsymbol{J}_f^l\dot{\boldsymbol{\theta}}^l} & \widehat{{}^b\boldsymbol{\omega}_b} & \boldsymbol{0} & \widehat{{}^b\boldsymbol{\omega}_b} {}^b_v\boldsymbol{J}_f^l + {}^b_v\dot{\boldsymbol{J}}_f^l \end{bmatrix}. && (2.132)
\end{aligned}$$

The foot orientation can be determined to be

$$\boldsymbol{R}_f^{leg} = \boldsymbol{R}_b \, {}^b\boldsymbol{R}_f^{leg}. \tag{2.133}$$

Its angular velocity is simply

$$\begin{aligned}
\boldsymbol{\omega}_f^{leg} &= \boldsymbol{R}_b \left({}^b\boldsymbol{\omega}_b + {}^b\boldsymbol{\omega}_f^{leg} \right) && (2.134)\\
&= \boldsymbol{R}_b \left({}^b\boldsymbol{\omega}_b + {}_{\omega}{}^b\boldsymbol{J}_f^{leg} \dot{\boldsymbol{\theta}}^{leg} \right) && (2.135)\\
\Rightarrow \boldsymbol{\omega}_f^r &= \underbrace{\boldsymbol{R}_b \begin{bmatrix} \mathbb{I} & \mathbf{0} & {}_{\omega}{}^b\boldsymbol{J}_f^r & \mathbf{0} \end{bmatrix}}_{{}_{\omega}\boldsymbol{J}_f^r} \dot{\boldsymbol{q}} = {}_{\omega}\boldsymbol{J}_f^r \dot{\boldsymbol{q}}, && (2.136)\\
\boldsymbol{\omega}_f^l &= \underbrace{\boldsymbol{R}_b \begin{bmatrix} \mathbb{I} & \mathbf{0} & \mathbf{0} & {}_{\omega}{}^b\boldsymbol{J}_f^l \end{bmatrix}}_{{}_{\omega}\boldsymbol{J}_f^l} \dot{\boldsymbol{q}} = {}_{\omega}\boldsymbol{J}_f^l \dot{\boldsymbol{q}}. && (2.137)
\end{aligned}$$

Its angular acceleration is thus

$$\begin{aligned}
\boldsymbol{\alpha}_f^r &= {}_{\omega}\boldsymbol{J}_f^r \ddot{\boldsymbol{q}} + {}_{\omega}\dot{\boldsymbol{J}}_f^r \dot{\boldsymbol{q}}, && (2.138)\\
\boldsymbol{\alpha}_f^l &= {}_{\omega}\boldsymbol{J}_f^l \ddot{\boldsymbol{q}} + {}_{\omega}\dot{\boldsymbol{J}}_f^l \dot{\boldsymbol{q}}, && (2.139)
\end{aligned}$$

where

$$\begin{aligned}
{}_{\omega}\dot{\boldsymbol{J}}_f^r &= \dot{\boldsymbol{R}}_b \begin{bmatrix} \mathbb{I} & \mathbf{0} & {}_{\omega}{}^b\boldsymbol{J}_f^r & \mathbf{0} \end{bmatrix} + \boldsymbol{R}_b \begin{bmatrix} \mathbf{0} & \mathbf{0} & {}_{\omega}{}^b\dot{\boldsymbol{J}}_f^r & \mathbf{0} \end{bmatrix} && (2.140)\\
&= \boldsymbol{R}_b \left(\widehat{{}^b\boldsymbol{\omega}_b} \begin{bmatrix} \mathbb{I} & \mathbf{0} & {}_{\omega}{}^b\boldsymbol{J}_f^r & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & {}_{\omega}{}^b\dot{\boldsymbol{J}}_f^r & \mathbf{0} \end{bmatrix} \right) && (2.141)\\
&= \boldsymbol{R}_b \begin{bmatrix} \cancel{\widehat{{}^b\boldsymbol{\omega}_b}} & \mathbf{0} & \widehat{{}^b\boldsymbol{\omega}_b} \, {}_{\omega}{}^b\boldsymbol{J}_f^r + {}_{\omega}{}^b\dot{\boldsymbol{J}}_f^r & \mathbf{0} \end{bmatrix}, && (2.142)\\
{}_{\omega}\dot{\boldsymbol{J}}_f^l &= \boldsymbol{R}_b \begin{bmatrix} \cancel{\widehat{{}^b\boldsymbol{\omega}_b}} & \mathbf{0} & \mathbf{0} & \widehat{{}^b\boldsymbol{\omega}_b} \, {}_{\omega}{}^b\boldsymbol{J}_f^l + {}_{\omega}{}^b\dot{\boldsymbol{J}}_f^l \end{bmatrix}. && (2.143)
\end{aligned}$$

Note that the first block $\widehat{{}^b\boldsymbol{\omega}_b}$ can be removed since $\widehat{{}^b\boldsymbol{\omega}_b} \, {}^b\boldsymbol{\omega}_b = \mathbf{0}$. In addition, we also need

the foot angular velocity Jacobian in its local foot frame and thus we have

$$ {}^{f}_{\omega}\boldsymbol{J}_{f}^{leg} = {}^{f}\boldsymbol{R}_{w}^{leg}\,{}_{\omega}\boldsymbol{J}_{f}^{leg}, \tag{2.144} $$

$$ {}^{f}_{\omega}\dot{\boldsymbol{J}}_{f}^{leg} = {}^{f}\boldsymbol{R}_{w}^{leg}\,{}_{\omega}\dot{\boldsymbol{J}}_{f}^{leg}. \tag{2.145} $$

2.3.3 Kinematics Verification

Several tests are conducted to verify the robot kinematic model.

Posture Setup

The robot posture can be fully defined by first specifying the transformation matrices of the base frame and the two foot frames relative to the world frame:

$$ \boldsymbol{T}_{b} = \begin{bmatrix} \boldsymbol{R}_{b} & \boldsymbol{p}_{b} \\ \boldsymbol{0} & 1 \end{bmatrix}, \quad \boldsymbol{T}_{f}^{leg} = \begin{bmatrix} \boldsymbol{R}_{f}^{leg} & \boldsymbol{p}_{f}^{leg} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{2.146} $$

We can then describe the foot frame relative to the base frame:

$$ {}^{b}\boldsymbol{T}_{f}^{leg} = \boldsymbol{T}_{b}^{-1}\boldsymbol{T}_{f}^{leg}. \tag{2.147} $$

Once we have the foot position and direction relative to the base frame, we can use the analytical IK solution (2.113) to solve for the corresponding joint angles. Note that the nominal standing posture of BRUCE is designed in this way.

Seesaw Balancing

Note that this is only a naive balancing strategy. It is not designed for push recovery, which requires robot dynamic information. We are only trying to keep the base frame fixed based solely on the kinematic model whenever the seesaw terrain is changing. Also, we are only considering the variation in the seesaw orientation for now, i.e., BRUCE will be

commanded to keep its torso in the upright configuration when the seesaw is moving while the translational effect will just be disregarded (well it is problematic to measure the absolute position of the seesaw).

Let BRUCE start with the nominal standing posture on a seesaw. We can specify $\boldsymbol{T}_b^{ref}$ which is our reference. We need to find out $\boldsymbol{T}_f^{leg}$ such that when the feet move with the seesaw $\boldsymbol{T}_b$ is maintained (and we can then use IK again to get the corresponding joint angles). We can further decompose $\boldsymbol{T}_f^{leg}$ into two parts:

$$\boldsymbol{T}_f^{leg} = \boldsymbol{T}_s \, {}^s\boldsymbol{T}_f^{leg}, \tag{2.148}$$

where $\boldsymbol{T}_s$ describes the seesaw frame $\{s\}$ relative to the world frame which is unknown and varying while ${}^s\boldsymbol{T}_f^{leg}$ describes the foot frame relative to the seesaw frame which is known and fixed (assuming no slipping). Since we do not care about $\boldsymbol{p}_s$, it is zeroed for simplicity, i.e., $\boldsymbol{p}_s = \boldsymbol{0}$. If we further have the direct information of $\boldsymbol{R}_s$, the problem is trivial, e.g., we can mount an IMU on the seesaw for orientation measurement. However, to make the problem even more interesting ☺, we estimate the seesaw orientation using the on-board IMU with a PID control [168] on $\mathsf{SO}(3)$ (see Appendix C).

Suppose we have a small variation in base orientation at the kth time step:

$$\boldsymbol{e}_k = \mathrm{Log}\left(\boldsymbol{R}_{b,k}^\top \boldsymbol{R}_b^{ref}\right), \tag{2.149}$$

where $\boldsymbol{R}_{b,k}$ is the current base orientation. The seesaw orientation is then estimated as

$$\boldsymbol{R}_{s,k} = \boldsymbol{R}_{s,0}\mathrm{Exp}\left(\boldsymbol{K}_p\boldsymbol{e}_k + \boldsymbol{K}_i\boldsymbol{\mathfrak{e}}_k - \boldsymbol{K}_d \, {}^b\boldsymbol{\omega}_{b,k}\right), \tag{2.150}$$

$$\boldsymbol{\mathfrak{e}}_k = \boldsymbol{\mathfrak{e}}_{k-1} + \left(\boldsymbol{K}_p\boldsymbol{e}_k - \boldsymbol{K}_d \, {}^b\boldsymbol{\omega}_{b,k}\right)\Delta t, \quad \boldsymbol{\mathfrak{e}}_0 = \boldsymbol{0}, \tag{2.151}$$

where $\boldsymbol{K}_p$, $\boldsymbol{K}_i$, $\boldsymbol{K}_d$ are the diagonal positive definite proportional, integral, and derivative gain matrices, respectively, which can be experimentally tuned, Δt is the control frequency,

and ${}^{b}\boldsymbol{\omega}_{b,k}$ is the current base angular velocity.

Open-Loop Walking

A preliminary walking experiment is also carried out to show the fundamental locomotion capability of BRUCE. The walking pattern generator is both kinematically and dynamically consistent, using MPC for the robot CoM based on LIP [122] (we will go over the LIP model in detail later in Section 3.2.1) and cycloidal interpolation for the swing leg motion [134]. The planned walking trajectory is then tracked using IK again.

Suppose we have N slices of body yaw angle ψ_k^{ref} and position $\boldsymbol{p}_{b,k}^{ref} = \left[x_k^{ref}, y_k^{ref}\right]^\top$, $k = 1, \ldots, N$, with $\psi_1^{ref} = 0$, $x_1^{ref} = y_1^{ref} = 0$. Note that we are only considering the horizontal motion with a fixed CoM height. Given the desired step changes Δx_k, Δy_k, $\Delta\psi_k$ in the robot body frame, we can determine the body yaw angle and position reference as follows

$$\psi_{k+1}^{ref} = \psi_k^{ref} + \Delta\psi_k, \tag{2.152}$$

$$\boldsymbol{p}_{b,k+1}^{ref} = \boldsymbol{p}_{b,k}^{ref} + \boldsymbol{R}_z(\psi_k^{ref}) \begin{bmatrix} \Delta x_k \\ \Delta y_k \end{bmatrix}, \tag{2.153}$$

where $\boldsymbol{R}_z$ is the planar rotation matrix. We can further determine the foothold locations

$$\boldsymbol{p}_{r,k}^{ref} = \boldsymbol{p}_{b,k}^{ref} + \boldsymbol{R}_z(\psi_k^{ref}) \begin{bmatrix} 0 \\ d/2 \end{bmatrix}, \tag{2.154}$$

$$\boldsymbol{p}_{l,k}^{ref} = \boldsymbol{p}_{b,k}^{ref} - \boldsymbol{R}_z(\psi_k^{ref}) \begin{bmatrix} 0 \\ d/2 \end{bmatrix}, \tag{2.155}$$

where d is the nominal foot lateral distance. Suppose we start with swinging the right leg

and thus the CoP locations are determined to be

$$\boldsymbol{u}_k^{ref} = \begin{cases} \boldsymbol{p}_{l,k}^{ref} & \text{if } k \text{ is odd,} \\ \boldsymbol{p}_{r,k}^{ref} & \text{if } k \text{ is even.} \end{cases} \tag{2.156}$$

For each step we have

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}_{T_d} \left(\boldsymbol{A}_{T_s} \boldsymbol{x}_k + \boldsymbol{B}_{T_s} \boldsymbol{u}_k^{ref} \right), \tag{2.157}$$

where the state $\boldsymbol{x}_k = \left[\boldsymbol{p}_{b,k}, \dot{\boldsymbol{p}}_{b,k} \right]^\top$, the single-stance transition matrices

$$\boldsymbol{A}_{T_s} = \begin{bmatrix} \cosh(\omega T_s) & 0 & \sinh(\omega T_s)/\omega & 0 \\ 0 & \cosh(\omega T_s) & 0 & \sinh(\omega T_s)/\omega \\ \omega \sinh(\omega T_s) & 0 & \cosh(\omega T_s) & 0 \\ 0 & \omega \sinh(\omega T_s) & 0 & \cosh(\omega T_s) \end{bmatrix}, \tag{2.158}$$

$$\boldsymbol{B}_{T_s} = \begin{bmatrix} 1 - \cosh(\omega T_s) & 0 \\ 0 & 1 - \cosh(\omega T_s) \\ -\omega \sinh(\omega T_s) & 0 \\ 0 & -\omega \sinh(\omega T_s) \end{bmatrix}, \tag{2.159}$$

the double-stance state-transition matrix

$$\boldsymbol{\Phi}_{T_d} = \begin{bmatrix} 1 & 0 & T_d & 0 \\ 0 & 1 & 0 & T_d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.160}$$

assuming to preserve a constant velocity after each single-stance phase, T_s is the single-stance duration, T_d is the double-stance duration, and ω is the natural frequency of LIP. We can

further add an extra double-stance phase before the first step as well as after the final step to regulate the initial and final CoM states:

$$\boldsymbol{x}_1 = \boldsymbol{A}_{T_d}\boldsymbol{x}_0 + \boldsymbol{B}_{T_d}\boldsymbol{u}_0, \tag{2.161}$$

$$\boldsymbol{x}_{N+1} = \boldsymbol{A}_{T_d}\boldsymbol{x}_N + \boldsymbol{B}_{T_d}\boldsymbol{u}_N. \tag{2.162}$$

With the initial and final CoM states specified, e.g., $\boldsymbol{p}_{b,0} = \boldsymbol{p}_{b,1}^{ref}$, $\boldsymbol{p}_{b,N+1} = \boldsymbol{p}_{b,N}^{ref}$, and $\dot{\boldsymbol{p}}_{b,0} = \dot{\boldsymbol{p}}_{b,N+1} = \boldsymbol{0}$, we can solve for the rest:

$$\left[\begin{array}{cccccccc}
\boldsymbol{A}_T & -\mathbb{I} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{A}_T & -\mathbb{I} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_T & -\mathbb{I} & \boldsymbol{0} & \boldsymbol{0} & \ddots & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{A}_T & -\mathbb{I} & \boldsymbol{0} & \boldsymbol{0} \\
\hline
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_{T_d} & \boldsymbol{0} & \boldsymbol{B}_{T_d} \\
-\mathbb{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_{T_d} & \boldsymbol{0}
\end{array}\right]
\left[\begin{array}{c}
\boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \\ \vdots \\ \boldsymbol{x}_N \\ \hline \boldsymbol{u}_0 \\ \boldsymbol{u}_N
\end{array}\right] =
\left[\begin{array}{c}
-\boldsymbol{B}_T\boldsymbol{u}_1^{ref} \\ -\boldsymbol{B}_T\boldsymbol{u}_2^{ref} \\ -\boldsymbol{B}_T\boldsymbol{u}_3^{ref} \\ \vdots \\ -\boldsymbol{B}_T\boldsymbol{u}_N^{ref} \\ \hline \boldsymbol{x}_{N+1} \\ -\boldsymbol{A}_{T_d}\boldsymbol{x}_0
\end{array}\right], \tag{2.163}$$

where the matrices

$$\boldsymbol{A}_T = \boldsymbol{\Phi}_{T_d}\boldsymbol{A}_{T_s}, \tag{2.164}$$

$$\boldsymbol{B}_T = \boldsymbol{\Phi}_{T_d}\boldsymbol{B}_{T_s}. \tag{2.165}$$

Figure 2.8 shows the walking trajectory with an average forward velocity of 0.1 m/s and a yaw rate of 0.1 rad/s for a CoM height of 0.3 m, $d = 0.1$ m, $T_s = 0.3$ s, and $T_d = 0.1$ s.

2.4 Dynamics

Dynamics is the study of the forces required to cause the motion of the subject. In particular, we consider the equations of motion for a robot – the way in which motion of the robot arises

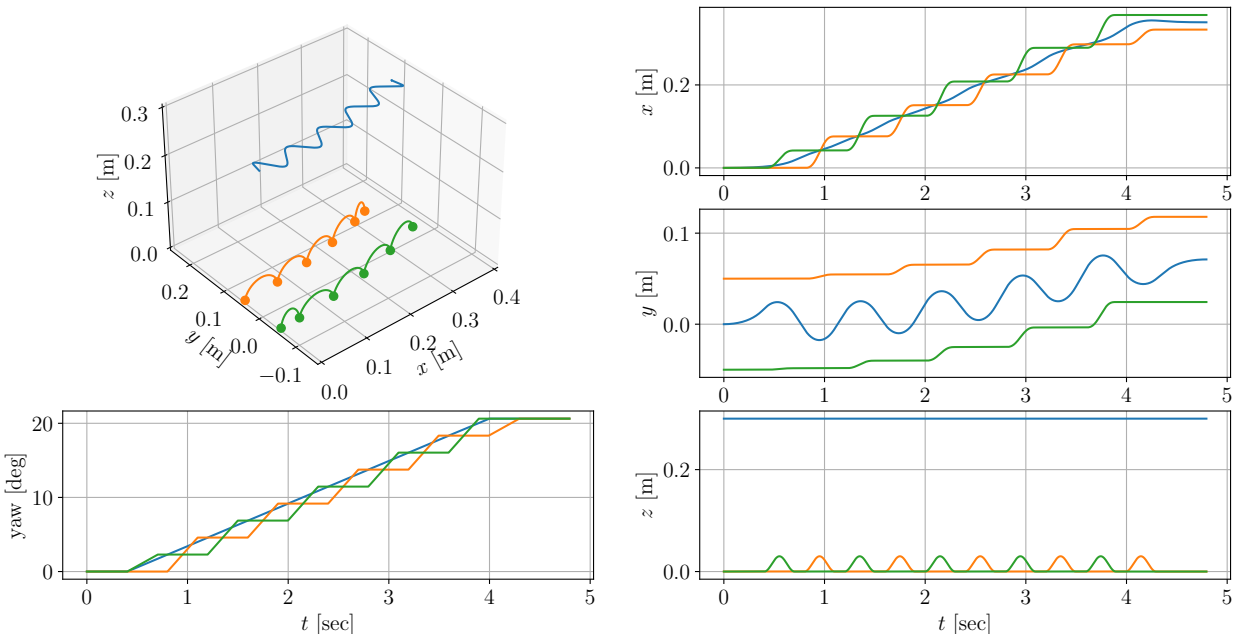


Figure 2.8: Open-loop walking trajectory.

from torques applied by the actuators or from external forces applied to the robot. There are two problems related to the dynamics of a robot. The first one is forward dynamics (FD) which is to calculate how the robot will move under application of a set of joint forces. This is usually useful for simulating the robot. The second one is inverse dynamics (ID) which is to find the required joint forces given a trajectory point. This formulation of dynamics is useful for controlling the robot. We will focus on ID here.

2.4.1 Inverse Dynamics

Given the generalized coordinates of BRUCE (2.120) yet without considering the external forces, the joint-space equations of motion can be written in the following canonical form:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{2.166}$$

where $\boldsymbol{H}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ is the vector of centrifugal and Coriolis terms, $\boldsymbol{G}(\boldsymbol{q})$ is the gravity vector, and $\boldsymbol{\tau}$ is the joint torque vector. Each element of $\boldsymbol{H}(\boldsymbol{q})$ and $\boldsymbol{G}(\boldsymbol{q})$

is a complex function that depends on $\boldsymbol{q}$. Each element of $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a complex function of both $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. Once these dependencies are understood, the shorter symbols $\boldsymbol{H}$, $\boldsymbol{C}$, and $\boldsymbol{G}$ are used. While the analytical solution of them is almost impossible to derive (the equations will be several pages long at best), we have many numerical algorithms which can run efficiently in practice.

Recursive Newton-Euler Algorithm

The computation of $\boldsymbol{C}\dot{\boldsymbol{q}}$ and $\boldsymbol{G}$ can be done using ID based on the recursive Newton-Euler algorithm (RNEA). RNEA is specifically to calculate ID of a kinematic tree (in comparison to closed-loop systems) and it is the simplest, most efficient algorithm for the job. It has a computational complexity of $O(n)$, which means that the amount of calculation grows linearly with the number of bodies, or joint variables, in the tree. RNEA in body coordinates can be summarized as follows:

$$\boldsymbol{v}_0 = \boldsymbol{0}, \tag{2.167}$$

$$\boldsymbol{a}_0 = \boldsymbol{g}, \tag{2.168}$$

$$\boldsymbol{v}_{\mathrm{J}i} = \boldsymbol{S}_i \dot{\boldsymbol{q}}_i, \tag{2.169}$$

$$\boldsymbol{a}_{\mathrm{J}i} = \boldsymbol{S}_i \ddot{\boldsymbol{q}}_i, \tag{2.170}$$

$$\boldsymbol{v}_i = {}^i\boldsymbol{X}_{\lambda(i)} \boldsymbol{v}_{\lambda(i)} + \boldsymbol{v}_{\mathrm{J}i}, \tag{2.171}$$

$$\boldsymbol{a}_i = {}^i\boldsymbol{X}_{\lambda(i)} \boldsymbol{a}_{\lambda(i)} + \boldsymbol{a}_{\mathrm{J}i} + \boldsymbol{v}_i \times \boldsymbol{v}_{\mathrm{J}i}, \tag{2.172}$$

$$\boldsymbol{f}_i^B = \boldsymbol{I}_i \boldsymbol{a}_i + \boldsymbol{v}_i \times^* \boldsymbol{I}_i \boldsymbol{v}_i, \tag{2.173}$$

$$\boldsymbol{f}_i = \boldsymbol{f}_i^B + \sum_{j \in \mu(i)} {}^i\boldsymbol{X}_j^* \boldsymbol{f}_j, \tag{2.174}$$

$$\boldsymbol{\tau}_i = \boldsymbol{S}_i^\top \boldsymbol{f}_i, \tag{2.175}$$

where $\boldsymbol{g} = [0, 0, -9.81]^\top$ m/s^2 is the gravitational acceleration, $\boldsymbol{S}_i$ is the joint motion subspace vector which is equal to $[0, 0, 1, 0, 0, 0]^\top$ for revolute joint, $\boldsymbol{v}_i$ and $\boldsymbol{a}_i$ are the spatial velocity

and acceleration of body i, $\boldsymbol{v}_{\mathrm{J}i}$ and $\boldsymbol{a}_{\mathrm{J}i}$ are the spatial velocity and acceleration across joint i, $\boldsymbol{I}_i$ is the spacial inertia, $\boldsymbol{f}_i^B$ is the net force acting on body i, $\boldsymbol{f}_i$ is the force transmitted from body $\lambda(i)$ to body i across joint i,

$$ {}^i\boldsymbol{X}_{\lambda(i)} = \begin{bmatrix} {}^i\boldsymbol{R}_{\lambda(i)} & \boldsymbol{0} \\ -{}^i\boldsymbol{R}_{\lambda(i)}\,\widehat{{}^i\boldsymbol{p}_{\lambda(i)}} & {}^i\boldsymbol{R}_{\lambda(i)} \end{bmatrix} \Rightarrow {}^i\boldsymbol{X}^*_{\lambda(i)} = {}^i\boldsymbol{X}^{-\top}_{\lambda(i)} = \begin{bmatrix} {}^i\boldsymbol{R}_{\lambda(i)} & -{}^i\boldsymbol{R}_{\lambda(i)}\,\widehat{{}^i\boldsymbol{p}_{\lambda(i)}} \\ \boldsymbol{0} & {}^i\boldsymbol{R}_{\lambda(i)} \end{bmatrix} \tag{2.176} $$

is the coordinate transform from $\lambda(i)$ to i, $\lambda(i)$ is the parent of body i, $\mu(i)$ is the set of children of body i, and $\times$ with its dual $\times^*$ is the spatial cross product. More detailed information is available in [65].

Now that we have an ID calculation function, **ID**, which performs the calculation

$$ \boldsymbol{\tau} = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{2.177} $$

On comparing this expression with (2.166), it follows immediately that

$$ \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G} = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{0}), \tag{2.178} $$

$$ \boldsymbol{G} = \mathbf{ID}(\boldsymbol{q}, \boldsymbol{0}, \boldsymbol{0}). \tag{2.179} $$

The ith column of $\boldsymbol{H}$ can also be computed as follows:

$$ \boldsymbol{H}_i = \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\delta}_i) - \mathbf{ID}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{0}), \tag{2.180} $$

where $\boldsymbol{\delta}_i$ is a column vector having a 1 in its ith element and zeros elsewhere. The algorithm has the advantage of being simple and straightforward. However, it is not the most efficient way to calculate $\boldsymbol{H}$. The fastest algorithm for that job is the composite-rigid-body algorithm (CRBA), which is the subject of the following subsubsection.

Composite-Rigid-Body Algorithm

CRBA has a computational complexity of $O(nd)$, where d is the depth of the kinematic tree. CRBA in body coordinates can be summarized as follows:

$$\boldsymbol{I}_i^c = \boldsymbol{I}_i + \sum_{j \in \mu(i)} {}^i\boldsymbol{X}_j^* \, \boldsymbol{I}_j^c \, {}^j\boldsymbol{X}_i, \tag{2.181}$$

$${}^{\lambda(i)}\boldsymbol{F}_i = {}^{\lambda(i)}\boldsymbol{X}_j^* \, {}^j\boldsymbol{F}_i, \tag{2.182}$$

$${}^i\boldsymbol{F}_i = \boldsymbol{I}_i^c \, \boldsymbol{S}_i, \tag{2.183}$$

$$\boldsymbol{H}_{ij} = \begin{cases} {}^j\boldsymbol{F}_i^\top \boldsymbol{S}_j & \text{if } i \in \nu(j), \\ \boldsymbol{H}_{ji} & \text{if } j \in \nu(i), \\ \boldsymbol{0} & \text{otherwise,} \end{cases} \tag{2.184}$$

where $\boldsymbol{I}_i^c$ is the inertia of the subtree rooted at body i, treated as a single composite rigid body (this is where the algorithm gets its name), ${}^j\boldsymbol{F}_i$ is the value of $\boldsymbol{I}_i^c \, \boldsymbol{S}_i$ expressed in body j coordinates (i.e., ${}^j\boldsymbol{F}_i = {}^j\boldsymbol{X}_i^* \, \boldsymbol{I}_i^c \, \boldsymbol{S}_i$), and $\nu(i)$ is the set of bodies in the subtree starting at body i (or the set of all bodies supported by joint i). More information is available in [65].

2.4.2 Centroidal Momentum Dynamics

The control of centroidal momentum is important for humanoid robots, which consists of the net linear momentum $\boldsymbol{l}_G$ as well as the net angular momentum $\boldsymbol{k}_G$ about the robot CoM. While the linear part has a well-known relationship with the CoM velocity $\boldsymbol{v}_G$:

$$\boldsymbol{l}_G = M\boldsymbol{v}_G, \tag{2.185}$$

the angular component is abstract as it represents the sum of the angular momenta of each individual body. As a result, properly designing the angular momentum trajectory is still an open problem. Fortunately, biomechanics studies of human walking have shown that

although humans have large nonzero angular momenta for individual bodies in the limbs, yet, the neuro-control system coordinates significant inter-segmental momentum cancelations, regulating their centroidal angular momentum to near zero [67].

To effectively control the centroidal momentum, the centroidal momentum matrix (CMM) $\boldsymbol{A}_G(\boldsymbol{q})$ and its derivative $\dot{\boldsymbol{A}}_G(\boldsymbol{q})$ are required, which relate the centroidal momentum $\boldsymbol{h}_G = \left[\boldsymbol{k}_G^\top, \boldsymbol{l}_G^\top\right]^\top$ and its rate of change $\dot{\boldsymbol{h}}_G$ to the generalized velocity and acceleration of the robot:

$$\boldsymbol{h}_G = \boldsymbol{A}_G \dot{\boldsymbol{q}}, \tag{2.186}$$

$$\dot{\boldsymbol{h}}_G = \boldsymbol{A}_G \ddot{\boldsymbol{q}} + \dot{\boldsymbol{A}}_G \dot{\boldsymbol{q}}. \tag{2.187}$$

The computation of $\boldsymbol{A}_G$ and $\dot{\boldsymbol{A}}_G\dot{\boldsymbol{q}}$ can be done from knowledge of the floating-base kinematics, the inertia matrix $\boldsymbol{H}$, and the the Coriolis term $\boldsymbol{C}\dot{\boldsymbol{q}}$ alone:

$$\boldsymbol{I}_b^c = \boldsymbol{S}_b \boldsymbol{H} \boldsymbol{S}_b^\top = \begin{bmatrix} \bar{\boldsymbol{I}}_b^c & M\widehat{{}^b\boldsymbol{p}_G} \\ -M\widehat{{}^b\boldsymbol{p}_G} & M\mathbb{I} \end{bmatrix}, \tag{2.188}$$

$${}^b\boldsymbol{X}_G^\top = \begin{bmatrix} \boldsymbol{R}_b & -\boldsymbol{R}_b\widehat{{}^b\boldsymbol{p}_G} \\ \boldsymbol{0} & \boldsymbol{R}_b \end{bmatrix}, \tag{2.189}$$

$$\boldsymbol{A}_G = {}^b\boldsymbol{X}_G^\top \boldsymbol{S}_b \boldsymbol{H}, \tag{2.190}$$

$$\dot{\boldsymbol{A}}_G\dot{\boldsymbol{q}} = {}^b\boldsymbol{X}_G^\top \boldsymbol{S}_b \boldsymbol{C}\dot{\boldsymbol{q}}, \tag{2.191}$$

where $\boldsymbol{S}_b$ is the base selection matrix. More detailed information is available in [169].

2.4.3 System Identification

An accurate dynamic model is essential for BRUCE to perform highly dynamic motions since the controller heavily relies on it. Meanwhile, for miniature robots, CAD measurement is less accurate due to the relatively high ratio of the electronics and accessories. Accordingly, mass-inertial parameters are obtained by performing a system identification which is formulated

as a least-squares problem with the joint measurement along some excitation trajectories [170]. However, ill-conditioned observation matrix can lead to inaccurate identification. To mitigate this issue, an optimal excitation trajectory is generated by minimizing the condition number of the observation matrix based on a parameterized trajectory with finite fourier series [171, 172], while regularization towards the nominal values obtained from the CAD model is also considered. The regularized least-squares problem is formulated as follows:

$$\min_{\boldsymbol{\Psi}} \ \|\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{\Psi} - \boldsymbol{\tau}\| + \lambda\, R(\boldsymbol{\Psi}, \boldsymbol{\Psi}_0)\,, \tag{2.192}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\tau}$ are respectively the measured joint positions, velocities, accelerations, torques, $\boldsymbol{\Gamma}$ is the state-dependant observation matrix, and $\boldsymbol{\Psi} \in \mathbb{R}^{10n}$ consists of the mass-inertial parameters for an n-link system including mass, first mass moment, and rotational inertia tensor [170]. As for the regularization term R with the scalar weight λ, the formulation in [173] is adopted to make the difference between $\boldsymbol{\Psi}$ and its nominal quantity $\boldsymbol{\Psi}_0$ physically consistent but still convex.

2.5 Conclusion

This chapter presents the early development of BRUCE, a next-generation miniature bipedal robot. With the designed differential cable-driven pulley system and linkage mechanism, the distribution of leg mass and inertia is optimized in favor of dynamic behaviors. Proprioceptive actuation and contact sensing further enable BRUCE to safely interact with unstructured environments while providing rich feedback information. Complete modeling and analysis of the robot system, e.g., kinematics, dynamics, system identification, are also carried out. The preliminary testing results verify the basic functionalities of the robot design and explore its dynamic capabilities. The BRUCE project is aimed to become an open-source, cost-effective, accessible, reliable, miniature humanoid robot platform for general research purposes in the near future.

Chapter 3

Divergent Component of Motion Based Analysis and Control of Dynamic Bipedal Locomotion Using Reduced-Order Models

3.1 Introduction

3.1.1 Background

Bipedal locomotion has been studied for decades and yet it remains an active research field. Besides the great demand for a reliable hardware platform, various challenges emerge in developing an efficient control algorithm, e.g., the complex robot dynamics and strict real-time requirements. To date, many successful online locomotion control strategies still utilize the reduced-order models, which only focus on the most salient aspect of the system dynamics, e.g., linear inverted pendulum (LIP) model [68].

In the presence of strong perturbations, when the ankle [113, 174] and hip [73, 175] strategies are no more effective, it is necessary to take recovery steps to avoid falling. While

linear feedback control and simple heuristics work for small footstep adjustment [71, 126], a real-time trajectory optimization (TO) scheme outperforms them with the consideration of a time prediction horizon and various physical constraints.

One of the main progresses in the field has been how the LIP is handled in this TO scheme. Considering piecewise constant center of mass (CoM) jerk as input, the LIP dynamics can be discretized as cubic polynomials. The footstep locations can be optimized through a quadratic program (QP), which minimizes the overall tracking errors while restricting the center of pressure (CoP) within the support polygon [114, 115, 116]. Assuming invariant CoP coincident with the footstep location, the LIP dynamics can be analytically integrated so that the intermediate process can be omitted and thus only an order of magnitude less decision variables are involved in the QP [121, 122]. Nevertheless, in all these strategies, step duration is always fixed to maintain the convexity of the TO problem. Notably, the LIP dynamics can be decomposed into two parts, one is stable and the other is unstable, and controlling only the unstable part, the so-called divergent component of motion (DCM), is enough to generate stable walking [78, 126]. Unlike the full CoM states including both position and velocity, the DCM follows a first-order dynamics, which helps simplify other aspects of locomotion analysis and control. Based on the DCM, both location and timing of the next footstep can be adapted within a convex TO formulation thanks to a suitable change of variable, which leads to even less decision variables [119, 130].

3.1.2 Motivation and Contribution

Though many progresses have been advanced, some problems remain open:

- The latest developments in robot hardware, e.g., proprioceptive actuation [50, 53, 54, 55], have enabled quadrupedal robots to perform highly dynamic motions [157], which will extensively ameliorate bipedal robots sooner or later so that a vast range of dynamic gaits can be achieved not just in simulations, e.g., running and hopping. However, almost all DCM-based locomotion frameworks only concentrate on the walking

scenario without considering a flight phase.

- Compared with walking, the horizontal and vertical CoM dynamics are more coupled in running and hopping, which makes it less consistent with the LIP assuming a constant CoM height during the stance phase. In this respect, the variable-height inverted pendulum [71] and spring-loaded inverted pendulum [79] generalize the LIP by getting rid of the fixed CoM height constraint, which can better describe the overall 3D CoM motion. Unfortunately, they are non-integrable and costly to evaluate, which increases complexity in the TO. It is still debatable to which extent the LIP is effective in modeling highly dynamic bipedal locomotion given that the decoupling of the CoM dynamics has seen success in the control of bipedal running [123, 132]. Notably, in biomechanics studies of human locomotion, Lee and Farley showed that the CoM of their subjects moved upwards by 0.031 m during the first half of the stance phase during walking and moved downwards by 0.073 m during the first half of the stance phase during running [176]. If we say the CoM height is changing between 0.9 and 1.0 m during the stance phase, the natural frequency of the pendulum will only change between 3.13 and 3.30 Hz. The small variance in the pendulum frequency verifies the effectiveness of the LIP in modeling running to some extent.

- When designing TO-based locomotion controllers, we always ask the question: *How many steps ahead shall we plan?* While most of them usually have a preview of adequate steps to ensure gait viability, it can be computationally expensive and unnecessary. Koolen *et al.* hypothesize bipedal robots can usually stop within two steps if ever possible [77]. Zaytsev *et al.* claim in most cases the robot does not perform better by looking more than two steps ahead [177]. Khadiv *et al.* argue it is sufficient to only consider the next step location and timing to ensure walking stability from any viable state [130]. Nonetheless, almost all of them only focus on the gait viability without thinking about other aspects.

This chapter addresses the aforementioned problems and the main contributions are summarized as follows:

- The decoupling of the CoM dynamics is preserved and the work of [119] is extended by further considering a flight phase for each gait cycle. Consequently, DCM-based analysis of both nominal gait pattern and viability condition for the reduced-order models is augmented in closed form. A similar study has been conducted but the results are deficient, e.g., assumption (16) is only valid with a zero nominal lateral CoM velocity [132].

- A corresponding TO-based locomotion control framework is designed, which can be formulated into a small-scale QP even with a horizon of multiple steps being planned ahead of time, so that both step location and timing can be optimally determined in real time.

- The theoretical capabilities of the proposed DCM-based locomotion analysis and control are studied independent of the actual robot. The approach is implemented on the reduced-order models and verified through push recovery simulations. The results also indicate while planning with one step ahead is mostly enough for gait viability, three preview steps is the wise choice in terms of system robustness[1], transient behavior[2], and computational efficiency.

[1]System robustness is defined as the ability to recover from external disturbances.

[2]Transient behavior is defined as the response to a deviation from the nominal gait pattern, e.g., due to external disturbances.

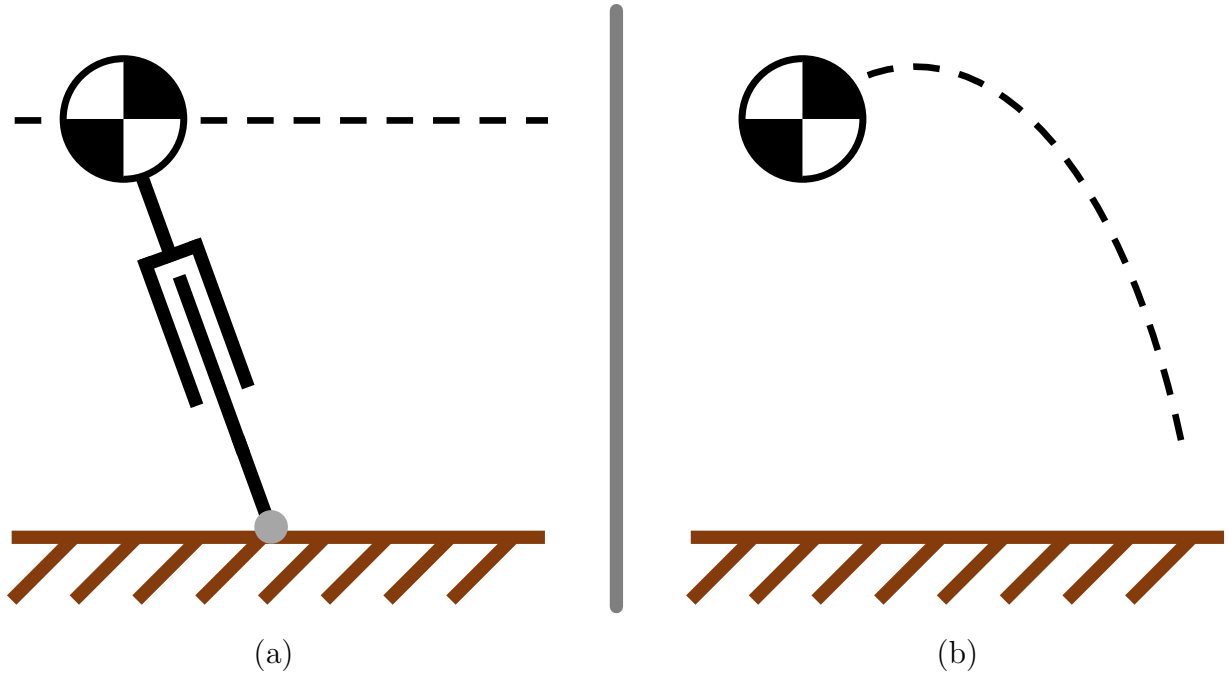


Figure 3.1: Schematic diagram of reduced-order models: (a) LIP during the stance phase and (b) BM during the flight phase.

3.2 Reduced-Order Models

This section elaborates the reduced-order models in order to characterize the evolution of the robot states in a computationally efficient manner. Note that only the horizontal motion is studied. The CoM horizontal and vertical motions have been decoupled for 3D bipedal locomotion analysis and control previously. Therefore, existing vertical results, e.g., [123, 132, 178], can be readily applied if needed in practice.

Generally, a nominal symmetric and periodic gait can be designed by considering two phases in each gait cycle, the stance phase with duration T_s followed by the flight phase with duration T_f. The total duration T of each gait cycle is thus equal to $T_s + T_f$. As illustrated in Figure 3.1, during the stance phase, LIP is used which allows closed-form dynamic integration. During the flight phase, the CoM movement is described by the simple ballistic motion (BM).

3.2.1 Linear Inverted Pendulum Model

Dynamic Equations

Without worrying about torque saturations, the centroidal momentum dynamics efficiently captures the effect of the net external wrench on the evolution of both linear and angular momentum, which is given by

$$\dot{\boldsymbol{l}} = m\ddot{\boldsymbol{c}} = -m\boldsymbol{g} + \sum_{i=1}^{N} \boldsymbol{f}_i, \tag{3.1}$$

$$\dot{\boldsymbol{k}} = \sum_{i=1}^{N} (\boldsymbol{p}_i - \boldsymbol{c}) \times \boldsymbol{f}_i, \tag{3.2}$$

where $\boldsymbol{l}, \boldsymbol{k} \in \mathbb{R}^3$ are respectively the linear and angular momentum, $\boldsymbol{c} = \begin{bmatrix} c_x, c_y, c_z \end{bmatrix}^\top \in \mathbb{R}^3$ is the robot CoM position, $\boldsymbol{f}_i \in \mathbb{R}^3$ is the ith contact force exerted at the corresponding contact location $\boldsymbol{p}_i \in \mathbb{R}^3$, $N \in \mathbb{N}$ is the number of contacts, $\boldsymbol{g} = [0, 0, g]^\top \in \mathbb{R}^3$ is the gravity vector with gravitational acceleration $g = 9.81$ m/s^2, and $m \in \mathbb{R}$ is the robot mass.

For general robot locomotion design synthesis, one can usually simplify the multi-contact scenario with one single equivalent resultant force $\boldsymbol{f} := \sum_{i=1}^{N} \boldsymbol{f}_i$ exerted at the center of pressure (CoP) location $\boldsymbol{p} = \begin{bmatrix} p_x, p_y, p_z \end{bmatrix}^\top \in \mathbb{R}^3$ on the ground:

$$\boldsymbol{p} := \frac{\sum_{i=1}^{N} (\widehat{\boldsymbol{n}} \cdot \boldsymbol{f}_i)\, \boldsymbol{p}_i}{\sum_{i=1}^{N} \widehat{\boldsymbol{n}} \cdot \boldsymbol{f}_i} = \frac{\sum_{i=1}^{N} f_i^n \boldsymbol{p}_i}{\sum_{i=1}^{N} f_i^n}, \tag{3.3}$$

which gives

$$\dot{\boldsymbol{l}} = m\ddot{\boldsymbol{c}} = -m\boldsymbol{g} + \boldsymbol{f}, \tag{3.4}$$

$$\dot{\boldsymbol{k}} = (\boldsymbol{p} - \boldsymbol{c}) \times \boldsymbol{f}. \tag{3.5}$$

Note that we assume the contact points are all on the same ground plane with a common unit normal vector $\widehat{\boldsymbol{n}} \in \mathbb{R}^3$. We thus have

$$(\boldsymbol{p}_i - \boldsymbol{p}) \cdot \widehat{\boldsymbol{n}} = 0, \tag{3.6}$$

and $f_i^n = \widehat{\boldsymbol{n}} \cdot \boldsymbol{f}_i \in \mathbb{R}$ denotes the normal component. With the unilateral constraint of the contact forces as follows:

$$\widehat{\boldsymbol{n}} \cdot \boldsymbol{f}_i = f_i^n \geq 0, \tag{3.7}$$

the CoP must lie within the convex hull of the contact points. We can further verify that the tangential angular momenta of the contact forces with respect to the CoP vanishes:

$$\widehat{\boldsymbol{t}} \cdot \left(\sum_{i=1}^{N} (\boldsymbol{p}_i - \boldsymbol{p}) \times \underbrace{\left(f_i^n \widehat{\boldsymbol{n}} + f_i^{t_i} \widehat{\boldsymbol{t}}_i \right)}_{= \boldsymbol{f}_i} \right) \tag{3.8}$$

$$= \sum_{i=1}^{N} (\boldsymbol{p}_i - \boldsymbol{p}) \cdot \left(f_i^n \underbrace{\left(\widehat{\boldsymbol{n}} \times \widehat{\boldsymbol{t}} \right)}_{= \widehat{\boldsymbol{t}}'} + f_i^{t_i} \underbrace{\left(\widehat{\boldsymbol{t}}_i \times \widehat{\boldsymbol{t}} \right)}_{= a_i \widehat{\boldsymbol{n}}} \right) \tag{3.9}$$

$$= \widehat{\boldsymbol{t}}' \cdot \underbrace{\sum_{i=1}^{N} f_i^n (\boldsymbol{p}_i - \boldsymbol{p})}_{= 0 \text{ per } (3.3)} + \sum_{i=1}^{N} a_i f_i^{t_i} \underbrace{(\boldsymbol{p}_i - \boldsymbol{p}) \cdot \widehat{\boldsymbol{n}}}_{= 0 \text{ per } (3.6)} \tag{3.10}$$

$$= 0, \tag{3.11}$$

where $\widehat{\boldsymbol{t}}$, $\widehat{\boldsymbol{t}}_i \in \mathbb{R}^3$ are the unit tangential vectors and $f_i^{t_i} = \widehat{\boldsymbol{t}}_i \cdot \boldsymbol{f}_i \in \mathbb{R}$ thus denotes the tangential component. As a result, the CoP is also referred to as the Zero-Moment Point (ZMP) [75].

If we further assume the change rate of the centroidal angular momentum is negligible, i.e., $\dot{\boldsymbol{k}} = \boldsymbol{0}$, the resultant force $\boldsymbol{f}$ needs to point towards the CoM according to (3.5), which

indicates $\boldsymbol{f}$ is in the form of

$$\boldsymbol{f} = \omega^2 \cdot m\left(\boldsymbol{c} - \boldsymbol{p}\right), \tag{3.12}$$

where the parameter $\omega \in \mathbb{R}$ is known as the natural frequency of the pendulum. Substituting (3.12) into (3.4) yields

$$\ddot{\boldsymbol{c}} = \omega^2\left(\boldsymbol{c} - \boldsymbol{p}\right) - \boldsymbol{g} \Rightarrow \begin{cases} \ddot{c}_x = \omega^2\left(c_x - p_x\right) \\ \ddot{c}_y = \omega^2\left(c_y - p_y\right) \\ \ddot{c}_z = \omega^2\left(c_z - p_z\right) - g \end{cases}. \tag{3.13}$$

The resultant CoM dynamics will be the well-known LIP with an additional assumption of a constant vertical CoM acceleration, i.e., $\ddot{c}_z = 0$, which leads (3.13) to

$$0 = \ddot{c}_z = \omega^2\left(c_z - p_z\right) - g \Rightarrow \omega = \sqrt{\frac{g}{c_z - p_z}}. \tag{3.14}$$

In addition, $p_z = 0$ is usually considered. Finally, the dynamics of the classic LIP can be written as

$$\ddot{\boldsymbol{c}} = \omega^2\left(\boldsymbol{c} - \boldsymbol{p}\right) \tag{3.15}$$

with the vertical CoM dynamics (3.14) embedded, where we redefine $\boldsymbol{c} = \left[c_x, c_y\right]^\top \in \mathbb{R}^2$ and $\boldsymbol{p} = \left[p_x, p_y\right]^\top \in \mathbb{R}^2$. From now on, we will only focus on the horizontal CoM motions. We can then rewrite (3.15) in its state-space form as follows:

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{c}} \\ \ddot{\boldsymbol{c}} \end{bmatrix}}_{\dot{\boldsymbol{x}}} = \underbrace{\begin{bmatrix} \boldsymbol{0} & \mathbb{I} \\ \omega^2\mathbb{I} & \boldsymbol{0} \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \boldsymbol{c} \\ \dot{\boldsymbol{c}} \end{bmatrix}}_{\boldsymbol{x}} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\omega^2\mathbb{I} \end{bmatrix}}_{\boldsymbol{B}} \underbrace{\boldsymbol{p}}_{\boldsymbol{u}}. \tag{3.16}$$

Let's define new state-space variables $\boldsymbol{z}$ related to $\boldsymbol{x}$ via $\boldsymbol{x} = \boldsymbol{T}\boldsymbol{z}$, where

$$\boldsymbol{T} = \begin{bmatrix} \mathbb{I} & \mathbf{0} \\ -\omega\mathbb{I} & \omega\mathbb{I} \end{bmatrix}, \quad \boldsymbol{T}^{-1} = \begin{bmatrix} \mathbb{I} & \mathbf{0} \\ \mathbb{I} & \omega^{-1}\mathbb{I} \end{bmatrix}, \tag{3.17}$$

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{\xi} \end{bmatrix}, \tag{3.18}$$

$$\boldsymbol{\xi} := \boldsymbol{c} + \omega^{-1}\dot{\boldsymbol{c}}. \tag{3.19}$$

The state-space equations in the new coordinates are

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{c}} \\ \dot{\boldsymbol{\xi}} \end{bmatrix}}_{\dot{\boldsymbol{z}}} = \underbrace{\begin{bmatrix} -\omega\mathbb{I} & \omega\mathbb{I} \\ \mathbf{0} & \omega\mathbb{I} \end{bmatrix}}_{\boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}} \underbrace{\begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{\xi} \end{bmatrix}}_{\boldsymbol{z}} + \underbrace{\begin{bmatrix} \mathbf{0} \\ -\omega\mathbb{I} \end{bmatrix}}_{\boldsymbol{T}^{-1}\boldsymbol{B}} \underbrace{\boldsymbol{p}}_{\boldsymbol{u}}. \tag{3.20}$$

We can see that (3.20) can be split into two first-order differentiation equations:

$$\dot{\boldsymbol{c}} = \omega\left(\boldsymbol{\xi} - \boldsymbol{c}\right), \tag{3.21}$$

$$\dot{\boldsymbol{\xi}} = \omega\left(\boldsymbol{\xi} - \boldsymbol{p}\right). \tag{3.22}$$

The dynamics (3.21) is stable, i.e., $\boldsymbol{c}$ always follows $\boldsymbol{\xi}$, whereas the dynamics (3.22) is unstable, i.e., $\boldsymbol{\xi}$ is pushed away by $\boldsymbol{p}$, which implies that controlling only the unstable part is enough to achieve stable locomotion. The variable $\boldsymbol{\xi}$ is known as the DCM [78].

Analytical Solutions

Assuming ω is a constant during the stance phase, the linear CoM dynamics (3.15) can be solved in closed form as an initial value problem (IVP). With the initial CoM condition $\boldsymbol{c}(0) = \boldsymbol{c}_0$ and $\dot{\boldsymbol{c}}(0) = \dot{\boldsymbol{c}}_0$, as well as the initial CoP location $\boldsymbol{p}_0$, the solution is given as

follows:

$$\boldsymbol{c}(t) = \rho(t)\boldsymbol{r}_0 + \omega^{-1}\sigma(t)\dot{\boldsymbol{c}}_0 + \boldsymbol{p}_0, \tag{3.23}$$

$$\dot{\boldsymbol{c}}(t) = \omega\sigma(t)\boldsymbol{r}_0 + \rho(t)\dot{\boldsymbol{c}}_0, \tag{3.24}$$

for $0 \leq t \leq T_s$, where

$$\rho(t) := \cosh(\omega t), \tag{3.25}$$

$$\sigma(t) := \sinh(\omega t), \tag{3.26}$$

and the CoM offset $\boldsymbol{r} = \begin{bmatrix} r_x, r_y \end{bmatrix}^\top \in \mathbb{R}^2$ is defined as

$$\boldsymbol{r} := \boldsymbol{c} - \boldsymbol{p}. \tag{3.27}$$

The CoM states at the end of the stance phase, i.e., $\boldsymbol{c}_{T_s} := \boldsymbol{c}(T_s)$ and $\dot{\boldsymbol{c}}_{T_s} := \dot{\boldsymbol{c}}(T_s)$, are thus given by

$$\boldsymbol{c}_{T_s} = \rho_{T_s}\boldsymbol{r}_0 + \omega^{-1}\sigma_{T_s}\dot{\boldsymbol{c}}_0 + \boldsymbol{p}_0, \tag{3.28}$$

$$\dot{\boldsymbol{c}}_{T_s} = \omega\sigma_{T_s}\boldsymbol{r}_0 + \rho_{T_s}\dot{\boldsymbol{c}}_0, \tag{3.29}$$

where $\rho_{T_s} := \rho(T_s)$ and $\sigma_{T_s} := \sigma(T_s)$. Note that the subscript indicates the gait time.

The DCM dynamics (3.22) can also be analytically solved as an IVP with the initial DCM state $\boldsymbol{\xi}(0) = \boldsymbol{\xi}_0$:

$$\boldsymbol{\xi}(t) = \eta(t)\boldsymbol{b}_0 + \boldsymbol{p}_0, \tag{3.30}$$

for $0 \leq t \leq T_s$, where

$$\eta(t) := e^{\omega t}, \tag{3.31}$$

and the DCM offset $\boldsymbol{b} = \left[b_x, b_y\right]^\top$ is defined as

$$\boldsymbol{b} := \boldsymbol{\xi} - \boldsymbol{p}. \tag{3.32}$$

The DCM state at the end of the stance phase, i.e., $\boldsymbol{\xi}_{T_s} := \boldsymbol{\xi}(T_s)$, is thus given by

$$\boldsymbol{\xi}_{T_s} = \eta_{T_s} \boldsymbol{b}_0 + \boldsymbol{p}_0, \tag{3.33}$$

where $\eta_{T_s} := \eta(T_s)$.

3.2.2 Ballistic Model

Dynamic Equations

When the robot is in flight, no external forces can affect its horizontal CoM states:

$$\ddot{\boldsymbol{c}} = \boldsymbol{0}. \tag{3.34}$$

The corresponding DCM dynamics based on the DCM definition (3.19) is given by

$$\dot{\boldsymbol{\xi}} = \dot{\boldsymbol{c}}. \tag{3.35}$$

Analytical Solutions

Based on (3.34), the equations of motion for the horizontal CoM states are given by

$$\boldsymbol{c}(t) = \boldsymbol{c}_{T_s} + (t - T_s)\, \dot{\boldsymbol{c}}_{T_s}, \tag{3.36}$$

$$\dot{\boldsymbol{c}}(t) = \dot{\boldsymbol{c}}_{T_s}, \tag{3.37}$$

for $T_s \leq t \leq T$. The CoM states at the end of the flight phase, i.e., $\boldsymbol{c}_T := \boldsymbol{c}(T)$ and $\dot{\boldsymbol{c}}_T := \dot{\boldsymbol{c}}(T)$, are thus given by

$$\boldsymbol{c}_T = \boldsymbol{c}_{T_s} + T_f \dot{\boldsymbol{c}}_{T_s}, \tag{3.38}$$

$$\dot{\boldsymbol{c}}_T = \dot{\boldsymbol{c}}_{T_s}. \tag{3.39}$$

Using (3.35) along with (3.37), the equations of motion for the horizontal DCM state is given by

$$\boldsymbol{\xi}(t) = \boldsymbol{\xi}_{T_s} + (t - T_s)\, \dot{\boldsymbol{c}}_{T_s}, \tag{3.40}$$

for $T_s \leq t \leq T$. The DCM state at the end of the flight phase, i.e., $\boldsymbol{\xi}_T := \boldsymbol{\xi}(T)$, can thus be determined to be

$$\boldsymbol{\xi}_T = \boldsymbol{\xi}_{T_s} + T_f \dot{\boldsymbol{c}}_{T_s}. \tag{3.41}$$

Note that the DCM trajectory (3.40) is not well defined during the flight phase due to the absence of the CoP and in fact only the final DCM state $\boldsymbol{\xi}_T$ is used, which initializes the next gait cycle.

3.3 Gait Analysis

The nominal symmetric and periodic gait is designed and analyzed in this section, as illustrated in Figure 3.2. Let's define the step difference $\boldsymbol{L} = \begin{bmatrix} L_x, L_y \end{bmatrix}^\top$ for the two gait cycles as follows:

$$\boldsymbol{L}_T := \boldsymbol{p}_T - \boldsymbol{p}_0, \tag{3.42}$$

$$\boldsymbol{L}_{2T} := \boldsymbol{p}_{2T} - \boldsymbol{p}_T, \tag{3.43}$$

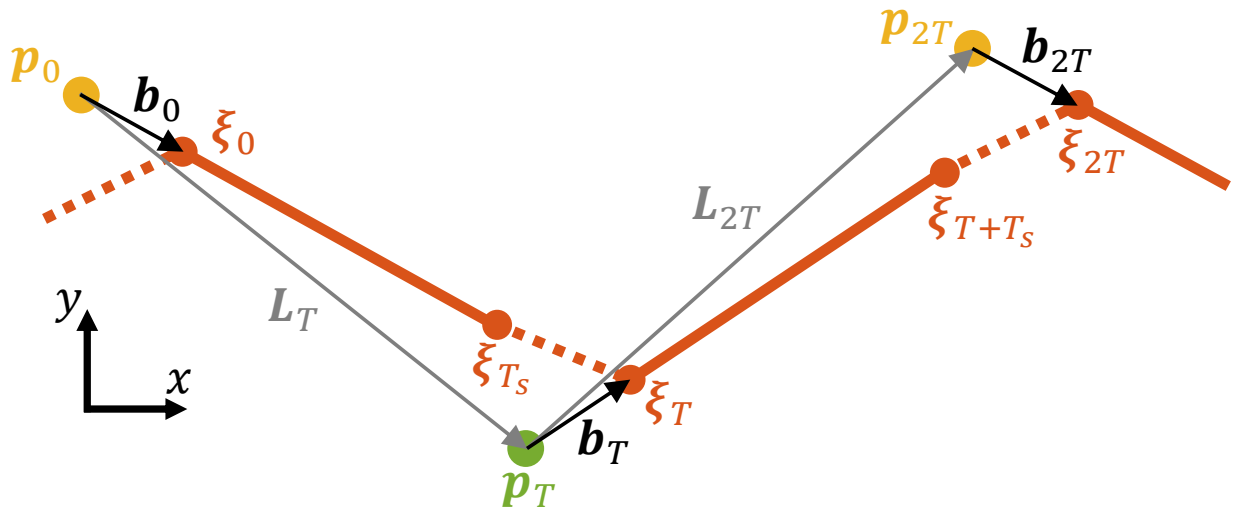


Figure 3.2: Illustration of the CoP, DCM, and initial DCM offset for two gait cycles. Each gait cycle includes a stance phase followed by a flight phase. Note that the robot is facing and locomoting to the positive x direction.

where $\boldsymbol{p}_T$ and $\boldsymbol{p}_{2T}$ are the next footstep location and the one after next, respectively. Note that we consider the CoP corresponds to the footstep location here and the robot is alternating between left and right stances.

3.3.1 Gait Prediction

At any moment, given the elapsed time τ since the start of the current phase, the current CoM states $\boldsymbol{c}_\tau$ and $\dot{\boldsymbol{c}}_\tau$, the current CoP location $\boldsymbol{p}_\tau$ if available, and the current DCM state $\boldsymbol{\xi}_\tau$, the DCM state at the end of the current cycle (i.e., upcoming touchdown moment) $\boldsymbol{\xi}_1$ can be determined per (3.40) using (3.23), (3.24), (3.30), (3.36) and (3.37) as follows:

$$\boldsymbol{\xi}_1 = \boldsymbol{\xi}_\tau + \left(T_f - \tau\right) \dot{\boldsymbol{c}}_\tau, \tag{3.44}$$

for $0 \leq \tau \leq T_f$ during the flight phase, or

$$\boldsymbol{\xi}_1 = \eta_{T_s} e^{-\omega\tau} \boldsymbol{b}_\tau + \boldsymbol{p}_\tau + T_f \dot{\boldsymbol{c}}_1, \tag{3.45}$$

for $0 \leq \tau \leq T_s$ during the stance phase, where

$$\dot{\boldsymbol{c}}_1 = \omega \sigma_{T_\tau} \boldsymbol{r}_\tau + \rho_{T_\tau} \dot{\boldsymbol{c}}_\tau, \tag{3.46}$$

$$\boldsymbol{c}_1 = \rho_{T_\tau} \boldsymbol{r}_\tau + \omega^{-1} \sigma_{T_\tau} \dot{\boldsymbol{c}}_\tau + \boldsymbol{p}_\tau + T_f \dot{\boldsymbol{c}}_1, \tag{3.47}$$

$\rho_{T_\tau} := \rho(T_s - \tau)$, and $\sigma_{T_\tau} := \sigma(T_s - \tau)$. Similarly, the next $N-1$ gait cycles can be predicted as follows:

$$\dot{\boldsymbol{c}}_{k+1} = \omega \sigma_{T_s} \boldsymbol{r}_k + \rho_{T_s} \dot{\boldsymbol{c}}_k, \tag{3.48}$$

$$\boldsymbol{c}_{k+1} = \rho_{T_s} \boldsymbol{r}_k + \omega^{-1} \sigma_{T_s} \dot{\boldsymbol{c}}_k + \boldsymbol{p}_k + T_f \dot{\boldsymbol{c}}_{k+1}, \tag{3.49}$$

$$\boldsymbol{\xi}_{k+1} = \eta_{T_s} \boldsymbol{b}_k + \boldsymbol{p}_k + T_f \dot{\boldsymbol{c}}_{k+1}, \tag{3.50}$$

for $k = 1, \ldots, N-1$.

3.3.2 Nominal Gait Pattern

The nominal gait can be characterized by the nominal phase durations T_s^{nom} and T_f^{nom}, as well as the nominal lengths $\boldsymbol{L}^{nom}$ and $\boldsymbol{b}^{nom}$. For the sake of readability, the superscript *nom* is omitted in this subsection from now on. The length parameters depend on the time parameters and thus need to be determined.

Longitudinal Direction

Assuming at steady state, a nominal average longitudinal velocity v_x is realized by a nominal step difference:

$$L_x = T \cdot v_x = p_{x,T} - p_{x,0}. \tag{3.51}$$

Consider for each gait cycle that the initial DCM offset should be equal to the nominal value b_x to sustain the periodicity:

$$b_x = b_{x,0} = b_{x,T}. \tag{3.52}$$

Similarly, the initial CoM velocity should also be equal to the corresponding nominal value $\dot{c}_x$:

$$\dot{c}_x = \dot{c}_{x,0} = \dot{c}_{x,T}. \tag{3.53}$$

Substituting (3.33) into (3.41) with (3.32) and (3.39) yields

$$\boldsymbol{p}_T - \boldsymbol{p}_0 = \eta_{T_s}\boldsymbol{b}_0 - \boldsymbol{b}_T + T_f\dot{\boldsymbol{c}}_T. \tag{3.54}$$

Substituting (3.51), (3.52), and (3.53) into (3.54) gives

$$L_x = \eta_{T_s}b_x - b_x + T_f\dot{c}_x \tag{3.55}$$

Substituting (3.29) into (3.39) with (3.19) and (3.32) yields

$$\dot{\boldsymbol{c}}_T = \omega\sigma_{T_s}\boldsymbol{b}_0 + \eta_{T_s}^{-1}\dot{\boldsymbol{c}}_0. \tag{3.56}$$

Substituting (3.52) and (3.53) into (3.56) gives

$$\dot{c}_x = \omega\sigma_{T_s}b_x + \eta_{T_s}^{-1}\dot{c}_x. \tag{3.57}$$

The system of equations (3.55) and (3.57) is now in terms of b_x and $\dot{c}_x$, solving them yields

$$b_x = \frac{L_x}{\eta_{T_s} - 1 + \omega\alpha_{T_s}T_f}, \tag{3.58}$$

$$\dot{c}_x = \omega \alpha_{T_s} b_x, \tag{3.59}$$

where the parameter

$$\alpha_{T_s} := \frac{\sigma_{T_s}}{1 - \eta_{T_s}^{-1}} \geq 1, \tag{3.60}$$

given $\omega T_s \geq 0$. We can further investigate the characteristics of b_x in terms of its components, e.g., T_s, T_f, v_x. Without the flight phase, i.e., $T_f = 0$, as shown in Figure 3.3(a), we can see that b_x increases as v_x increases but as T_s decreases. This makes sense because for faster v_x and shorter T_s, larger b_x is expected to produce greater CoM acceleration at the beginning. With the flight phase involved, as shown in Figure 3.3(b), for fixed v_x, T_f does not play an important role in b_x. This also makes sense because since the CoM and DCM share the same velocity throughout the flight phase, the nominal gait with and without the flight phase will result in almost identical b_x with only slight differences.

Lateral Direction

Assuming at steady state, a nominal average lateral velocity v_y is realized by a nominal step difference:

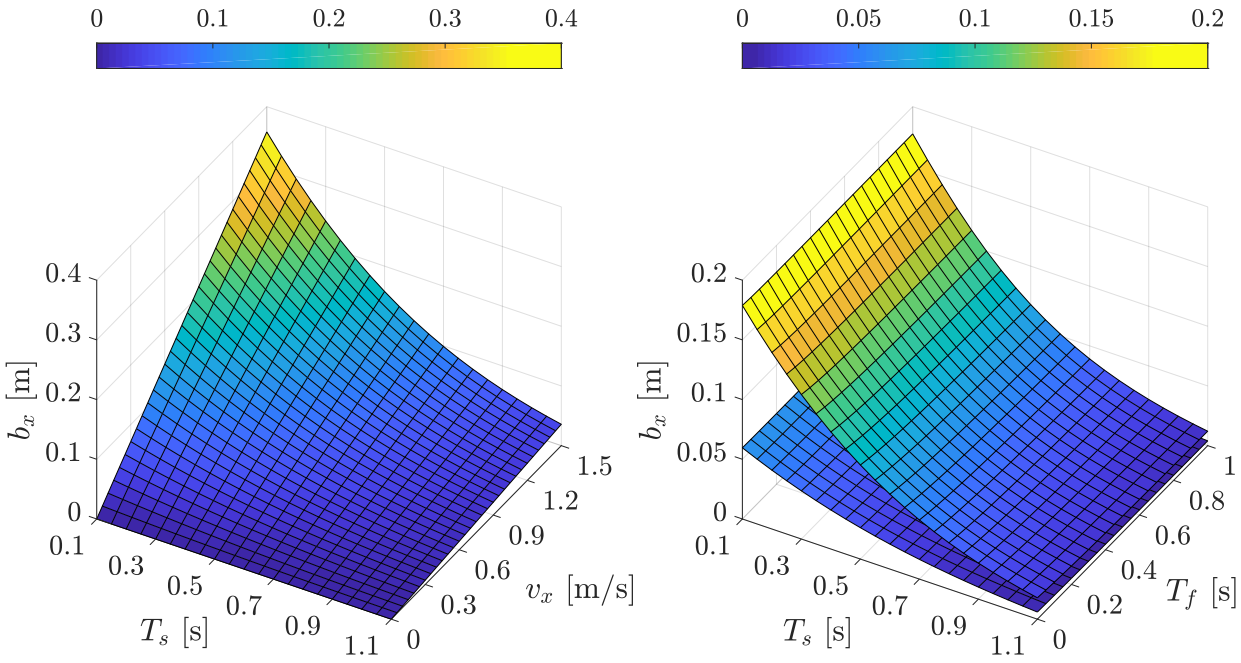
$$L_y^l = W_y - d_y = p_{y,T} - p_{y,0}, \tag{3.61}$$

$$L_y^r = W_y + d_y = p_{y,2T} - p_{y,T}, \tag{3.62}$$

for the left and right stances, respectively, where $W_y = T \cdot v_y$ is the nominal pelvis movement and d_y is the lateral footstep clearance resulting from the pelvis width. Note that two complete gait cycles need to be considered to sustain the periodicity in the lateral direction:

$$b_y^l = b_{y,0} = b_{y,2T}, \quad b_y^r = b_{y,T}, \tag{3.63}$$

$$\dot{c}_y^l = \dot{c}_{y,0} = \dot{c}_{y,2T}, \quad \dot{c}_y^r = \dot{c}_{y,T}. \tag{3.64}$$



(a) (b)

Figure 3.3: Characteristics of nominal initial DCM offset b_x, (a) without the flight phase and (b) with the flight phase, where the top and bottom layers correspond to $v_x = 0.75$ m/s and 0.25 m/s, respectively. Note that $\omega \approx 3.50$ Hz with $c_z = 0.8$ m is assumed.

Substituting (3.61), (3.62), (3.63), and (3.64) into (3.54) gives

$$W_y - d_y = \eta_{T_s} b_y^l - b_y^r + T_f \dot{c}_y^r, \tag{3.65}$$

$$W_y + d_y = \eta_{T_s} b_y^r - b_y^l + T_f \dot{c}_y^l. \tag{3.66}$$

Substituting (3.63) and (3.64) into (3.56) gives

$$\dot{c}_y^l = \omega \sigma_{T_s} b_y^r + \eta_{T_s}^{-1} \dot{c}_y^r, \tag{3.67}$$

$$\dot{c}_y^r = \omega \sigma_{T_s} b_y^l + \eta_{T_s}^{-1} \dot{c}_y^l. \tag{3.68}$$

The system of equations (3.65), (3.66), (3.67), and (3.68) is now in terms of b_y^l, b_y^r, $\dot{c}_y^l$ and $\dot{c}_y^r$, solving them yields

$$b_y^l = \frac{W_y}{\eta_{T_s} - 1 + \omega\alpha_{T_s} T_f} - \frac{d_y}{\eta_{T_s} + 1 + \omega\beta_{T_s} T_f}, \tag{3.69}$$

$$b_y^r = \frac{W_y}{\eta_{T_s} - 1 + \omega\alpha_{T_s} T_f} + \frac{d_y}{\eta_{T_s} + 1 + \omega\beta_{T_s} T_f}, \tag{3.70}$$

$$\dot{c}_y^l = \frac{\omega\beta_{T_s}\left(\eta_{T_s} b_y^r + b_y^l\right)}{\eta_{T_s} - 1}, \tag{3.71}$$

$$\dot{c}_y^r = \frac{\omega\beta_{T_s}\left(\eta_{T_s} b_y^l + b_y^r\right)}{\eta_{T_s} - 1}. \tag{3.72}$$

where the parameter

$$\beta_{T_s} := \frac{\sigma_{T_s}}{1 + \eta_{T_s}^{-1}} \geq 0, \tag{3.73}$$

given $\omega T_s \geq 0$.

A series of nominal gait patterns for walking (i.e., $T_f = 0$) with different step durations are shown in Figure 3.4. We can see that the robot CoM tends to swing more in the lateral direction as T_s increases. Note that it is undesirable to have a large T_s as the real robot does not behave as the LIP exactly especially when T_s is large and thus the nominal gait pattern becomes meaningless.

3.3.3 Viability Condition

The viability condition is further investigated to specify the boundary of the DCM-based gait pattern. In terms of the initial DCM offset, if it is within the limits, there exists at least one combination of step location and timing to keep the DCM from diverging. If it is out of the limits, no solution exists, i.e., the DCM will just diverge and the robot will fall down eventually. The DCM offset limits can be derived based on the concept of capturability [77].

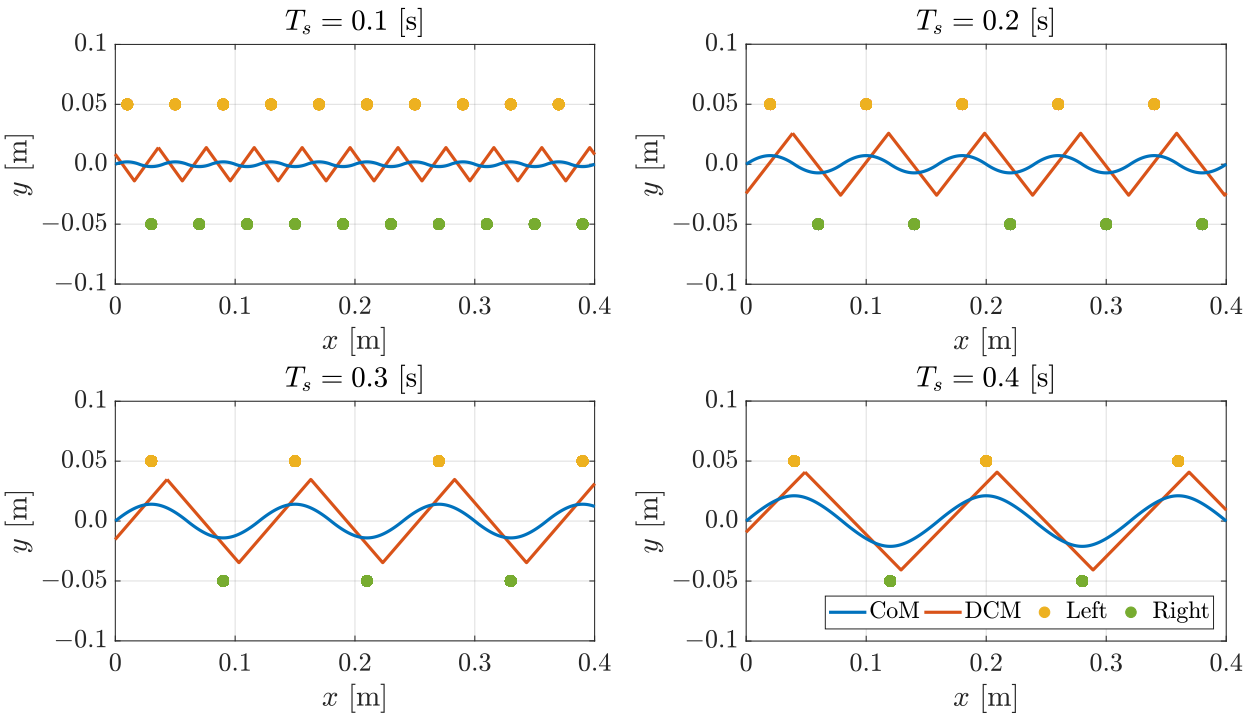


Figure 3.4: Nominal gait pattern with different step durations. Note that the robot is walking to the positive x direction with a fixed average CoM speed of 0.1 m/s, a fixed CoM height of 0.3 m, and a fixed step width of 0.1 m.

Longitudinal Direction

For the forward (i.e., positive x) direction, suppose the initial DCM offset $b_{x,0}$ is M-step capturable:

$$b_{x,0} \leq \gamma^{+}_{x,M}, \tag{3.74}$$

where $\gamma^{+}_{x,M}$ is the corresponding capturability bound. The initial DCM offset of the next gait cycle $b_{x,T}$ is thus $(M-1)$-step capturable:

$$b_{x,T} \leq \gamma^{+}_{x,M-1}. \tag{3.75}$$

Substituting (3.54) into (3.75) and rearranging gives

$$b_{x,0} \leq \eta_{T_s}^{-1} \left(\gamma_{x,M-1}^{+} + L_{x,T} - T_f \dot{c}_{x,T} \right). \tag{3.76}$$

The upper bound in (3.76) is maximized when $T_s = T_{s,\min}$ and $L_{x,T} = L_{x,\max} + T_f \dot{c}_{x,T}$, where we take into account both physical limit $L_{x,\max}$ and flight velocity, i.e., the feasible footstep region for the next gait cycle travels with the robot during the flight phase, which also leads to a similar result without considering the flight phase. This maximum upper bound should coincide with $\gamma_{x,M}^{+}$ in (3.74), which yields the recursive expression:

$$\gamma_{x,M}^{+} = \eta_{T_{s,\min}}^{-1} \left(\gamma_{x,M-1}^{+} + L_{x,\max} \right). \tag{3.77}$$

The solution with $\gamma_{x,0}^{+} = 0$ is given by

$$\gamma_{x,M}^{+} = L_{x,\max} \frac{1 - e^{-\omega M T_{s,\min}}}{\eta_{T_{s,\min}} - 1}, \tag{3.78}$$

and its limit is known as the ∞-step capturability bound:

$$\gamma_{x,\infty}^{+} = \lim_{M \to \infty} \gamma_{x,M}^{+} = \frac{L_{x,\max}}{\eta_{T_{s,\min}} - 1}, \tag{3.79}$$

which is also considered the DCM offset limit for the forward direction and we thus denote $b_{x,\max} = \gamma_{x,\infty}^{+}$. Similarly, for the backward (i.e., negative x) direction, we have

$$b_{x,\min} = \lim_{M \to \infty} \underbrace{L_{x,\min} \frac{1 - e^{-\omega M T_{s,\min}}}{\eta_{T_{s,\min}} - 1}}_{= \gamma_{x,M}^{-}} = \frac{L_{x,\min}}{\eta_{T_{s,\min}} - 1}. \tag{3.80}$$

Lateral Direction

For the inward (i.e., positive y) direction of the right leg, suppose the initial DCM offset $b_{y,0}$ is $2M$-step capturable:

$$b_{y,0} \leq \gamma^{r+}_{y,M}, \tag{3.81}$$

where $\gamma^{r+}_{y,M}$ is the corresponding capturability bound. The initial DCM offset after two gait cycles $b_{y,2T}$ is thus $2(M-1)$-step capturable:

$$b_{y,2T} \leq \gamma^{r+}_{y,M-1}. \tag{3.82}$$

Again, two gait cycles are considered for the lateral case. Substituting (3.54) into (3.82) twice and rearranging yields

$$b_{y,0} \leq \eta^{-1}_{T_s}\left(L_{y,T} - T_f\dot{c}_{y,T}\right) + \eta^{-2}_{T_s}\left(\gamma^{r+}_{y,M-1} + L_{y,2T} - T_f\dot{c}_{y,2T}\right). \tag{3.83}$$

The upper bound is maximized when $T_s = T_{s,\min}$, $L_{y,T} = L^r_{y,\max} + T_f\dot{c}_{y,T}$, and $L_{y,2T} = L^l_{y,\max} + T_f\dot{c}_{y,2T}$. This maximum upper bound should coincide with $\gamma^{r+}_{y,M}$ in (3.81), which yields the recursive expression:

$$\gamma^{r+}_{y,M} = \eta^{-1}_{T_{s,\min}} L^r_{y,\max} + \eta^{-2}_{T_{s,\min}}\left(\gamma^{r+}_{y,M-1} + L^l_{y,\max}\right). \tag{3.84}$$

The solution of this recursive expression with $\gamma^{r+}_{y,0} = 0$ and its limit are respectively

$$\gamma^{r+}_{y,M} = \left(\eta_{T_{s,\min}} L^r_{y,\max} + L^l_{y,\max}\right)\frac{1 - e^{-2\omega M T_{s,\min}}}{\eta^2_{T_{s,\min}} - 1}, \tag{3.85}$$

$$b^r_{y,\max} = \lim_{M\to\infty}\gamma^{r+}_{y,M} = \frac{\eta_{T_{s,\min}} L^r_{y,\max} + L^l_{y,\max}}{\eta^2_{T_{s,\min}} - 1}. \tag{3.86}$$

Similarly, for the outward (i.e., negative y) direction and the left leg, we have

$$\gamma_{y,M}^{r-} = \left(\eta_{T_{s,\min}} L_{y,\min}^{r} + L_{y,\min}^{l}\right) \frac{1 - e^{-2\omega M T_{s,\min}}}{\eta_{T_{s,\min}}^{2} - 1}, \tag{3.87}$$

$$b_{y,\min}^{r} = \lim_{M\to\infty} \gamma_{y,M}^{r-} = \frac{\eta_{T_{s,\min}} L_{y,\min}^{r} + L_{y,\min}^{l}}{\eta_{T_{s,\min}}^{2} - 1}, \tag{3.88}$$

$$\gamma_{y,M}^{l+} = \left(\eta_{T_{s,\min}} L_{y,\max}^{l} + L_{y,\max}^{r}\right) \frac{1 - e^{-2\omega M T_{s,\min}}}{\eta_{T_{s,\min}}^{2} - 1}, \tag{3.89}$$

$$b_{y,\max}^{l} = \lim_{M\to\infty} \gamma_{y,M}^{l+} = \frac{\eta_{T_{s,\min}} L_{y,\max}^{l} + L_{y,\max}^{r}}{\eta_{T_{s,\min}}^{2} - 1}, \tag{3.90}$$

$$\gamma_{y,M}^{l-} = \left(\eta_{T_{s,\min}} L_{y,\min}^{l} + L_{y,\min}^{r}\right) \frac{1 - e^{-2\omega M T_{s,\min}}}{\eta_{T_{s,\min}}^{2} - 1}, \tag{3.91}$$

$$b_{y,\min}^{l} = \lim_{M\to\infty} \gamma_{y,M}^{l-} = \frac{\eta_{T_{s,\min}} L_{y,\min}^{r} + L_{y,\min}^{l}}{\eta_{T_{s,\min}}^{2} - 1}. \tag{3.92}$$

Figure 3.5 helps better understand the gait viability condition. Given the left foot $\boldsymbol{p}_0$ of the current gait cycle, the feasible footstep region for the right foot of the next gait cycle can be first determined. The DCM viability kernel [179] can be further obtained by adding the DCM offset limits. The system is viable, i.e., the robot is able to avoid falling, as long as the initial DCM of the next gait cycle $\boldsymbol{\xi}_T$ is within the viability kernel, or equivalently, the initial DCM offset is bounded by the DCM offset limits. The viability condition can be useful for robot to make its own judgement, e.g., recovery stepping, fall detection, prevention, and preparation.

3.4 Locomotion Control

In previous sections, the DCM evolution is predicted, the nominal gait pattern is derived, and the viability condition is also investigated. We can now construct our locomotion control framework which is capable of real-time planning for the next few steps including both step location and timing, such that robust locomotion can be achieved in the presence of external

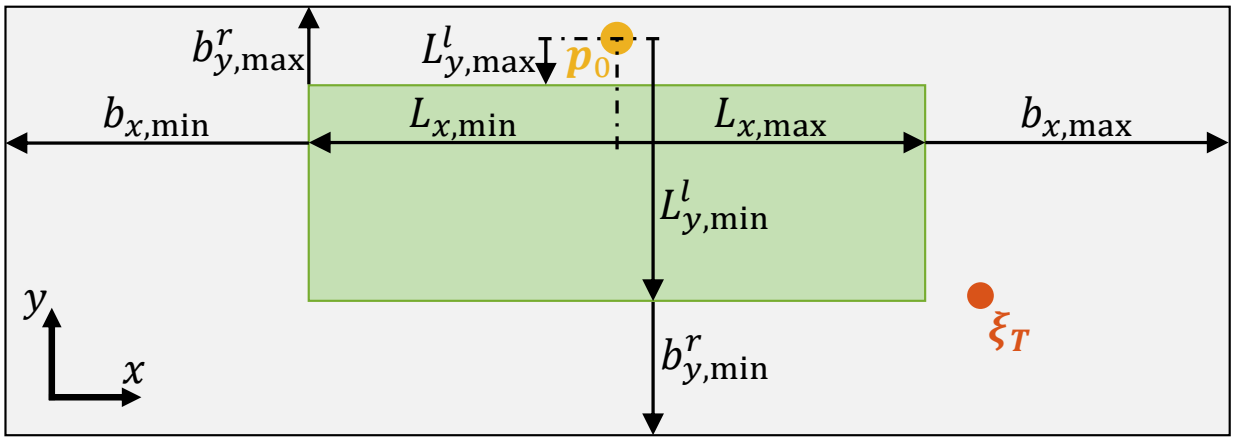


Figure 3.5: Illustration of gait viability condition. The green and gray boxes are respectively the next footstep region and DCM viability kernel. The system is viable as long as $\boldsymbol{\xi}_T$ is within the viability kernel.

disturbances. This TO problem is formulated as the following mathematical optimization (a QP problem essentially):

$$\min_{\substack{\boldsymbol{p}_k,\boldsymbol{b}_k,\\ \eta_{T_s},T_f}} \sum_{k=1}^{N} \|\boldsymbol{L}_k - \boldsymbol{R}_k \boldsymbol{L}_k^{nom}\|^2_{\boldsymbol{W}_L} + \|\boldsymbol{b}_k - \boldsymbol{R}_k \boldsymbol{b}_k^{nom}\|^2_{\boldsymbol{W}_b}$$

$$+ w_s \left|\eta_{T_s} - \eta_{T_s^{nom}}\right|^2 + w_f \left|T_f - T_f^{nom}\right|^2 \tag{3.93}$$

$$\text{s.t.} \quad \boldsymbol{\xi}_1 = \begin{cases} \eta_{T_s} e^{-\omega\tau} \boldsymbol{b}_\tau + \boldsymbol{p}_\tau + T_f \dot{\boldsymbol{c}}_1 & \text{if in stance,} \\ \boldsymbol{\xi}_\tau + \left(T_f - \tau\right) \dot{\boldsymbol{c}}_\tau & \text{if in flight,} \end{cases} \tag{3.94}$$

$$\boldsymbol{\xi}_{k+1} = \eta_{T_s^{nom}} \boldsymbol{b}_k + \boldsymbol{p}_k + T_f^{nom} \boldsymbol{R}_k \dot{\boldsymbol{c}}_{k+1}^{nom}, \quad k = 1, \ldots, N-1, \tag{3.95}$$

$$\boldsymbol{L}_{\min} \leq \boldsymbol{R}_k^\top \boldsymbol{L}_k \leq \boldsymbol{L}_{\max}, \quad k = 1, \ldots, N, \tag{3.96}$$

$$\eta_{T_{s,\min}} \leq \eta_{T_s} \leq \eta_{T_{s,\max}}, \tag{3.97}$$

$$T_{f,\min} \leq T_f \leq T_{f,\max}, \tag{3.98}$$

Table 3.1: QP specifications.

Decision Variables	$\underbrace{2N}_{\#(\boldsymbol{p}_k)} + \underbrace{2N}_{\#(\boldsymbol{b}_k)} + \underbrace{1}_{\#(\eta_{T_s})} + \underbrace{1}_{\#(T_f)} = 4N + 2$
Equality Constraints	$\underbrace{2}_{\#(3.94)} + \underbrace{2(N-1)}_{\#(3.95)} = 2N$
Inequality Constraints	$\underbrace{4N}_{\#(3.96)} + \underbrace{2}_{\#(3.97)} + \underbrace{2}_{\#(3.98)} = 4N + 4$

where

$$\boldsymbol{L}_k = \begin{cases} \boldsymbol{p}_k - \boldsymbol{p}_\tau & \text{if } k = 1, \\ \boldsymbol{p}_k - \boldsymbol{p}_{k-1} & \text{if } k = 2, \dots, N, \end{cases} \tag{3.99}$$

N is the number of preview steps, and $\boldsymbol{R} \in \mathsf{SO}(2)$ is the planar rotation matrix describing the desired robot orientation, which transfers vectors from the robot frame to the world frame. Note that all the nominal gait length parameters are derived in the robot frame. Table 3.1 summarizes the QP specifications, which is really a small-scale optimization even with multiple steps being considered ahead of time. Let us go over the details of its formulation.

3.4.1 Decision Variables

The decision variables first include the footstep locations $\boldsymbol{p}_k$ and the initial DCM offsets $\boldsymbol{b}_k$ for the next N gait cycles, where $k = 1, \dots, N$. The DCM states $\boldsymbol{\xi}_k$ can be easily expressed by them using (3.32) and thus are excluded. Since the flight phase is considered as well, the CoM states should also have been involved as decision variables so as to properly predict the DCM states per Section 3.3.1. However, for the sake of linearity and simplicity, they are excluded and only their current measurements or nominal values are used, which still works well in practice.

In addition, the phase durations of the current gait cycle are also included as decision

variables because step timing is just as critical as step location to robust locomotion, if not more so. Note that a change of variable is introduced for the stance duration in order to maintain the linearity of the constraints, i.e., η_{T_s} is used instead of T_s. The phase durations of the rest gait cycles in the long horizon are not of high priority. They are not optimized and are set to their nominal values for linearity and simplicity.

3.4.2 Cost Function

The optimizer adapts the nominal gait pattern which is characterized by the gait parameters T_s, T_f, $\boldsymbol{L}$, and $\boldsymbol{b}$. Accordingly, the cost function (3.93) penalizes the overall deviation from their corresponding nominal values in the least-squares sense, where the weighted vector norm square is defined as $\|\boldsymbol{e}\|^2_{\boldsymbol{W}} := \boldsymbol{e}^\top \boldsymbol{W} \boldsymbol{e}$ for $\boldsymbol{e} \in \mathbb{R}^n$ and $\boldsymbol{W} \in \mathbb{S}^n_+$. In practice, having a much larger $\boldsymbol{W}_b$ than $\boldsymbol{W}_L$ enhances gait viability. Furthermore, steering capability can be easily achieved by introducing the desired robot orientation $\boldsymbol{R}_k$ for each gait cycle. Finally, the first cost term only considers the relative step differences $\boldsymbol{L}_k$, which however can be replaced or extra guided with global references if necessary.

3.4.3 Constraints

Several constraints need to be imposed in order to satisfy the various requirements, which are all linear in terms of the decision variables.

DCM Prediction

For locomotion with several steps being planned ahead of time, the DCM state needs to be predicted properly. Specifically, (3.94) predicts $\boldsymbol{\xi}_1$ based on (3.45) during the stance phase and (3.44) during the flight phase. In the long run, (3.95) predicts the DCM evolution for the next $N-1$ gait cycles using (3.50).

Kinematic Reachability

To ensure each stepping is physically feasible, (3.96) bounds the relative step difference between each adjacent pair of the footstep locations, with flight phase considered. Note that similar to $\boldsymbol{L}^{nom}$, the limits in the lateral direction depend on the stance foot due to the pelvis width, and meanwhile, leg crossing is also prevented.

Phase Duration Limit

The stance and flight durations for the current gait cycle are constrained by (3.97) and (3.98), respectively, where the minimum prevents large swing foot acceleration while the maximum avoids slow stepping.

Viability Condition

In general, the viability condition could also be imposed as a hard constraint to guarantee the system viability, which bounds the initial DCM offset per Section 3.3.3. However, this viability constraint along with the kinematic reachability constraint (3.96) can overconstrain the problem in extreme scenarios, which may result in optimizer failure, i.e., infeasible. In practice, this constraint is omitted and instead, the weight $\boldsymbol{W}_b$ is set large in the cost (3.93) to drive the planned DCM offset as close to its nominal value as possible, even if viability cannot be sustained. In this way, the problem is always feasible.

3.4.4 QP Formulation

The locomotion TO can be manipulated and rearranged into the standard QP form as (A.3), where

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{b}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{p}_{N-1} \\ \boldsymbol{b}_{N-1} \\ \boldsymbol{p}_N \\ \boldsymbol{b}_N \\ \hline \eta_{T_s} \\ T_f \end{bmatrix}, \tag{3.100}$$

$$\boldsymbol{P} = 2 \left[\begin{array}{cccccccccc|cc} 2\boldsymbol{W}_L & \boldsymbol{0} & -\boldsymbol{W}_L & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}_b & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{W}_L & \boldsymbol{0} & 2\boldsymbol{W}_L & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_b & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & 2\boldsymbol{W}_L & \boldsymbol{0} & -\boldsymbol{W}_L & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{W}_b & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & -\boldsymbol{W}_L & \boldsymbol{0} & \boldsymbol{W}_L & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_b & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & w_s & 0 \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & 0 & w_f \end{array}\right], \tag{3.101}$$

$$\boldsymbol{q} = -2 \left[\begin{array}{c} \boldsymbol{W}_L \left(\boldsymbol{p}_\tau + \boldsymbol{R}_1 \boldsymbol{L}_1^{nom} - \boldsymbol{R}_2 \boldsymbol{L}_2^{nom}\right) \\ \boldsymbol{W}_b \boldsymbol{R}_1 \boldsymbol{b}_1^{nom} \\ \boldsymbol{W}_L \left(\boldsymbol{R}_2 \boldsymbol{L}_2^{nom} - \boldsymbol{R}_3 \boldsymbol{L}_3^{nom}\right) \\ \boldsymbol{W}_b \boldsymbol{R}_2 \boldsymbol{b}_2^{nom} \\ \vdots \\ \boldsymbol{W}_L \left(\boldsymbol{R}_{N-1} \boldsymbol{L}_{N-1}^{nom} - \boldsymbol{R}_N \boldsymbol{L}_N^{nom}\right) \\ \boldsymbol{W}_b \boldsymbol{R}_{N-1} \boldsymbol{b}_{N-1}^{nom} \\ \boldsymbol{W}_L \boldsymbol{R}_N \boldsymbol{L}_N^{nom} \\ \boldsymbol{W}_b \boldsymbol{R}_N \boldsymbol{b}_N^{nom} \\ \hline w_s \eta_{T_s^{nom}} \\ w_f T_f^{nom} \end{array}\right], \tag{3.102}$$

$$
\boldsymbol{G} = \left[\begin{array}{cccccccccc|cc}
\boldsymbol{R}_1^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
-\boldsymbol{R}_2^\top & \mathbf{0} & \boldsymbol{R}_2^\top & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & -\boldsymbol{R}_3^\top & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{R}_{N-1}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & -\boldsymbol{R}_N^\top & \mathbf{0} & \boldsymbol{R}_N^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\hline
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & 0 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & 1 \\
\hline\hline
-\boldsymbol{R}_1^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\boldsymbol{R}_2^\top & \mathbf{0} & -\boldsymbol{R}_2^\top & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \boldsymbol{R}_3^\top & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & -\boldsymbol{R}_{N-1}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{R}_N^\top & \mathbf{0} & -\boldsymbol{R}_N^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\hline
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -1 & 0 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & -1
\end{array}\right], \tag{3.103}
$$

$$\boldsymbol{h} = \left[\begin{array}{c} \boldsymbol{L}_{\max} + \boldsymbol{R}_1^\top \boldsymbol{p}_\tau \\ \boldsymbol{L}_{\max} \\ \boldsymbol{L}_{\max} \\ \vdots \\ \boldsymbol{L}_{\max} \\ \boldsymbol{L}_{\max} \\ \hline \eta_{T_{s,\max}} \\ T_{f,\max} \\ \hline\hline -\boldsymbol{L}_{\min} - \boldsymbol{R}_1^\top \boldsymbol{p}_\tau \\ -\boldsymbol{L}_{\min} \\ -\boldsymbol{L}_{\min} \\ \vdots \\ -\boldsymbol{L}_{\min} \\ -\boldsymbol{L}_{\min} \\ \hline -\eta_{T_{s,\min}} \\ -T_{f,\min} \end{array}\right], \tag{3.104}$$

$$\boldsymbol{A} = \left[\begin{array}{cccccccc|c} -\mathbb{I} & -\mathbb{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbb{I} & \eta_{T_s^{nom}}\mathbb{I} & -\mathbb{I} & -\mathbb{I} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbb{I} & \eta_{T_s^{nom}}\mathbb{I} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & -\mathbb{I} & -\mathbb{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbb{I} & \eta_{T_s^{nom}}\mathbb{I} & -\mathbb{I} & -\mathbb{I} \end{array}\right| \left\{\begin{array}{cc} e^{-\omega\tau}\boldsymbol{b}_\tau & \dot{\boldsymbol{c}}_1 \quad \text{if in stance} \\ \mathbf{0} & \dot{\boldsymbol{c}}_\tau \quad \text{if in flight} \end{array}\right. \begin{array}{cc} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \vdots & \vdots \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{array}\right], \tag{3.105}$$

$$\boldsymbol{b} = \begin{bmatrix} \begin{cases} -\boldsymbol{p}_\tau & \text{if in stance} \\ -\boldsymbol{\xi}_\tau + \tau \dot{\boldsymbol{c}}_\tau & \text{if in flight} \end{cases} \\ -T_f^{nom} \boldsymbol{R}_1 \dot{\boldsymbol{c}}_2^{nom} \\ -T_f^{nom} \boldsymbol{R}_2 \dot{\boldsymbol{c}}_3^{nom} \\ \vdots \\ -T_f^{nom} \boldsymbol{R}_{N-2} \dot{\boldsymbol{c}}_{N-1}^{nom} \\ -T_f^{nom} \boldsymbol{R}_{N-1} \dot{\boldsymbol{c}}_N^{nom} \end{bmatrix}. \tag{3.106}$$

3.4.5 Discussion

- We are constantly solving and adapting the phase duration of the current gait cycle and therefore, first, we need to make sure the solution respects causality, i.e., the solved phase duration should be at least greater than the elapsed time τ. In addition, to prevent instantaneous stepping, i.e., the optimizer may suggest ending the current cycle instantly while the swing foot could be still high in the air, we ensure the minimum phase duration longer than $\tau + T_g$, where $T_g > 0$ is the timing gap. Finally, we keep adapting the step location and timing only if $\tau + T_g < T^*$, where T^* is the solved optimal total gait duration:

$$T^* = T_s^* + T_f^*, \tag{3.107}$$

$$T_s^* = \omega^{-1} \ln \eta_{T_s}^*, \tag{3.108}$$

  via the inverse of (3.31).

- Recall the desired T_s is involved to compute $\dot{\boldsymbol{c}}_1$ per (3.46), which is true only for the very first optimization of the current gait cycle. Afterwards, the previous T_s^* is actually used for better prediction results.

- During the flight phase, optimizing the previous stance duration of the current gait

Table 3.2: Parameters of model and QP.

Parameter	Value	Parameter	Value
m	60 [kg]	$\boldsymbol{W}_L$	$\mathbf{diag}(1, 1)$
c_z	0.8 [m]	$\boldsymbol{W}_b$	$\mathbf{diag}(1000, 1000)$
d_y	0.2 [m]	w_s	10
L_x	$-0.5 \sim 0.5$ [m]	w_f	0.1
L_y^l	$-0.4 \sim -0.05$ [m]	T_s	$0.2 \sim 0.5$ [s]
L_y^r	$0.05 \sim 0.4$ [m]	T_f	$0.0 \sim 0.2$ [s]

cycle is meaningless and thus η_{T_s} is excluded from the decision variables.

3.5 Numerical Results

To study the theoretical capabilities of the proposed approach independent of the actual robot, it was first implemented on the reduced-order models, i.e., LIP during the stance phase and BM during the flight phase, in a numerical simulation environment. In particular, the closed-form dynamic solutions (3.23), (3.24) and (3.36), (3.37) are used to determine the CoM states at any moment during the stance and flight phases, respectively. The locomotion controller optimally plans the step location and timing via the QP in Section 3.4. Table 3.2 summarizes the parameters of the model and QP, where the ranges indicate the parameter limits, e.g., $\boldsymbol{L}_{\min\,/\,\max}$ in (3.96).

3.5.1 Test Setup

We are most interested in the system performance in terms of external disturbance rejection and a push recovery test was thus conducted. In this test, a nominal CoM velocity $\dot{\boldsymbol{c}}^{nom} = [0.3, 0]^\top$ m/s, a nominal stance duration $T_s^{nom} = 0.35$ s, and a nominal flight duration $T_f^{nom} = 0.15$ s are considered. The nominal step difference and initial DCM offset can then

be determined to be $\boldsymbol{L}^{nom} = [0.15, \pm 0.2]^\top$ m and $\boldsymbol{b}^{nom} = [0.0421, \pm 0.0397]^\top$ m per Section 3.3.2. The QP in Section 3.4 is solved using the *quadprog* function in MATLAB and the optimal solution is adapted at a frequency of 500 Hz. At steady state, a constant external force $\boldsymbol{F}$ with a duration of 0.1 s was exerted on the robot CoM at $t = 1.6$ s, when its left foot was in stance.

3.5.2 System Robustness

Figure 3.6 compares the range of the external force that the robot can resist. For each push direction θ, we gradually increased the force magnitude $\|\boldsymbol{F}\|$ until the states diverged and the maximum value was recorded. With phase duration fixed, the allowable force region is always the same no matter how many preview steps N are considered in advance. With phase duration optimized, the force region expands significantly, and it gets slightly larger as N gets bigger. We also tried $N > 3$ but the results are almost identical to $N = 3$. The theoretical force boundary is plotted as well to verify the viability condition. With a constant force $\boldsymbol{F}$ applied from t_1 to t_2 since the stance phase begins, the DCM state at the end of the stance phase $\boldsymbol{\xi}_{T_s}$ becomes

$$\boldsymbol{\xi}_{T_s} = \boldsymbol{p}_0 + \left(\boldsymbol{\xi}_{t_2} - \boldsymbol{p}_0\right) e^{\omega\left(T_s - t_2\right)}, \tag{3.109}$$

$$\boldsymbol{\xi}_{t_2} = \boldsymbol{p}_0 + \left(\boldsymbol{\xi}_{t_1} - \boldsymbol{p}_0\right) e^{\omega \Delta t} + \frac{e^{\omega \Delta t} - 1}{\omega^2} \cdot \frac{\boldsymbol{F}}{m}, \tag{3.110}$$

by adding $\boldsymbol{F}/(m\omega)$ to the right of (3.22), where $\Delta t = t_2 - t_1$. The force region can be solved by putting in the viability kernel of $\boldsymbol{\xi}_{T_s}$ per Figure 3.5. Note that the viability condition of $\boldsymbol{\xi}_{T_s}$ is the same as $\boldsymbol{\xi}_T$ if there is no disturbance during the flight phase. We did not consider disturbance during the flight phase, but interested readers can derive that theoretical region with a similar technique. The simulated results well match the theoretical ones with phase duration fixed. However, the results deviate with phase duration optimized. This is because the theoretical boundary is calculated with $T_{s,\min}$, but in simulation, the timing gap

mechanism T_g can prevent the current stance duration from reaching its minimum in order to avoid instantaneous stepping, as observed in Figure 3.7. While we do see better matching result with smaller T_g, we still keep Figure 3.6 as it is since it is more reasonable in practice.

3.5.3 Transient Behavior

We further compare the detailed simulation results of the case with $\|\boldsymbol{F}\| = 370$ N and $\theta = 90°$ in Figure 3.7. With phase duration fixed, the robot fell down as expected since it is beyond the force boundary. With phase duration optimized, as N gets bigger, the robot was able to recover faster and closer to its original path. The updated phase duration was also less chaotic. In addition, the nominal gait pattern was sacrificed for the best of viability, i.e., $T_f^* = 0$ immediately after push. Again, we tried $N > 3$ but the results are almost identical to $N = 3$. Similar behavior can be observed for other cases. We thus conclude that optimization of phase duration is as critical as footstep location to robust locomotion, if not more so. Furthermore, the system performs better as N gets larger yet without significant improvement for $N > 3$, which is also supported by the full bipedal robot simulation later in the next chapter.

3.6 Conclusion

In this chapter, the DCM-based analysis and control of dynamic bipedal locomotion with flight phase included is presented. The nominal gait pattern is first derived and the viability condition is also investigated using the reduced-order models. The locomotion control is solving a small-scale QP with a horizon of multiple steps being planned in advance, which optimally determines the footstep location and phase duration in real time. The proposed locomotion strategy is verified with numerical simulations in the presence of external disturbances. The results further suggest that planning with three preview steps is a wise choice in terms of system robustness, transient behavior, and computational efficiency.

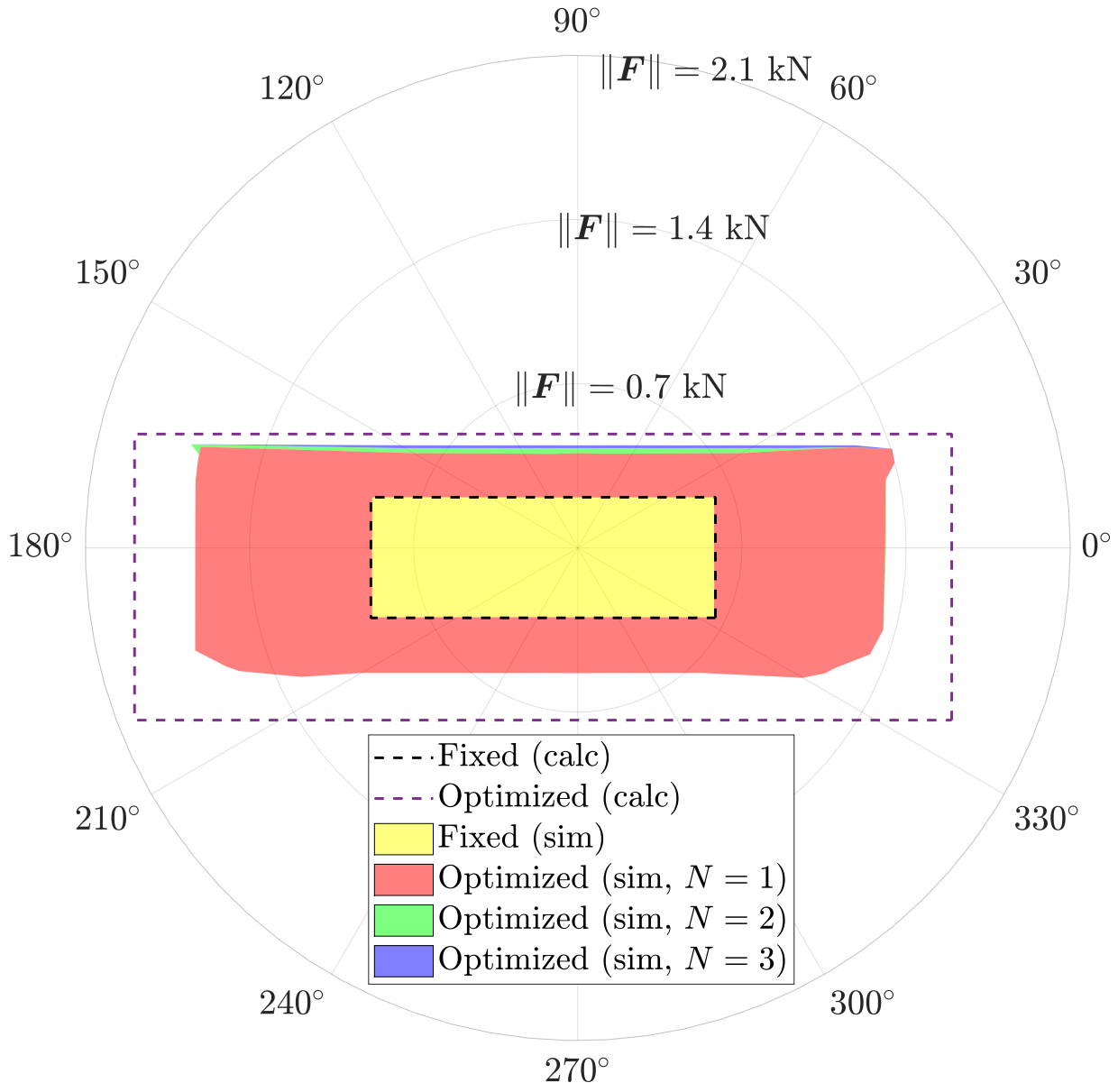


Figure 3.6: System robustness comparison in terms of external force plotted in polar coordinates, e.g., $\theta = 0°$ and $90°$ correspond to positive x and y directions, respectively. The dashed lines indicate the theoretically calculated boundary (i.e., maximum allowable force) for phase duration both fixed and optimized. The shaded areas represent the corresponding simulated results.

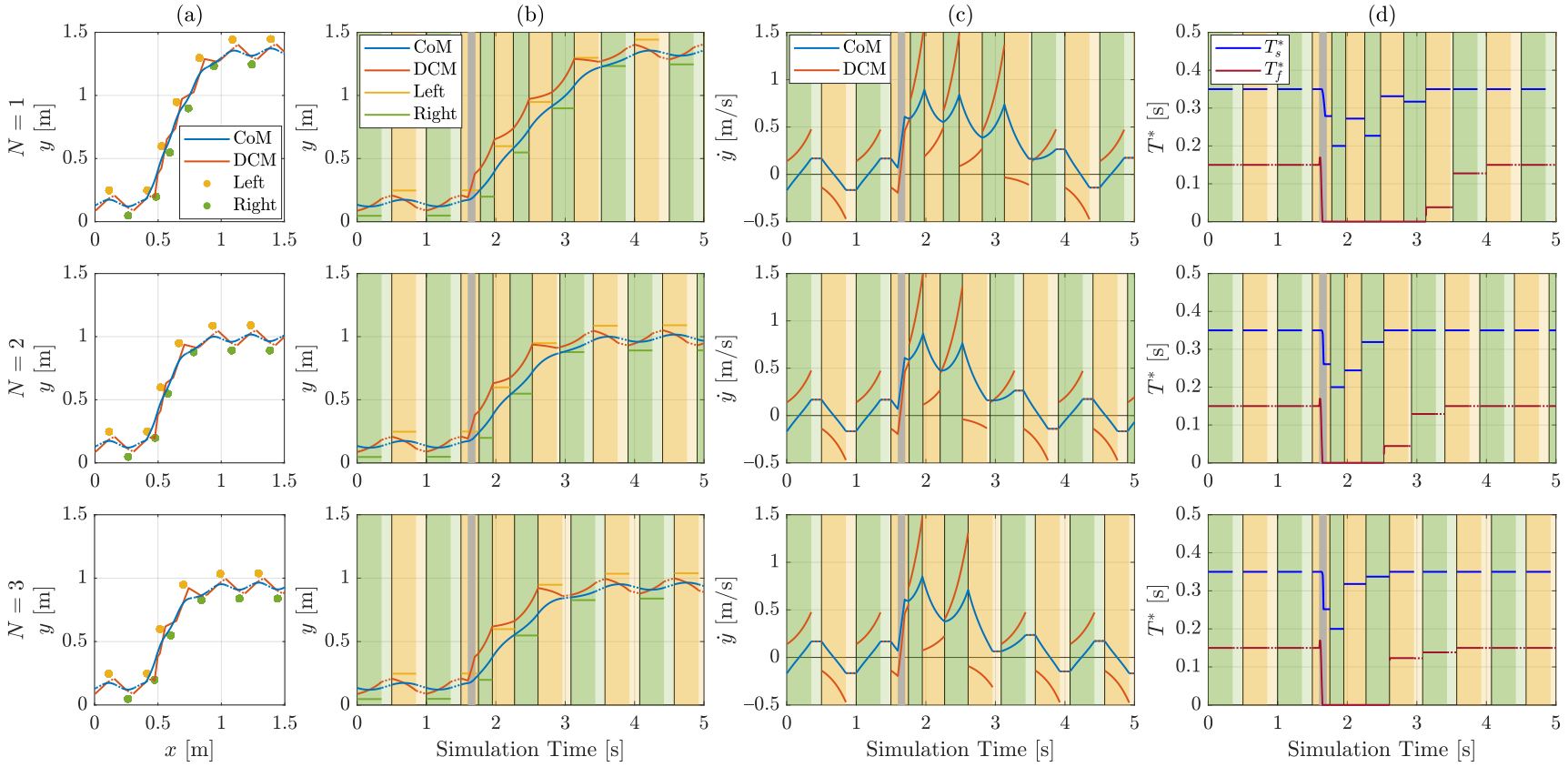


Figure 3.7: Simulation results of $\|\boldsymbol{F}\| = 370$ N and $\theta = 90°$ with phase duration optimized. The first, second, and third rows correspond to $N = 1$, 2, and 3, respectively. (a) The first column shows the position trajectories of the CoM, DCM, left foot, and right foot, where the solid lines indicate stance phase while the dotted lines indicate flight phase. Note that the robot was locomoting to the positive x direction. (b) The second column shows the time series of all the lateral positions. The amber and green shaded areas indicate the gait cycles with left and right stances, respectively. The dark and light shades distinguish the stance and flight phases, respectively. The gray area indicates the duration of the disturbance. (c) The third column shows the time series of the lateral velocities. (d) The fourth column shows the time series of the solved optimal stance and flight durations. Note that during the flight phase, optimizing the previous stance duration of the current gait cycle is meaningless and thus it is excluded.

Chapter 4

Implementation Details

4.1 Introduction

4.1.1 Background

Trajectory optimization (TO) is a powerful technique for gait generation as it considers a time prediction horizon and various physical constraints. However, TO-based approaches heavily depend on system models of the process. Because of the complexity of bipedal robots, e.g., hybrid dynamics, high nonlinearity, strong restrictions, and considerable degrees of freedom (DoF), solving the problem holistically is extremely challenging even offline, i.e., optimizing over contact schedule and contact forces simultaneously with the full-order model [105, 106].

To date, most successful online locomotion control strategies break down the problem into multiple stages and simplify the robot model according to the use of each stage, sacrificing the global optimality while achieving the real-time execution [122, 131, 146]. A practical paradigm is to have a two-level structure [119, 121, 153, 178], where the high level plans the footsteps and the low level controls the whole-body motions while establishing the foot contact as planned.

4.1.2 Motivation and Contribution

Comparatively slower development in bipedal robots than quadruped robots in terms of dynamic behaviors is being noticeable. Besides the limited accessibility to reliable hardware, one of the main reasons is that bipedal robots impose a serious challenge on developing dynamic motion controllers due to the required real-time coordination of multiple tasks in a complex, high-dimensional state space. The two-level structure described in the previous subsection for locomotion purpose has been proven an effective approach for many bipedal robots, e.g., Sarcos [130], Atlas [153], and Cassie [178]. However, practical guidance on implementation of such framework is rarely covered fully in detail or out of date [22, 112]. Complete discussions on implementation details, including but not limited to parameter selection experience and matters needing attention, are also valuable to promote research on dynamic behaviors of bipedal robots, especially for new people to dive into the field.

Meanwhile, investigation on the performance of the state-of-the-art bipedal control strategy for platforms in different scales, e.g. full-size and miniature bipedal robots, is not done yet. Differences between them may put new challenges on both control algorithms and hardware design principles. One of the obvious differences is the lower center of mass (CoM) height of miniature bipedal robots, which will lead to higher natural frequency and quicker diverging speed. As for dynamic locomotion applications, step duration must be reduced accordingly to catch the fall in time but too fast swing trajectories may degenerate the tracking performance.

This chapter discusses similar issues for implementation of a two-level dynamic locomotion controller on a miniature bipedal robot, as presented in Chapter 2. The high-level footstep planner is based on the divergent component of motion (DCM) where both the step location and timing can be optimized for stronger robustness, as presented in Chapter 3. The low-level whole-body control (WBC) is using a weighted quadratic programming (QP) scheme in consideration of computational cost despite sacrificing the strict task priority.

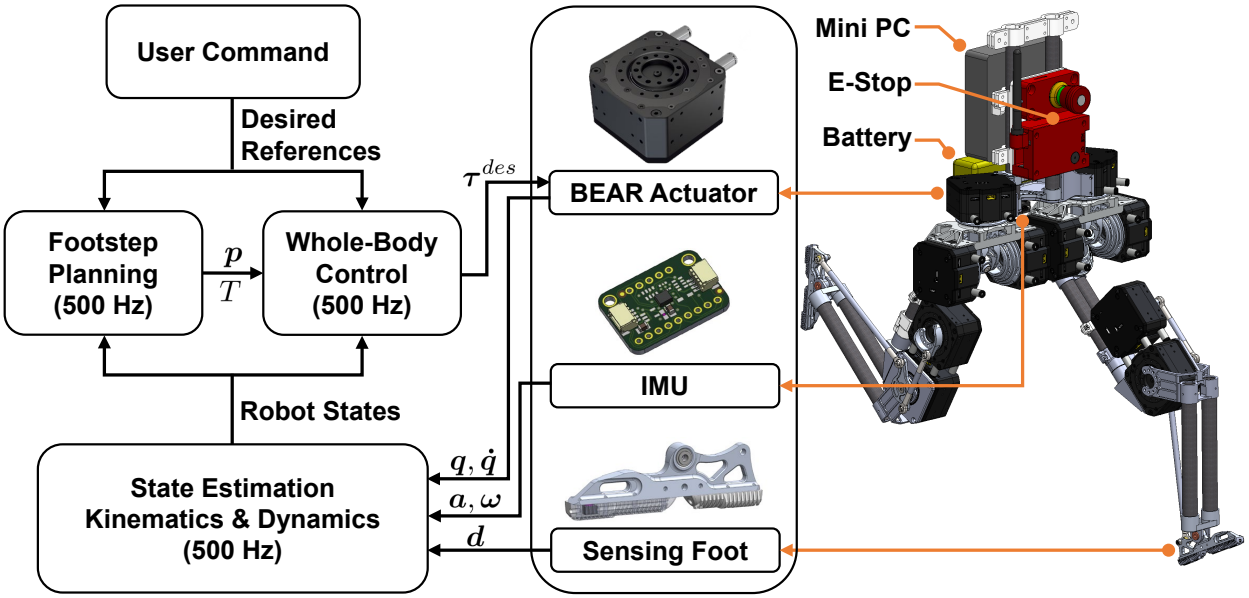


Figure 4.1: System overview. The robot BRUCE is designed to be fully untethered, powered by a LiPo battery, controlled by a mini PC, and equipped with a wireless E-Stop. The state estimator uses a complementary filter that fuses information from the onboard IMU (acceleration $\boldsymbol{a}$ and angular rate $\boldsymbol{\omega}$), the joint encoders (position $\boldsymbol{q}$ and velocity $\dot{\boldsymbol{q}}$), and the sensing feet (boolean contact states $\boldsymbol{d}$). Given the current robot states and desired references, the high-level footstep planner determines the next footstep location $\boldsymbol{p}$ and phase duration T while the low-level WBC calculates the desired joint torques $\boldsymbol{\tau}^{des}$.

4.2 System Overview

Figure 4.1 shows the block diagram for the entire system.

4.2.1 Hardware Platform

As presented in Chapter 2, Bipedal Robot Unit with Compliance Enhanced (BRUCE) is a low-cost miniature bipedal robotic platform for dynamic behaviors. BRUCE has a total height of 660 mm adapting an average human body proportion. It is composed of a torso and two 5-DoF legs. Each leg of BRUCE has a spherical hip joint, a single DoF knee joint, and a single DoF ankle joint. Accordingly, each foot has a line contact with the ground. To have better torque transparency and compliance to the unknown environment, proprioceptive actuation is equipped for each joint, using the Koala BEAR actuators [55]

from Westwood Robotics [160]. Moreover, BRUCE's legs are designed to have low inertia for performing highly dynamic motions. A 2-DoF parallel actuation configuration realized by a cable-driven differential pulley system is applied to the hip joint to reduce the mass and inertia of the femur link. Meanwhile, two pairs of 4-bar linkage mechanisms are used to relocate the ankle actuator to the femur link for the light weight of the tibia link.

To have a convenient hardware experimental setup, it is desired to make BRUCE fully untethered. All the electronics are directly mounted onto the torso for easy installation and better weight distribution. A mini PC with an Intel Core i5-7260U Dual-Core CPU at 2.2 GHz is utilized as the onboard computing resource. A 14.8 V 2200 mAh LiPo battery is used to power the whole system with a running time of approximately 20 minutes. For the sake of safety during operation, a wireless E-Stop built in-house is mounted onboard as well to cut the power in emergencies. In total, BRUCE has a net weight of around 5.2 kg.

4.2.2 Software Architecture

To make BRUCE favorable to dynamic behaviors which require fast response, the overall software framework is developed in a multithreaded environment, which includes a state estimation thread combined with robot model computation, a high-level footstep planning thread, and a low-level WBC thread. The locomotion controller presented in Section 3.4 optimally plans the step location and timing. The low-level WBC presented in Section 4.4 utilizes a weighted QP scheme, leveraging the full-body dynamics to establish the foot contact as planned while regulating other task-space behaviors, e.g., CoM height and torso orientation. Both of them are updated at a frequency of 500 Hz using the off-the-shelf QP solver OSQP [180], which is sufficient for real-time feedback control. The main advantage of multithreading is that the idle time of the central processing unit (CPU) can be kept to minimum since the waiting time for sensor response can be well distributed and utilized, which leads to faster overall execution as the CPU would have been idle if only a single thread was executed. Data communication utilizes a custom shared memory library similar

to the setup developed in [45], All programs are implemented in Python while some parts, including kinematics, dynamics, and state estimation, are precompiled using Numba [181] for acceleration.

4.3 State Estimation

A reliable state estimation is crucial to a good performance of any systems in control. For some systems, e.g., robotic manipulators, the states are directly provided with the on-board sensors. However, for mobile robots which are not fixed in the world frame, their floating-base joint (typically referred to as the robot main body) states cannot be measured directly and are estimated in different ways according to the specific system and task. When it comes to legged robots, an accurate estimate of their floating-base joint is of particular importance since it takes the lead in describing the system dynamic behavior. Unlike other mobile robots, legged robots experience intermittent contact with the ground during locomotion, which suggests a unique state estimation approach by combining both the on-board inertial measurement unit (IMU) information and the kinematic model of the robot. While Kalman filter (KF) has been widely used for legged state estimation in various forms [182, 183], its global convergence is not guaranteed for nonlinear systems. In addition, it is computationally expensive, not to mention the annoying tuning of the numerous covariance parameters in order to give a reasonable performance. Meanwhile, simple linear filters, e.g., complementary filter (CF), work robustly in practice [92]. As an aside, CF is simply a weighted combination of different measurements which are complementary, but in essence it applies low-pass filtering to the noisy measurements and high-pass filtering to the biased measurements. In the end, CF gives us the best of both worlds, i.e., no noise and no drift.

In this section, we will go over the state estimation of BRUCE using CF in detail. Two types of measurement sources are used, i.e., one is the on-board IMU sensor (ISM330DHCX) which measures the base acceleration and angular velocity while the other is Koala BEAR

actuator encoder sensor which gives the rotor rotational position and velocity. Note that the sensor raw data are processed before use for better results. The estimation of the base rotational part (orientation and angular velocity) and linear part (position and velocity) are decoupled. In specific, we use the IMU solely for base rotational estimation while we use both sensors for base linear estimation. Once we have the floating-base joint estimation results, we can use the robot kinematic model in Section 2.3 as well as the dynamic model in Section 2.4 to directly get other quantities of interest, e.g., foot position and orientation, foot linear and angular velocities, CoM position and velocity.

4.3.1 Sensor Raw Data Processing

On-Board IMU Sensor

At any time t_k, the angular velocity raw measurement ${}^b\widetilde{\boldsymbol{\omega}}_k$ from the gyroscope and the linear acceleration raw measurement ${}^b\widetilde{\boldsymbol{a}}_k^g$ (the superscript g indicates it also includes the gravitational acceleration $\boldsymbol{g} = [0, 0, -g]^\top$, where $g = 9.81$ m/s^2) from the accelerometer are first unbiased:

$$ {}^b\overline{\boldsymbol{\omega}}_k^g = {}^b\widetilde{\boldsymbol{\omega}}_k - {}^b\boldsymbol{b}_\omega, \tag{4.1} $$

$$ {}^b\overline{\boldsymbol{a}}_k^g = {}^b\widetilde{\boldsymbol{a}}_k^g - {}^b\boldsymbol{b}_a, \tag{4.2} $$

where ${}^b\boldsymbol{b}_\omega$ and ${}^b\boldsymbol{b}_a$ are the measured biases. Note that the raw readings are both expressed in the IMU frame which is aligned with the base frame. The signals are then smoothed out using an exponential filter (acting as low-pass filters to remove high-frequency noises):

$$ {}^b\boldsymbol{\omega}_k = \alpha_\omega\, {}^b\overline{\boldsymbol{\omega}}_k^g + (1 - \alpha_\omega)\, {}^b\boldsymbol{\omega}_{k-1}, \qquad {}^b\boldsymbol{\omega}_0 = {}^b\overline{\boldsymbol{\omega}}_0, \tag{4.3} $$

$$ {}^b\boldsymbol{a}_k^g = \alpha_a\, {}^b\overline{\boldsymbol{a}}_k^g + (1 - \alpha_a)\, {}^b\boldsymbol{a}_{k-1}^g, \qquad {}^b\boldsymbol{a}_0^g = {}^b\overline{\boldsymbol{a}}_0^g, \tag{4.4} $$

where $0 \leq \alpha_\omega, \alpha_a \leq 1$ are the smoothing factors, e.g., we choose $\alpha_\omega = 0.1$ and $\alpha_a = 0.1$. We regard ${}^b\boldsymbol{\omega}_k$ as a good estimate of the angular velocity while we will further work on ${}^b\boldsymbol{a}_k^g$.

Koala BEAR Actuator Encoder Sensor

At any time t_k, the rotational position raw measurement $\boldsymbol{\varphi}_k$ from the encoders are used directly to get the joint positions $\boldsymbol{\theta}_k$ using (2.114) and (2.115). The rotational velocity raw measurement $\widetilde{\dot{\boldsymbol{\varphi}}}_k$ are first smoothed out using an exponential filter (the raw velocity feedback from the encoder is simply the difference between two consecutive position readings divided by the encoder sampling period so it is pretty noisy as the sampling period is extremely short, e.g., typically at 2 kHz):

$$\dot{\boldsymbol{\varphi}}_k = \alpha_{\dot{\varphi}} \widetilde{\dot{\boldsymbol{\varphi}}}_k + \left(1 - \alpha_{\dot{\varphi}}\right) \dot{\boldsymbol{\varphi}}_{k-1}, \qquad \dot{\boldsymbol{\varphi}}_0 = \widetilde{\dot{\boldsymbol{\varphi}}}_0, \tag{4.5}$$

where we choose the smoothing factor $\alpha_{\dot{\varphi}} = 0.1$. We then compute the corresponding joint velocities $\dot{\boldsymbol{\theta}}_k$ using (2.116) and (2.117).

4.3.2 Base Orientation Estimation

For estimating the base orientation, given the previous estimation results $\boldsymbol{R}_{k-1}$ and ${}^b\boldsymbol{\omega}_{k-1}$, we can first predict for the current iteration (using the forward Euler method and assuming zero angular acceleration):

$$\widetilde{\boldsymbol{R}}_k = \boldsymbol{R}_{k-1} \mathrm{Exp}\left({}^b\boldsymbol{\omega}_{k-1} \Delta t\right), \tag{4.6}$$

where Δt is the sampling period.

If the base external acceleration (excluding $\boldsymbol{g}$) is small, i.e., the accelerometer exclusively measures gravity, which means it should point straight downward in the world frame, i.e.,

ideally we should have

$$\boldsymbol{R}_k {}^b\boldsymbol{a}_k^g = \boldsymbol{g}. \tag{4.7}$$

Note that although we know both ${}^b\boldsymbol{a}_k^g$ and $\boldsymbol{g}$, $\boldsymbol{R}_k$ generally cannot be solved since there will be infinite numbers of solutions. However, due to sensor noise and integration error, we will probably get something different:

$$\widetilde{\boldsymbol{R}}_k {}^b\boldsymbol{a}_k^g = \widetilde{\boldsymbol{g}}_k. \tag{4.8}$$

We can thus correct $\widetilde{\boldsymbol{R}}_k$ (particularly tilt, i.e., pitch and roll) with some small rotation $\delta\boldsymbol{R}_k$ (to be determined), i.e., $\boldsymbol{R}_k = \delta\boldsymbol{R}_k\widetilde{\boldsymbol{R}}_k$, such that

$$\delta\boldsymbol{R}_k\widetilde{\boldsymbol{g}}_k = \boldsymbol{g}. \tag{4.9}$$

There are several ways one could go about calculating $\delta\boldsymbol{R}_k$ but one intuitive way is to find the minimum rotation between $\widetilde{\boldsymbol{g}}_k$ and $\boldsymbol{g}$ (or simply the $-\boldsymbol{z}$ axis) using the cross product along with the dot product:

$$\widetilde{\boldsymbol{g}}_k \times \boldsymbol{g} = -g \sin\delta\phi_k \|\widetilde{\boldsymbol{g}}_k\| \boldsymbol{n}_k, \tag{4.10}$$

$$\widetilde{\boldsymbol{g}}_k \cdot \boldsymbol{g} = -g \cos\delta\phi_k \|\widetilde{\boldsymbol{g}}_k\|, \tag{4.11}$$

where $\boldsymbol{n}_k$ is the rotation axis and $\delta\phi_k$ is the minimum angle. We can solve

$$\delta\phi_k = \arccos\left(\frac{\widetilde{\boldsymbol{g}}_k \cdot \boldsymbol{g}}{-g\|\widetilde{\boldsymbol{g}}_k\|}\right), \tag{4.12}$$

$$\boldsymbol{n}_k = \begin{cases} \dfrac{\widetilde{\boldsymbol{g}}_k \times \boldsymbol{g}}{-g \sin\delta\phi_k \|\widetilde{\boldsymbol{g}}_k\|} & \text{if } \sin\delta\phi_k \neq 0, \\ \text{arbitrary} & \text{if } \sin\delta\phi_k = 0. \end{cases} \tag{4.13}$$

Let us now choose

$$\delta \boldsymbol{R}_k = \mathrm{Exp}\left(\alpha_\phi \delta\phi_k \boldsymbol{n}_k\right), \tag{4.14}$$

where the user-defined parameter $0 \leq \alpha_\phi \leq 1$ decides how aggressively tilt correction is being applied, e.g., $\alpha_\phi = 0$ means no correction at all and we choose $\alpha_\phi = 0.001$. Finally, the corrected orientation is given by

$$\boldsymbol{R}_k = \delta \boldsymbol{R}_k \widetilde{\boldsymbol{R}}_k = \mathrm{Exp}\left(\alpha_\phi \delta\phi_k \boldsymbol{n}_k\right) \boldsymbol{R}_{k-1} \mathrm{Exp}\left({}^b\boldsymbol{\omega}_{k-1} \Delta t\right). \tag{4.15}$$

In general, this naive orientation estimation approach should have heading (or yaw) drift issue over time as there is nothing to correct it, e.g., a magnetometer could resolve. Surprisingly, this issue is almost unnoticeable with the ISM330DHCX module during nominal operation. By the way, people usually do not trust the heading direction estimation result without exteroceptive information, e.g., vision, and therefore do not control the absolute heading of the robots with only proprioceptive sensors (like what we do here).

4.3.3 Base Position and Velocity Estimation

For estimating the base position and velocity, given the previous estimation results $\boldsymbol{p}_{k-1}$ and $\boldsymbol{v}_{k-1}$, we can first predict for the current iteration:

$$\widetilde{\boldsymbol{p}}_k = \boldsymbol{p}_{k-1} + \boldsymbol{v}_{k-1}\Delta t + \frac{1}{2}\boldsymbol{a}_{k-1}\Delta t^2, \tag{4.16}$$

$$\widetilde{\boldsymbol{v}}_k = \boldsymbol{v}_{k-1} + \boldsymbol{a}_{k-1}\Delta t, \tag{4.17}$$

where

$$\boldsymbol{a}_{k-1} = \boldsymbol{R}_{k-1}\, {}^b\boldsymbol{a}^g_{k-1} - \boldsymbol{g} \tag{4.18}$$

is the acceleration measurement compensated with gravity (thus without the superscript g).

We can then correct these predictions with the kinematic model using the complementary filter. For the position correction, given the stance foot location $\boldsymbol{p}_m^{leg}$ (we consider the middle point as the foot location) in the world frame as well as ${}^b\boldsymbol{p}_m^{leg}\left(\boldsymbol{\theta}_k^{leg}\right)$ (as a function of $\boldsymbol{\theta}_k^{leg}$) in the base frame, we can reversely compute the base position using (2.121) such that

$$\overline{\boldsymbol{p}}_k^{leg} = \boldsymbol{p}_m^{leg} - \boldsymbol{R}_k\, {}^b\boldsymbol{p}_m^{leg}. \tag{4.19}$$

The overall base position $\overline{\boldsymbol{p}}_k$ from the kinematic model will be the average if both legs are in stance. Now combining the prediction and the kinematic model yields

$$\boldsymbol{p}_k = \alpha_p \overline{\boldsymbol{p}}_k + \left(1 - \alpha_p\right) \widetilde{\boldsymbol{p}}_k, \tag{4.20}$$

where the blending weight $0 \leq \alpha_p \leq 1$ can be experimentally tuned, e.g., we have $\alpha_p = 0.1$. Note that the stance foot location is explicitly fixed when it is in contact with the ground. When the contact is broken, it will not be considered in the position correction (same for the velocity correction later). When the contact is created again, the foot location is updated using the kinematic model and then fixed again. As a result, the base position estimation cannot handle foot slippage, not to mention inaccurate stance foot location update. While the global position accuracy is not guaranteed, the estimation works fairly well locally for each step, which suffices our application.

For the velocity correction, given the stance foot velocity ${}^b\boldsymbol{v}_m^{leg}\left(\boldsymbol{\theta}_k^{leg}, \dot{\boldsymbol{\theta}}_k^{leg}\right) = {}_v^b\boldsymbol{J}_m^{leg}\left(\boldsymbol{\theta}_k^{leg}\right)\dot{\boldsymbol{\theta}}_k^{leg}$ (as a function of both $\boldsymbol{\theta}_k^{leg}$ and $\dot{\boldsymbol{\theta}}_k^{leg}$) in the base frame, the base velocity can be reversely solved using (2.124) such that

$$\overline{\boldsymbol{v}}_k^{leg} = -\boldsymbol{R}_k \left(\widehat{{}^b\boldsymbol{\omega}_k}\, {}^b\boldsymbol{p}_m^{leg} + {}^b\boldsymbol{v}_m^{leg} \right), \tag{4.21}$$

where we assume the stance foot velocity is always zero in the world frame. The overall base

velocity $\overline{\boldsymbol{v}}_k$ will be the average of the stance legs. Now fusing together the prediction and the kinematic model yields

$$\boldsymbol{v}_k = \alpha_v \overline{\boldsymbol{v}}_k + (1 - \alpha_v)\, \widetilde{\boldsymbol{v}}_k, \tag{4.22}$$

where the blending weight $0 \leq \alpha_v \leq 1$ can be experimentally tuned and we choose $\alpha_v = 0.1$.

4.4 Whole-Body Control

The joint-space equations of motion for a bipedal robot can be written in the following canonical form:

$$\boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G} = \boldsymbol{S}_a^\top \boldsymbol{\tau} + \sum_{j=1}^{N_c} \boldsymbol{J}_{c_j}^\top \boldsymbol{f}_j, \tag{4.23}$$

where $\boldsymbol{q}$ is the vector of generalized coordinates, $\boldsymbol{H}$ is the inertia matrix, $\boldsymbol{C}$ is the vector of centrifugal and Coriolis terms, $\boldsymbol{G}$ is the gravity vector, $\boldsymbol{S}_a$ is the actuation selection matrix, $\boldsymbol{\tau}$ is the joint torque vector, $\boldsymbol{J}_{c_j}$ and $\boldsymbol{f}_j$ are respectively the foot contact Jacobian and contact force at the jth contact vertex, and N_c is the number of contact vertices.

Given desired operational space acceleration $\ddot{\boldsymbol{x}}_i^{des}$ for the ith task, the goal of WBC is then to find the instantaneously required, dynamically consistent $\ddot{\boldsymbol{q}}$, $\boldsymbol{\tau}$, and $\boldsymbol{f}_j$. This can be done using a weighted WBC scheme, which is formulated as the following optimization (a QP problem essentially):

$$\min_{\ddot{\boldsymbol{q}}, \boldsymbol{f}_j} \; \sum_{i=1}^{N_t} \left\| \boldsymbol{J}_i \ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}} - \ddot{\boldsymbol{x}}_i^{des} \right\|_{\boldsymbol{W}_i}^2 + \sum_{j=1}^{N_c} \left\| \boldsymbol{f}_j \right\|_{\boldsymbol{W}_f}^2 + \left\| \ddot{\boldsymbol{q}} \right\|_{\boldsymbol{W}_{\ddot{q}}}^2 \tag{4.24}$$

$$\text{s.t.} \;\; \boldsymbol{S}_f \left(\boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G} - \sum_{j=1}^{N_c} \boldsymbol{J}_{c_j}^\top \boldsymbol{f}_j \right) = \boldsymbol{0}, \tag{4.25}$$

$$\boldsymbol{f}_j \in \boldsymbol{\mathcal{C}}_j, \quad j = 1, \cdots, N_c, \tag{4.26}$$

where $\boldsymbol{J}_i$ is the ith task Jacobian and N_t is the number of tasks. As we can see, the ith operational task is set as a QP cost with priority implicitly being enforced with weight $\boldsymbol{W}_i$. In particular, $\ddot{\boldsymbol{x}}_i^{des}$ consists of both feedforward and feedback terms, which is specified with the form of

$$\ddot{\boldsymbol{x}}_i^{des} = \boldsymbol{a}_i^{ref} + \boldsymbol{K}_p \left(\boldsymbol{p}_i^{ref} - \boldsymbol{p}_i \right) + \boldsymbol{K}_d \left(\boldsymbol{v}_i^{ref} - \boldsymbol{v}_i \right), \tag{4.27}$$

$$\ddot{\boldsymbol{x}}_i^{des} = \boldsymbol{\alpha}_i^{ref} + \boldsymbol{K}_p \mathrm{Log} \left(\boldsymbol{R}_i^{\top} \boldsymbol{R}_i^{ref} \right) + \boldsymbol{K}_d \left(\boldsymbol{\omega}_i^{ref} - \boldsymbol{\omega}_i \right), \tag{4.28}$$

for the linear and angular motion, respectively, where $\boldsymbol{a}_i$, $\boldsymbol{v}_i$, $\boldsymbol{p}_i$ are the linear acceleration, velocity, position while $\boldsymbol{\alpha}_i$, $\boldsymbol{\omega}_i$, $\boldsymbol{R}_i$ are the angular acceleration, velocity, orientation, the superscript ref corresponds to the reference, and $\boldsymbol{K}_{p/d}$ is the proportional/derivative (P/D) feedback gain matrix. Table 4.1 shows all the task weights $\boldsymbol{W}$ (first row) and gains $\boldsymbol{K}_{p/d}$ (second/third row) for our system. Note that $\boldsymbol{W}$, $\boldsymbol{K}_p$, and $\boldsymbol{K}_d$ are all 3×3 diagonal positive semi-definite matrices so only the diagonal elements are listed in the order of x, y, z direction. The symbol / means the object is not used. In addition to the task costs, regularization costs are added to the decision variables $\ddot{\boldsymbol{q}}$ and $\boldsymbol{f}_j$ with small weights $\boldsymbol{W}_{\ddot{q}}$ and $\boldsymbol{W}_f$ respectively to ensure the overall QP cost is strictly positive definite even when the task Jacobians contain singularities, which avoids potential numerical issues. Let us now go over the details of the WBC framework.

Table 4.1: WBC task weight and gain setup.

Task	**Right Stance**	**Left Stance**	**Double Stance**	**Flight**
Angular Momentum	$(1, 1, 1)$	$(1, 1, 1)$	$(1, 1, 1)$	/
	/	/	/	/
	$(10, 10, 1)$	$(10, 10, 1)$	$(10, 10, 1)$	/

Linear Momentum	$(1, 1, 100)$	$(1, 1, 100)$	$(10, 10, 100)$	/
	$(1, 1, 100)$	$(1, 1, 100)$	$(50, 50, 200)$	/
	$(5, 5, 10)$	$(5, 5, 10)$	$(10, 10, 15)$	/
Torso Orientation	$(10, 10, 10)$	$(10, 10, 10)$	$(10, 10, 10)$	$(10, 10, 10)$
	$(500, 500, 200)$	$(500, 500, 200)$	$(500, 500, 200)$	$(500, 500, 200)$
	$(25, 25, 20)$	$(25, 25, 20)$	$(25, 25, 20)$	$(25, 25, 20)$
Right	$(10^3, 10^3, 10^3)$	/	$(10^3, 10^3, 10^3)$	/
Stance	/	/	/	/
Contact	/	/	/	/
Right	/	$(10, 10, 10)$	/	$(10, 10, 10)$
Swing	/	$(100, 100, 100)$	/	$(100, 100, 100)$
Position	/	$(10, 10, 10)$	/	$(10, 10, 10)$
Right	/	$(/, 1, 1)$	/	$(/, 1, 1)$
Swing	/	$(/, 50, 300)$	/	$(/, 50, 300)$
Orientation	/	$(/, 10, 50)$	/	$(/, 10, 50)$
Left	/	$(10^3, 10^3, 10^3)$	$(10^3, 10^3, 10^3)$	/
Stance	/	/	/	/
Contact	/	/	/	/
Left	$(10, 10, 10)$	/	/	$(10, 10, 10)$
Swing	$(100, 100, 100)$	/	/	$(100, 100, 100)$
Position	$(10, 10, 10)$	/	/	$(10, 10, 10)$
Left	$(/, 1, 1)$	/	/	$(/, 1, 1)$
Swing	$(/, 50, 300)$	/	/	$(/, 50, 300)$
Orientation	$(/, 10, 50)$	/	/	$(/, 10, 50)$

4.4.1 System Dynamics

The consistency of the variables $\ddot{\boldsymbol{q}}$, $\boldsymbol{\tau}$, and $\boldsymbol{f}_j$ with the system dynamics (4.23) must be strictly enforced. Notably, the equations can be split into the floating-base dynamics and joint dynamics. To accelerate the QP performance, only the floating-base dynamics are considered as (4.25), where $\boldsymbol{S}_f$ is the base selection matrix. In this manner, variables for $\boldsymbol{\tau}$ can be removed if it is assumed that there is always enough torque to achieve the generated motion, i.e., no torque limits. Once the QP is solved with optimal solution $\ddot{\boldsymbol{q}}^*$ and $\boldsymbol{f}_j^*$, the corresponding joint torques can be computed as follows:

$$\boldsymbol{\tau}^* = \boldsymbol{S}_a \left(\boldsymbol{H}\ddot{\boldsymbol{q}}^* + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G} - \sum_{j=1}^{N_c} \boldsymbol{J}_{c_j}^\top \boldsymbol{f}_j^* \right). \tag{4.29}$$

4.4.2 Centroidal Momentum

The control of centroidal momentum is a critical component of WBC for bipedal robots, which consists of the linear momentum $\boldsymbol{l}_G$ as well as the centroidal angular momentum (CAM) $\boldsymbol{k}_G$ about the robot CoM. While the linear part has a straightforward relationship with the CoM velocity, i.e., (2.185), the angular part is abstract. Notably, biomechanics studies have shown that for human walking, the CAM is well regulated to near zero by the neuro-control system [67]. Accordingly, the angular momentum task is just to damp out the excessive CAM:

$$\dot{\boldsymbol{k}}_G^{des} = -\boldsymbol{K}_d \boldsymbol{k}_G. \tag{4.30}$$

The linear momentum task is decoupled in the vertical and horizontal directions. For the vertical direction, we want the robot CoM to maintain a nominal height above the ground to match the LIP in Section 3.2.1. For the horizontal direction, it is simply tracking the velocity command with low priority and PD gains since the robot movement is mainly realized by taking steps. Another reason is that our BRUCE robot has only five DoFs for each leg,

which means during single stance not all the six spatial DoFs can be directly controlled. We prioritize the CoM height and torso orientation for a good posture, which generally contributes to the locomotion stability. Note that for the momentum tasks, the centroidal momentum matrix [169] is used as the task Jacobian, as discussed in Section 2.4.2.

4.4.3 Torso Orientation

Controlling the torso orientation is essential for a good locomotion posture, e.g., to avoid unwanted torso oscillations which can largely affect the robot CoM and complicate the control process. Accordingly, since we also have a small yaw drift rate, all the three angles are controlled globally.

4.4.4 Stance Foot

For BRUCE with line feet, we consider two point contacts per foot, the toe and heel. The constraint (4.26) ensures each contact force is bounded and lies within the local friction cone $\boldsymbol{\mathcal{C}}_j$ which is approximated by a square pyramid for linearity, e.g., on the even ground we have

$$\underbrace{\begin{bmatrix} \pm 1 & 0 & -\mu \\ 0 & \pm 1 & -\mu \\ 0 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{G}_f} \boldsymbol{f}_j \leq \underbrace{\begin{bmatrix} 0 \\ 0 \\ -f_{z,\min} \\ f_{z,\max} \end{bmatrix}}_{\boldsymbol{h}_f}, \tag{4.31}$$

where μ is the friction coefficient and $f_{z,\min\,/\,\max}$ is the minimum/maximum normal force. Note that a positive minimum can prevent loose contact, or otherwise one foot might lose contact if the CoM is shifted to the other one.

Moreover, for each contact force constraint, a corresponding zero contact acceleration

constraint needs to be specified to prevent the stance foot from moving:

$$\boldsymbol{J}_{c_j}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{c_j}\dot{\boldsymbol{q}} = \mathbf{0}. \tag{4.32}$$

In practice, we treat (4.32) as one of the task-space objectives, i.e., a soft constraint, which can generally speed up the QP and give better numerical stability [146]. Then with sufficient task weight, it will act as a nullspace projector so that other tasks will properly respect the nonmoving contact condition.

4.4.5 Swing Foot

For a multi-layered control scheme, accurate execution of the high-level plan is important for the low-level controller. For our case, the high-level planner determines when and where to take the next step in an optimal manner, which is essentially realized by the swing foot task of the WBC. Note that for the foot position, instead of considering some point on the foot bottom which is sensitive to the ankle joint, we choose to control the ankle position for simplicity. As a result, the ankle joint is only responsible for foot orientation. This decoupling is also beneficial for the tuning process.

Trajectory Generation

The swing foot orientation can be simply set constant relative to the torso. However, recall that each leg of BRUCE has only five DoFs and thus the rotation in the foot roll direction is excluded since it has the least effect for a line foot. In addition, because the robot is walking blindly with no terrain information, the gain in the pitch direction is intentionally set low, which can make the foot adaptive to a certain range of terrains.

The swing foot position trajectory needs to be carefully designed to adapt the changing footstep location and timing. In the horizontal direction at some time t_0, when a new step is planned with optimal solution $\boldsymbol{p}$ and T, the reference trajectory is regenerated using a

fifth-order polynomial to ensure continuity up to acceleration:

$$p_{x/y}^{ref}(t) = \sum_{i=0}^{5} c_i t^i = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 + c_5 t^5, \tag{4.33}$$

$$\dot{p}_{x/y}^{ref}(t) = \sum_{i=1}^{5} c_i i t^{i-1} = c_1 + 2c_2 t + 3c_3 t^2 + 4c_4 t^3 + 5c_5 t^4, \tag{4.34}$$

$$\ddot{p}_{x/y}^{ref}(t) = \sum_{i=2}^{5} c_i i(i-1) t^{i-2} = 2c_2 + 6c_3 t + 12c_4 t^2 + 20c_5 t^3, \tag{4.35}$$

with boundary conditions given as follows:

$$p_{x/y}^{ref}(t_0) = p_{x/y}^{pre}(t_0), \tag{4.36}$$

$$\dot{p}_{x/y}^{ref}(t_0) = \dot{p}_{x/y}^{pre}(t_0), \tag{4.37}$$

$$\ddot{p}_{x/y}^{ref}(t_0) = \ddot{p}_{x/y}^{pre}(t_0), \tag{4.38}$$

$$p_{x/y}^{ref}(T) = p_{x/y}, \tag{4.39}$$

$$\dot{p}_{x/y}^{ref}(T) = 0, \tag{4.40}$$

$$\ddot{p}_{x/y}^{ref}(T) = 0, \tag{4.41}$$

where $\boldsymbol{p}^{pre}(t)$ is the previously generated trajectory. We can then solve for the coefficients as follows:

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & T & T^2 & T^3 & T^4 & T^5 \\ 0 & 1 & 2T & 3T^2 & 4T^3 & 5T^4 \\ 0 & 0 & 2 & 6T & 12T^2 & 20T^3 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} p_{x/y}^{pre}(t_0) \\ \dot{p}_{x/y}^{pre}(t_0) \\ \ddot{p}_{x/y}^{pre}(t_0) \\ p_{x/y} \\ 0 \\ 0 \end{bmatrix}. \tag{4.42}$$

In the vertical direction, the swing foot height first increases to a fixed apex value until a fixed time, and then decreases to prepare for landing on the ground with a similar trajectory

regeneration method in the horizontal direction. In addition, due to modeling and state estimation errors, the landing height of the swing foot needs to be adjusted based on the stance foot to mitigate the touchdown impact.

Inverse Kinematics Compensation

While ID-based WBCs such as ours are able to provide compliant behaviors and strong robustness, they heavily depend on the high quality of the dynamic model which is often difficult to obtain in practice. In addition, considering the short step duration for BRUCE, the swing foot acceleration is usually significant and thus accurate control of the foot movement is typically hard even with a good system identification. On the contrary, IK-based approaches only require the robot kinematic model which is much easier to get. On top of that, utilizing joint position PD control benefits bipedal systems due to its modeling error compensation and high updating frequency [146, 184], e.g., the BEAR actuator runs internal control loop at 2 kHz which is four times faster than our WBC. As a result, in addition to the optimal joint torques (4.29) from the WBC, i.e., think of it as the feedforward term, we take into account the joint position and velocity references by solving the swing foot IK, which can greatly enhance the tracking performance:

$$\tau^{des} = \tau^{*} + k_p \left(q^{ref} - q\right) + k_d \left(\dot{q}^{ref} - \dot{q}\right), \tag{4.43}$$

where $k_{p/d}$ is the P/D feedback gain for each joint.

4.4.6 Task Transition

During contact changes, i.e., foot transition from stance to swing and vice versa, task transition needs to be performed, e.g., for a stance foot, the stance contact task is activated while the swing position and orientation tasks are deactivated. This can be handled by simply changing the relative task weight, e.g., if a task is deactivated, its weight can be assigned

zero or tiny value and we can easily bring it back if it becomes activated again. Note that foot contact transition happens instantaneously and thus smooth task transition [185] is unnecessary for the locomotion scenario. No chatter of the joint acceleration is observed either in the simulation or on the hardware. In addition, the contact force constraint (4.31) needs to be adjusted accordingly as well, e.g., the limit $f_{z,\min\,/\,\max}$ should be reduced to zero for a swing foot.

4.4.7 QP Formulation

The weighted WBC optimization can be manipulated and rearranged into the standard QP form as (A.3), where

$$\boldsymbol{x} = \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \hline \boldsymbol{f}_{rt} \\ \boldsymbol{f}_{rh} \\ \boldsymbol{f}_{lt} \\ \boldsymbol{f}_{lh} \end{bmatrix}, \tag{4.44}$$

$$\boldsymbol{P} = 2 \left[\begin{array}{c|cccc} \boldsymbol{W}_{\ddot{q}} + \sum_{i=1}^{N_t} \boldsymbol{J}_i^\top \boldsymbol{W}_i \boldsymbol{J}_i & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{W}_f & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_f & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_f & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_f \end{array}\right], \tag{4.45}$$

$$\boldsymbol{q} = 2 \begin{bmatrix} \sum_{i=1}^{N_t} \boldsymbol{J}_i^\top \boldsymbol{W}_i \left(\dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}} - \ddot{\boldsymbol{x}}_i^{des} \right) \\ \hline \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}, \tag{4.46}$$

$$\boldsymbol{G} = \left[\begin{array}{c|cccc} \boldsymbol{0} & \boldsymbol{G}_f & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{G}_f & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{G}_f & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{G}_f \end{array}\right], \tag{4.47}$$

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{h}_f \\ \boldsymbol{h}_f \\ \boldsymbol{h}_f \\ \boldsymbol{h}_f \end{bmatrix}, \tag{4.48}$$

$$\boldsymbol{A} = \left[\begin{array}{c|cccc} \boldsymbol{S}_f \boldsymbol{H} & -\boldsymbol{S}_f \boldsymbol{J}_{c_{rt}}^\top & -\boldsymbol{S}_f \boldsymbol{J}_{c_{rh}}^\top & -\boldsymbol{S}_f \boldsymbol{J}_{c_{lt}}^\top & -\boldsymbol{S}_f \boldsymbol{J}_{c_{lh}}^\top \end{array}\right], \tag{4.49}$$

$$\boldsymbol{b} = -\boldsymbol{S}_f \left(\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G}\right). \tag{4.50}$$

Note that the subscript rt, rh, lt, and lh indicate the right toe, right heel, left toe, and left heel, respectively.

4.5 Experimental Results

Finally, to verify the effectiveness of the proposed approach on the full bipedal robot BRUCE, an extensive series of simulation and hardware experiments were conducted. The simulation of BRUCE is built based on Gazebo [186], an open-source 3D robotics simulator using the ODE physics engine. The hardware experiments were conducted fully untethered.

4.5.1 Simulation Results

Versatile Dynamic Bipedal Locomotion Gaits

We first demonstrate that our controller is able to stabilize several typical dynamic bipedal locomotion gaits, including walking, running, single-legged hopping, and double-legged hopping, as shown in Figure 4.2. Note that for the single-stance phase, the stance foot center

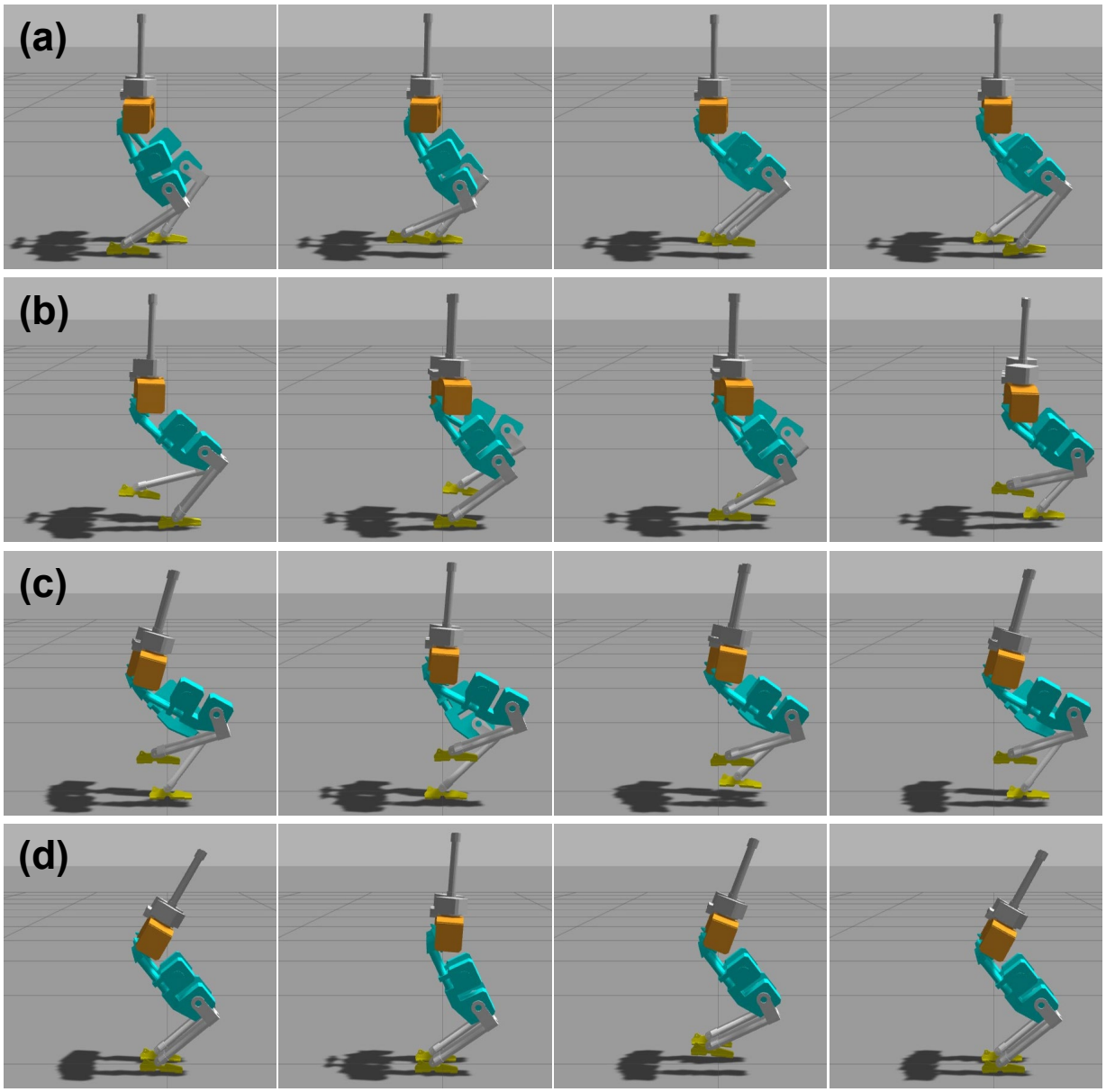


Figure 4.2: Snapshots of one complete gait cycle for BRUCE (a) walking, (b) running, (c) single-legged hopping, and (d) double-legged hopping.

is simply considered as the CoP location while for the double-stance phase, the midpoint of the two foot centers is considered as the CoP. In addition, for the gaits with a flight phase, a constant CoM height reference during the stance phase is good enough to stabilize the vertical CoM motion thanks to the compliance of the WBC. The controller is also versatile. Besides the typical bipedal locomotion gaits mentioned before, it is also possible to generate any arbitrary gaits by simply modifying the nominal gait parameters in Section 3.3.2, e.g.,

some interesting gaits just like dancing can be produced by having a negative lateral footstep clearance d_y or even introducing a longitudinal footstep clearance d_x. We can also change the time parameters to make each step slower or faster.

Omnidirectional Locomotion

In this experiment, to verify DCM-based footstep planning is effective for generating stable locomotion in general, the robot was commanded and managed to locomote omnidirectionally, e.g., forward and backward, left and right, stationary yaw rotation, as well as any combinations of them. Note that only a velocity command needs to be specified, e.g., longitudinal and lateral velocity, yaw rate. In addition, due to modeling error, a velocity calibration is needed as the robot may drift slightly even with a zero velocity command.

Push Recovery

We then demonstrate the strong robustness of our controller with a push recovery test. In the first scenario, as shown in Figure 4.3, when BRUCE was locomoting in place, external pushes were exerted on its torso in both lateral and longitudinal directions, and BRUCE was able to recover within the next few steps by adapting both step location and timing. In the second scenario, BRUCE was commanded to walk with a forward velocity of 0.3 m/s. At t around 2, 4, 6, 8 seconds, a constant external force with a duration of 0.1 s and a magnitude of 12 N (BRUCE has a weight of only 5.2 kg) was exerted on its torso in the left, right, forward, and backward directions, respectively. The detailed simulation results are shown in Figure 4.4. For all the experiments, a preview of three steps is considered in our controller, which is suggested by the numerical simulation results of the reduced-order models in Section 3.5. In the third scenario, we had a similar push recovery comparison on BRUCE and the results agree with each other, as shown in Figure 4.5.

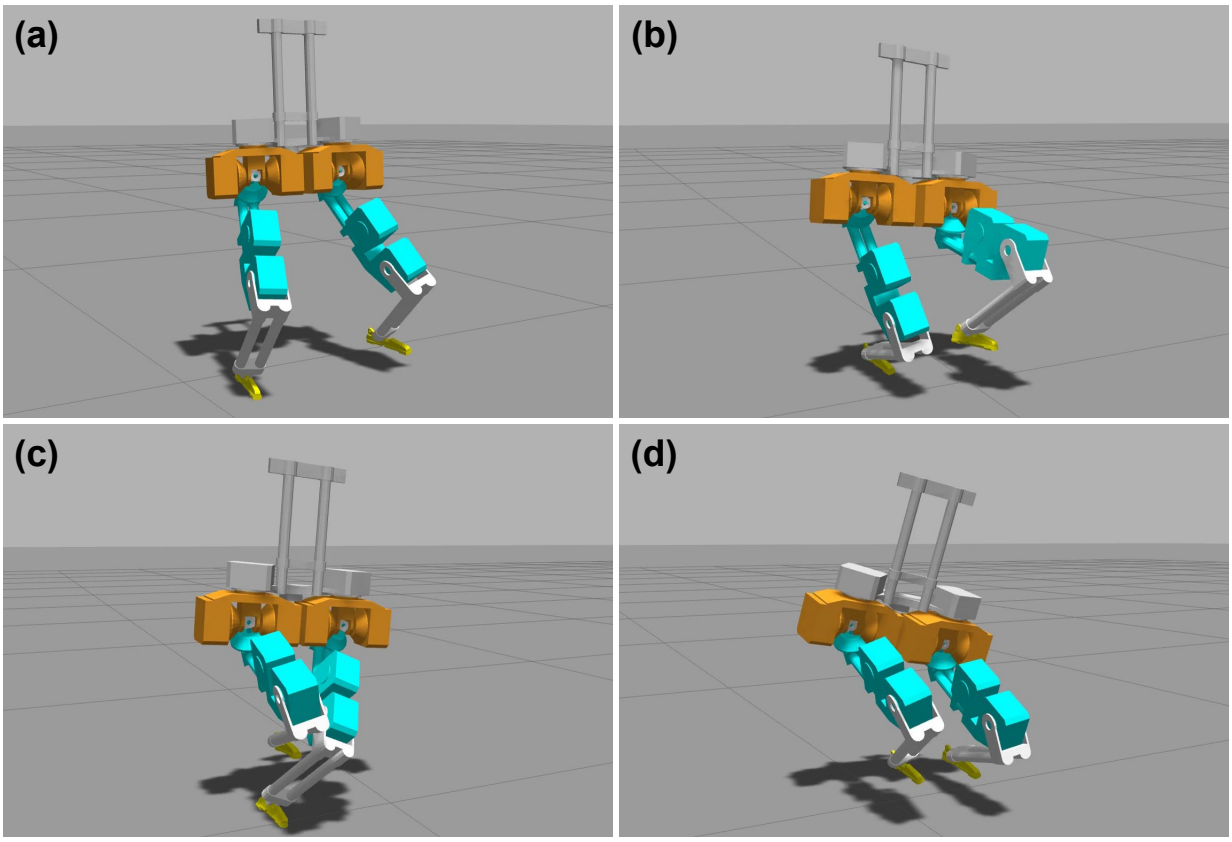


Figure 4.3: Snapshots of BRUCE taking recovery steps after its torso was pushed during (a) walking, (b) running, (c) single-legged hopping, and (d) double-legged hopping.

Uneven Terrain

In the simulation, BRUCE was commanded to walk with a forward velocity of 0.3 m/s and on the ground there were random wood slats with different sizes but a fixed height of 1 cm (BRUCE has a CoM height of only 30 cm), as shown in Figure 4.7. Thanks to the robust footstep planning and compliance of the WBC, BRUCE successfully conquered this uneven terrain.

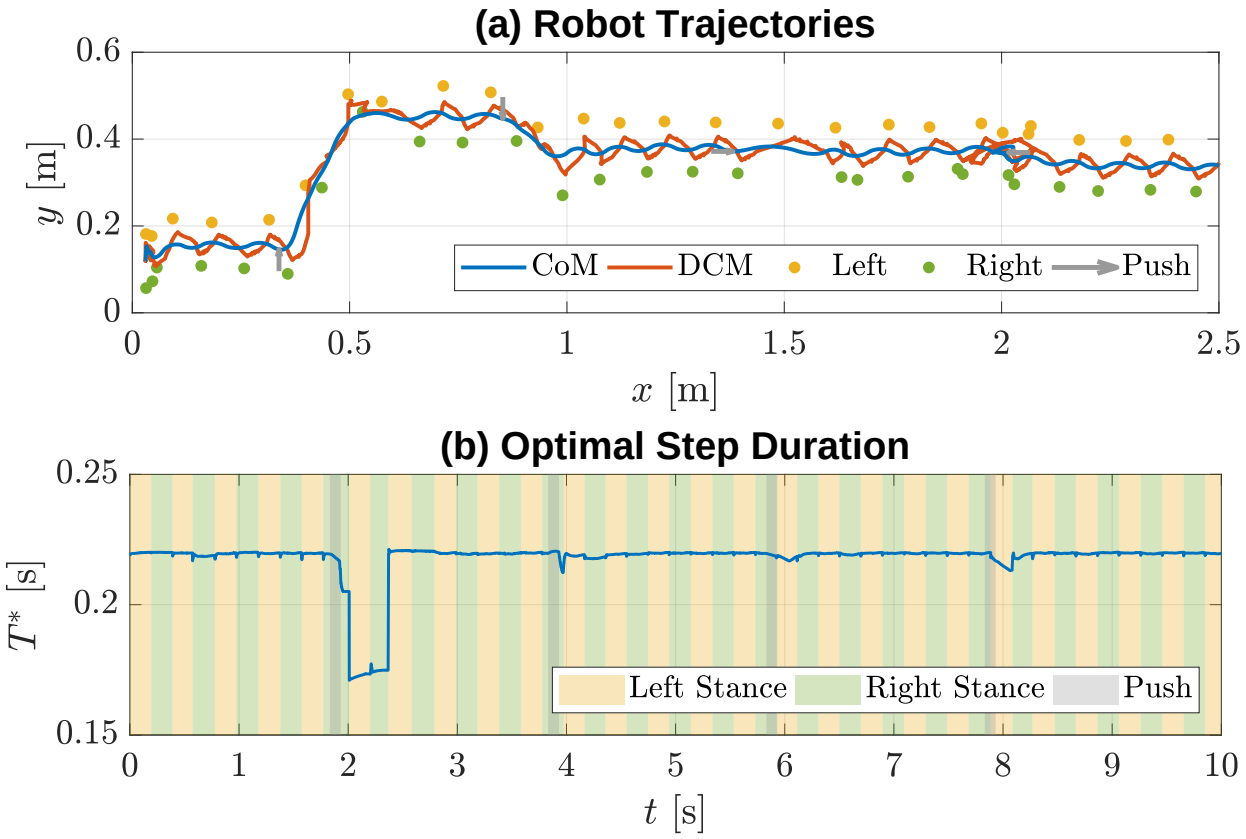


Figure 4.4: Simulation results of push recovery. (a) Position trajectories of robot CoM, DCM, and feet. Note that the robot was locomoting to the positive x direction with a forward velocity command of 0.3 m/s. (b) Time series of the optimal step duration suggested by the footstep planner. The amber and green shaded areas indicate the left- and right-stance phases, respectively. The gray areas indicate the duration of the external pushes.

4.5.2 Hardware Results

We further implemented the proposed locomotion controller on the hardware platform. Note that only the walking gait was realized at this point.

Push Recovery

The push recovery test was also conducted on the real hardware. We pushed BRUCE on different parts, e.g., torso and leg, in various directions, e.g., longitudinal and lateral, at random times with random durations, and BRUCE managed to survive. An example clip is shown in Figure 4.6.

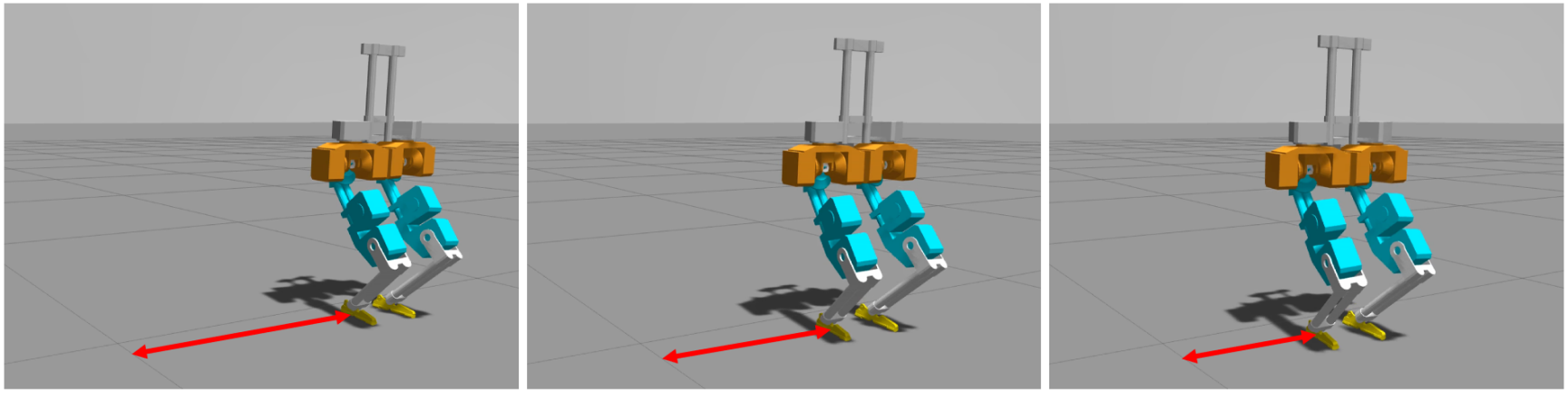


Figure 4.5: Comparison of various numbers of preview steps. The system performs better with more steps planned in advance.

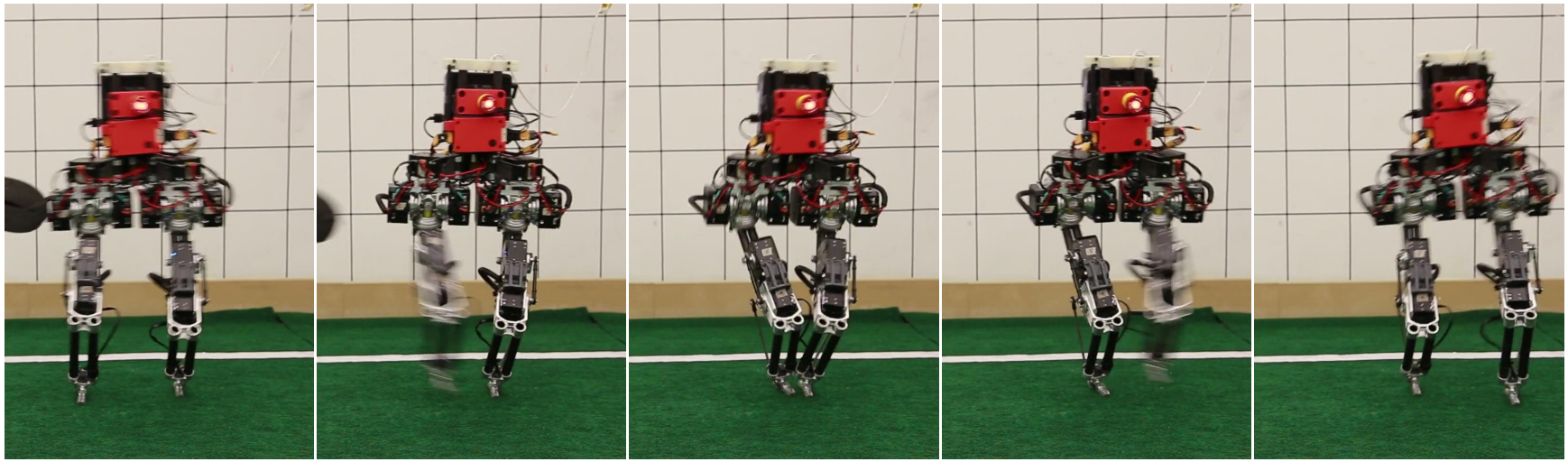


Figure 4.6: Snapshots (from left to right) of BRUCE taking recovery steps after its torso was pushed to the right.

Irregular Terrains

In this experiment, to gauge the overall system robustness in terms of terrain uncertainty, BRUCE was challenged to a series of irregular terrains, as shown in Figure 4.7. Note that BRUCE was walking blindly without any terrain information.

Uneven Terrain In the real world, BRUCE was also able to walk with small ground height variations.

Soft Terrain In this scenario, BRUCE was stepping on a yoga mat in the real world. This kind of soft terrain is challenging as it is difficult for the stance foot to remain stationary due to the surface compliance, which can easily cause oscillations and even instability of the system. During the experiment, the stance leg (in particular the ankle) was constantly adapting in order to keep the balance and our walking controller was able to stabilize the system on this soft terrain.

Sliding Terrain In this scenario, BRUCE was stepping on foam boards which can easily slide on the ground. The sliding can also mess up the state estimation which assumes fixed contact location. However, our walking controller could still stabilize the system with an adapted gait.

4.6 Conclusion

In this chapter, a two-level dynamic bipedal locomotion controller was presented. Specifically, the DCM-based high-level footstep planner is solving a TO with multiple steps being planned in advance, which optimally determines the footstep location and timing. The ID-based low-level WBC is finding the instantaneously required, dynamically consistent joint torques to best realize the task-space behaviors. Both problems can be transcribed into small-scale QPs which can be solved efficiently with guaranteed optimality for real-time

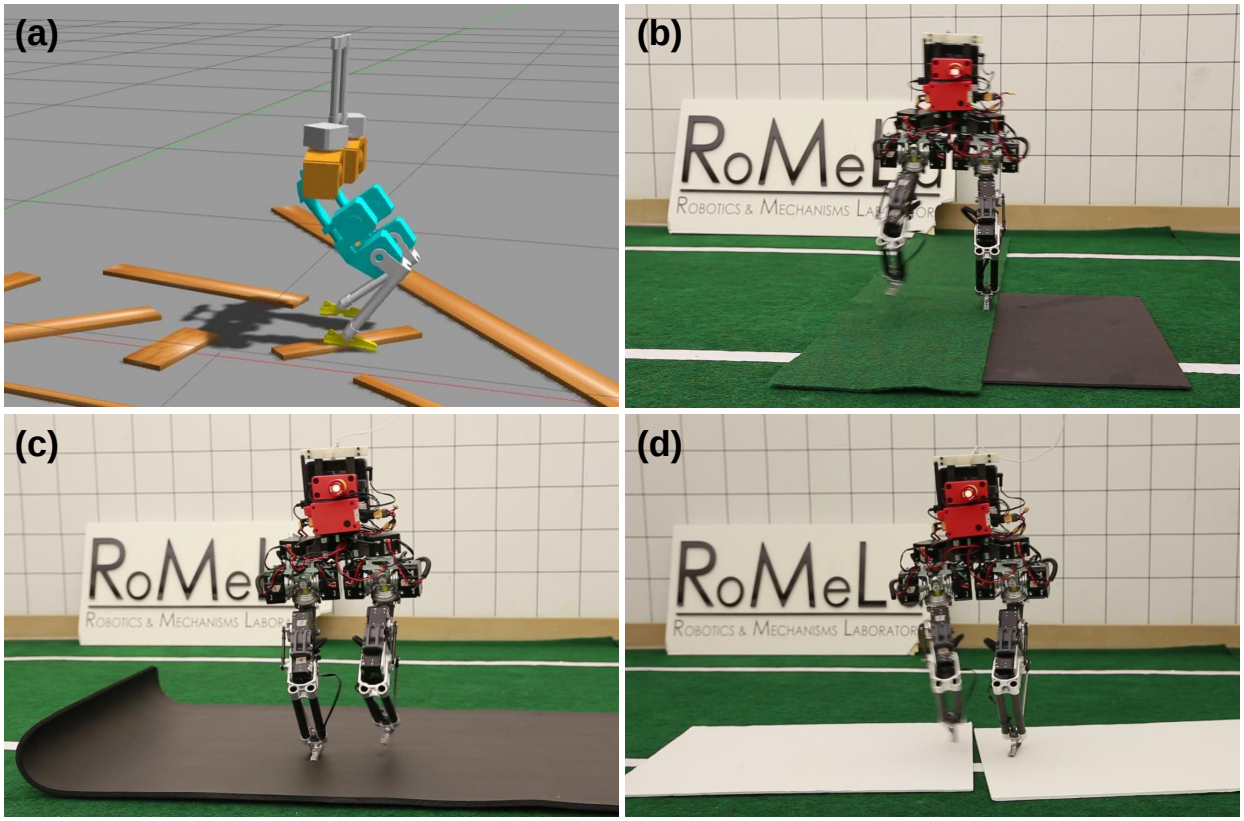


Figure 4.7: Snapshots of BRUCE walking on irregular terrains. (a) Uneven terrain. (b) Height variation. (c) Soft terrain. (d) Sliding terrain.

applications. Additionally, detailed implementation guidance of the control framework is provided on BRUCE, a miniature bipedal robot with proprioceptive actuation. To the best of our knowledge, this is the first fully-untethered miniature bipedal robot which can achieve robust dynamic walking using this framework. The presented practical experience and insightful discussion will benefit the robotics community, especially for new people to dive into the field. Lastly, an extensive series of simulation and hardware locomotion experiments were carried out, including omnidirectional walking, push recovery, and irregular terrains, which demonstrate the strong robustness of the approach in the presence of various disturbances and uncertainties.

Chapter 5

Conclusion

5.1 Summary

Humanoid robots, despite their increased complexity compared with other types of robot, have the potential to exert a much larger influence on human society in the future. What is fascinating about humanoid robots is their inimitable possibility of going anywhere humans can go and doing anything humans can do. It is these abilities that we wish to replicate in machines which can potentially overturn our daily lives. While progress has been made, humanoid robots are only beginning to fulfill this great potential.

This dissertation aims at addressing some of the challenges in the current study of humanoid robots, i.e., limited hardware accessibility and locomotion control. Chapter 1 introduces the necessary ingredients of humanoid locomotion and their technological advancements in history, e.g., humanoid robot deign, sensing technology, actuation scheme, motion planning and control. Chapter 2 details the design and analysis of a miniature bipedal robot called BRUCE – Bipedal Robot Unit with Compliance Enhanced. Preliminary experiments were also conducted to verify the basic functionalities of the robot design as well as to explore its dynamic capabilities. The BRUCE robot is desired to serve as an accessible and reliable humanoid robot platform for general research purposes. Chapter 3 elaborates the online lo-

comotion gait regeneration strategy using the reduced-order models. The locomotion control is solving a small-scale trajectory optimization problem with multiple steps being considered in advance, which optimally plans the step location and timing in real time. Chapter 4 presents other critical components of the locomotion control framework, e.g, software architecture, state estimation, and low-level whole-body control. The effectiveness of the overall approach was finally validated on the BRUCE robot with an extensive series of successful simulation and hardware experiments.

5.2 Future Works

The advancements made in this dissertation can lead to the following potential research directions in the future:

- For the BRUCE robot, an upgraded upper body with arms will be added for more capabilities, e.g., standing up on its own and loco-manipulation. A future concept is shown in Fig. 5.1. The overall system will also benefit from the upper body due to the additional DoFs and more lumped inertia at the hip. In addition, the integration of a liquid cooling system into the actuators will ensure steadier actuation performance with longer endurance. Moreover, we are working on making the BRUCE robot an open-source platform for the robotics community with an affordable cost. We envision it will boost the studies of humanoid robots as an accessible and reliable miniature humanoid robot platform.

- In terms of locomotion studies, realizing the running and hopping gaits with the physical system is also on the future list. The inclusion of the flight phase can bring new challenges to both the hardware platform and control framework, e.g., the physical structure may not be strong enough to handle the landing impact and the state estimation cannot deal with the flight phase with just a simple complementary filter. Meanwhile, with an upgraded upper body, arm-assisted locomotion strategy can be

Figure 5.1: BRUCE future concept (https://www.westwoodrobotics.io/bruce).

investigated for stronger robustness. Other dynamic behaviors including jumping and parkour is under exploration as well.

Bibliography

[1] Xiaomi. https://www.mi.com/cyberone. Accessed: 2022-09-02.

[2] Tesla. https://www.tesla.com/AI. Accessed: 2022-10-13.

[3] Honda. https://asimo.honda.com. Accessed: 2022-09-02.

[4] Agility Robotics. https://www.agilityrobotics.com/robots. Accessed: 2022-03-04.

[5] Boston Dynamics. https://www.bostondynamics.com/atlas. Accessed: 2022-09-02.

[6] Deborah Levine Gera. *Ancient Greek Ideas on Speech, Language, and Civilization*. Oxford University Press, 2003.

[7] Joseph Needham. *Science and Civilisation in China: Volume 2, History of Scientific Thought*. Cambridge University Press, 1991.

[8] Mark E. Rosheim. *Robot Evolution: The Development of Anthrobotics*. John Wiley & Sons, 1994.

[9] Michael E. Moran. The da Vinci robot. *Journal of Endourology*, 20(12):986–990, 2006.

[10] Gaby Wood. *Living Dolls: A Magical History of the Quest for Mechanical Life*. Faber & Faber, 2002.

[11] Jane Marie Law. *Puppets of Nostalgia - The Life, Death and Rebirth of the Japanese Awaji Ningyō Tradition*. Princeton University Press, 1998.

[12] Ashok K. Hemal and Mani Menon. *Robotics in Genitourinary Surgery*. Springer, 2018.

[13] Jessica Riskin. *The Restless Clock: A History of the Centuries-Long Argument Over What Makes Living Things Tick*. University of Chicago Press, 2016.

[14] Scott Schaut. *Robots of Westinghouse: 1924 – Today*. Mansfield Memorial Museum, 2006.

[15] Isaac Asimov. *I, Robot*. Doubleday, 1950.

[16] Shimon Y. Nof. *Handbook of Industrial Robotics*. John Wiley & Sons, 1999.

[17] Ichiro Kato, Sadamu Ohteru, Hiroshi Kobayashi, Katsuhiko Shirai, and Akihiko Uchiyama. Information-power machine with senses and limbs. In *On Theory and Practice of Robots and Manipulators*, pages 11–24. Springer, 1974.

[18] Hun-ok Lim and Atsuo Takanishi. Biped walking robots created at Waseda University: WL and WABIAN family. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1850):49–64, 2007.

[19] Kenji Kaneko, Fumio Kanehiro, Shuuji Kajita, Kazuhiko Yokoyama, Kazuhiko Akachi, Toshikazu Kawasaki, Shigehiko Ota, and Takakatsu Isozumi. Design of prototype humanoid robotics platform for HRP. In *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2431–2436. IEEE, 2002.

[20] Giorgio Metta, Giulio Sandini, David Vernon, Lorenzo Natale, and Francesco Nori. The iCub humanoid robot: an open platform for research in embodied cognition. In *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, pages 50–56, 2008.

[21] David Gouaillier, Vincent Hugel, Pierre Blazevic, Chris Kilner, Jérôme Monceaux, Pascal Lafourcade, Brice Marnier, Julien Serre, and Bruno Maisonnier. Mechatronic

design of NAO humanoid. In *2009 IEEE International Conference on Robotics and Automation*, pages 769–774. IEEE, 2009.

[22] Inyong Ha, Yusuke Tamura, Hajime Asama, Jeakweon Han, and Dennis W. Hong. Development of open humanoid platform DARwIn-OP. In *SICE Annual Conference 2011*, pages 2178–2181. IEEE, 2011.

[23] Olivier Stasse, Thomas Flayols, Rohan Budhiraja, Kevin Giraud-Esclasse, Justin Carpentier, Joseph Mirabel, Andrea Del Prete, Philippe Souères, Nicolas Mansard, Florent Lamiraux, Jean-Paul Laumond, Luca Marchionni, Hilario Tomé, and Francesco Ferro. TALOS: A new humanoid research platform targeted for industrial applications. In *2017 IEEE-RAS 17th International Conference on Humanoid Robots*, pages 689–695. IEEE, 2017.

[24] Grzegorz Ficht and Sven Behnke. Bipedal humanoid hardware design: A technology review. *Current Robotics Reports*, 2(2):201–210, 2021.

[25] Jeffrey Chen Yu. *Control Implementation of Dynamic Locomotion on Compliant, Underactuated, Force-Controlled Legged Robots with Non-Anthropomorphic Design*. University of California, Los Angeles, 2020.

[26] Sebastian Lohmeier, Thomas Buschmann, Markus Schwienbacher, Heinz Ulbrich, and Friedrich Pfeiffer. Leg design for a humanoid walking robot. In *2006 IEEE-RAS 6th International Conference on Humanoid Robots*, pages 536–541. IEEE, 2006.

[27] Jeakweon Han and Dennis Hong. Development of a full-sized bipedal humanoid robot utilizing spring assisted parallel four-bar linkages with synchronized actuation. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 799–806, 2011.

[28] Derek Lahr, Viktor Orekhov, Bryce Lee, and Dennis Hong. Early developments of a parallelly actuated humanoid, SAFFiR. In *International Design Engineering Technical*

Conferences and Computers and Information in Engineering Conference, pages 1–7. American Society of Mechanical Engineers, 2013.

[29] Johannes Englsberger, Alexander Werner, Christian Ott, Bernd Henze, Maximo A. Roa, Gianluca Garofalo, Robert Burger, Alexander Beyer, Oliver Eiberger, Korbinian Schmid, and Alin Albu-Schäffer. Overview of the torque-controlled humanoid robot TORO. In *2014 IEEE-RAS 14th International Conference on Humanoid Robots*, pages 916–923. IEEE, 2014.

[30] Nicolaus A. Radford, Philip Strawser, Kimberly Hambuchen, Joshua S. Mehling, William K. Verdeyen, A. Stuart Donnan, James Holley, Jairo Sanchez, Vienny Nguyen, Lyndon Bridgwater, Reginald Berka, Robert Ambrose, Mason Myles Markee, N. J. Fraser-Chanpong, Christopher McQuin, John D. Yamokoski, Stephen Hart, Raymond Guo, Adam Parsons, Brian Wightman, Paul Dinh, Barrett Ames, Charles Blakely, Courtney Edmondson, Brett Sommers, Rochelle Rea, Chad Tobler, Heather Bibby, Brice Howard, Lei Niu, Andrew Lee, Michael Conover, Lily Truong, Ryan Reed, David Chesney, Robert Platt Jr., Gwendolyn Johnson, Chien-Liang Fok, Nicholas Paine, Luis Sentis, Eric Cousineau, Ryan Sinnet, Jordan Lack, Matthew Powell, Benjamin Morris, Aaron Ames, and Jide Akinyode. Valkyrie: NASA's first bipedal humanoid robot. *Journal of Field Robotics*, 32(3):397–419, 2015.

[31] Christian Hubicki, Jesse Grimes, Mikhail Jones, Daniel Renjewski, Alexander Spröwitz, Andy Abate, and Jonathan Hurst. ATRIAS: Design and validation of a tether-free 3D-capable spring-mass bipedal robot. *The International Journal of Robotics Research*, 35(12):1497–1521, 2016.

[32] Francesca Negrello, Manolo Garabini, Manuel G. Catalano, Przemyslaw Kryczka, Wooseok Choi, Darwin G. Caldwell, Antonio Bicchi, and Nikolaos G. Tsagarakis. WALK-MAN humanoid lower body design optimization for enhanced physical per-

formance. In *2016 IEEE International Conference on Robotics and Automation*, pages 1817–1824. IEEE, 2016.

[33] Marco Ceccarelli, Daniele Cafolla, Matteo Russo, and Giuseppe Carbone. LARM bot humanoid design towards a prototype. *MOJ Applied Bionics and Biomechanics*, 1(2):48–49, 2017.

[34] Yohei Kakiuchi, Masayuki Kamon, Nobuyasu Shimomura, Sou Yukizaki, Noriaki Takasugi, Shunichi Nozawa, Kei Okada, and Masayuki Inaba. Development of life-sized humanoid robot platform with robustness for falling down, long time working and error occurrence. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 689–696. IEEE, 2017.

[35] Grzegorz Ficht, Philipp Allgeuer, Hafez Farazi, and Sven Behnke. NimbRo-OP2: Grown-up 3D printed open humanoid platform for research. In *2017 IEEE-RAS 17th International Conference on Humanoid Robots*, pages 669–675. IEEE, 2017.

[36] Kevin G. Gim, Joohyung Kim, and Katsu Yamane. Design of a serial-parallel hybrid leg for a humanoid robot. In *2018 IEEE International Conference on Robotics and Automation*, pages 6076–6081. IEEE, 2018.

[37] Chengxu Zhou and Nikos Tsagarakis. On the comprehensive kinematics analysis of a humanoid parallel ankle mechanism. *Journal of Mechanisms and Robotics*, 10(5):051015, 2018.

[38] Yeting Liu, Junjie Shen, Jingwen Zhang, Xiaoguang Zhang, Taoyuanmin Zhu, and Dennis Hong. Design and control of a miniature bipedal robot with proprioceptive actuation for dynamic behaviors. In *2022 IEEE International Conference on Robotics and Automation*, pages 8547–8553. IEEE, 2022.

[39] Katayon Radkhah, Christophe Maufroy, Moritz Maus, Dorian Scholz, Andre Seyfarth, and Oskar Von Stryk. Concept and design of the BioBiped1 robot for human-like

walking and running. *International Journal of Humanoid Robotics*, 8(03):439–458, 2011.

[40] Yuto Nakanishi, Yuki Asano, Toyotaka Kozuki, Hironori Mizoguchi, Yotaro Motegi, Masahiko Osada, Takuma Shirai, Junichi Urata, Kei Okada, and Masayuki Inaba. Design concept of detail musculoskeletal humanoid "Kenshiro" – Toward a real human body musculoskeletal simulator. In *2012 IEEE-RAS 12th International Conference on Humanoid Robots*, pages 1–6. IEEE, 2012.

[41] Yuki Asano, Toyotaka Kozuki, Soichi Ookubo, Masaya Kawamura, Shinsuke Nakashima, Takeshi Katayama, Iori Yanokura, Toshinori Hirose, Kento Kawaharazuka, Shogo Makino, Youhei Kakiuchi, Kei Okada, and Masayuki Inaba. Human mimetic musculoskeletal humanoid Kengoro toward real world physically interactive actions. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots*, pages 876–883. IEEE, 2016.

[42] Steffen Schütz, Atabak Nejadfard, Krzysztof Mianowski, Patrick Vonwirth, and Karsten Berns. CARL – a compliant robotic leg featuring mono- and biarticular actuation. In *2017 IEEE-RAS 17th International Conference on Humanoid Robots*, pages 289–296. IEEE, 2017.

[43] Wesley Roozing, Zeyu Ren, and Nikos G. Tsagarakis. An efficient leg with series-parallel and biarticular compliant actuation: design optimization, modeling, and control of the eLeg. *The International Journal of Robotics Research*, 40(1):37–54, 2021.

[44] Jeffrey Yu, Joshua Hooks, Sepehr Ghassemi, Alexandra Pogue, and Dennis Hong. Investigation of a non-anthropomorphic bipedal robot with stability, agility, and simplicity. In *2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence*, pages 11–15. IEEE, 2016.

[45] Jeffrey Yu, Joshua Hooks, Xiaoguang Zhang, Min Sung Ahn, and Dennis Hong. A proprioceptive, force-controlled, non-anthropomorphic biped for dynamic locomotion. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots*, pages 489–496. IEEE, 2018.

[46] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT press, 2011.

[47] Philipp Mittendorfer and Gordon Cheng. Humanoid multimodal tactile-sensing modules. *IEEE Transactions on Robotics*, 27(3):401–410, 2011.

[48] Liang Zou, Chang Ge, Z. Jane Wang, Edmond Cretu, and Xiaoou Li. Novel tactile sensor technology and smart tactile sensing systems: A review. *Sensors*, 17(11):2653, 2017.

[49] Nicholas Kellaris, Vidyacharan Gopaluni Venkata, Garrett M. Smith, Shane K. Mitchell, and Christoph Keplinger. Peano-HASEL actuators: Muscle-mimetic, electrohydraulic transducers that linearly contract on activation. *Science Robotics*, 3(14):eaar3276, 2018.

[50] Gavin Kenneally, Avik De, and Daniel E. Koditschek. Design principles for a family of direct-drive legged robots. *IEEE Robotics and Automation Letters*, 1(2):900–907, 2016.

[51] Gill A. Pratt and Matthew M. Williamson. Series elastic actuators. In *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 399–406. IEEE, 1995.

[52] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. ANYmal

– a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 38–44. IEEE, 2016.

[53] Sangok Seok, Albert Wang, David Otten, and Sangbae Kim. Actuator design for high force proprioceptive control in fast legged locomotion. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1970–1975. IEEE, 2012.

[54] Patrick M. Wensing, Albert Wang, Sangok Seok, David Otten, Jeffrey Lang, and Sangbae Kim. Proprioceptive actuator design in the MIT cheetah: Impact mitigation and high-bandwidth physical interaction for dynamic legged robots. *IEEE Transactions on Robotics*, 33(3):509–522, 2017.

[55] Taoyuanmin Zhu, Joshua Hooks, and Dennis Hong. Design, modeling, and analysis of a liquid cooled proprioceptive actuator for legged robots. In *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 36–43. IEEE, 2019.

[56] Taoyuanmin Zhu, Min Sung Ahn, and Dennis Hong. Design and experimental study of BLDC motor immersion cooling for legged robots. In *2021 20th International Conference on Advanced Robotics*, pages 1137–1143. IEEE, 2021.

[57] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, and Rob Playter. BigDog, the rough-terrain quadruped robot. *IFAC Proceedings Volumes*, 41(2):10822–10825, 2008.

[58] Hiroshi Kaminaga, Tianyi Ko, Ryo Masumura, Mitsuo Komagata, Shunsuke Sato, Satoshi Yorita, and Yoshihiko Nakamura. Mechanism and control of whole-body electro-hydrostatic actuator driven humanoid robot Hydra. In *International Symposium on Experimental Robotics*, pages 656–665. Springer, 2016.

[59] Sang-Ho Hyon, Daisuke Suewaka, Yuki Torii, and Narifumi Oku. Design and experimental evaluation of a fast torque-controlled hydraulic humanoid robot. *IEEE/ASME Transactions on Mechatronics*, 22(2):623–634, 2016.

[60] Frank Daerden and Dirk Lefeber. Pneumatic artificial muscles: actuators for robotics and automation. *European Journal of Mechanical and Environmental Engineering*, 47(1):11–21, 2002.

[61] Olivier Stasse and Thomas Flayols. An overview of humanoid robots technologies. *Biomechanics of Anthropomorphic Systems*, pages 281–310, 2019.

[62] Jimmy Vermeulen, Björn Verrelst, Bram Vanderborght, Dirk Lefeber, and Patrick Guillaume. Trajectory planning for the walking biped "Lucy". *The International Journal of Robotics Research*, 25(9):867–887, 2006.

[63] Ryuma Niiyama, Akihiko Nagakubo, and Yasuo Kuniyoshi. Mowgli: A bipedal jumping and landing robot with an artificial musculoskeletal system. In *2007 IEEE International Conference on Robotics and Automation*, pages 2546–2551. IEEE, 2007.

[64] Jenna Reher and Aaron D. Ames. Dynamic walking: Toward agile and efficient bipedal robots. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:535–572, 2021.

[65] Roy Featherstone. *Rigid Body Dynamics Algorithms*. Springer, 2008.

[66] Jerry Pratt, Chee-Meng Chew, Ann Torres, Peter Dilworth, and Gill Pratt. Virtual model control: An intuitive approach for bipedal locomotion. *The International Journal of Robotics Research*, 20(2):129–143, 2001.

[67] Hugh Herr and Marko Popovic. Angular momentum in human walking. *The Journal of Experimental Biology*, 211(4):467–481, 2008.

[68] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3D linear inverted pendulum mode: A simple modeling for a biped walking pattern generation. In *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 239–246. IEEE, 2001.

[69] 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*, pages 1620–1626. IEEE, 2003.

[70] Jerry E. Pratt and Sergey V. Drakunov. Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model. In *2007 IEEE International Conference on Robotics and Automation*, pages 4653–4660. IEEE, 2007.

[71] Michael A. Hopkins, Dennis W. Hong, and Alexander Leonessa. Humanoid locomotion on uneven terrain using the time-varying divergent component of motion. In *2014 IEEE-RAS 14th International Conference on Humanoid Robots*, pages 266–272. IEEE, 2014.

[72] Jong H. Park and Kyong D. Kim. Biped robot walking using gravity-compensated inverted pendulum mode and computed torque control. In *1998 IEEE International Conference on Robotics and Automation*, pages 3528–3533. IEEE, 1998.

[73] Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. Capture point: A step toward humanoid push recovery. In *2006 IEEE-RAS 6th International Conference on Humanoid Robots*, pages 200–207. IEEE, 2006.

[74] Miomir Vukobratović and Branislav Borovac. Zero-moment point – thirty five years of its life. *International Journal of Humanoid Robotics*, 1(1):157–173, 2004.

[75] 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.

[76] A.L. Hof, M.G.J. Gazendam, and W.E. Sinke. The condition for dynamic stability. *Journal of Biomechanics*, 38(1):1–8, 2005.

[77] Twan Koolen, Tomas de Boer, John Rebula, Ambarish Goswami, and Jerry Pratt. Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models. *The International Journal of Robotics Research*, 31(9):1094–1113, 2012.

[78] Toru Takenaka, Takashi Matsumoto, and Takahide Yoshiike. Real time motion generation and control for biped robot – 1 st report: Walking gait pattern generation. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1084–1091. IEEE, 2009.

[79] Reinhard Blickhan. The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11):1217–1227, 1989.

[80] Hartmut Geyer and Uluc Saranli. Gait based on the spring-loaded inverted pendulum. In *Humanoid Robotics: A Reference*, pages 1–25. Springer, 2017.

[81] Marc H. Raibert. *Legged Robots That Balance*. MIT press, 1986.

[82] Tad McGeer. Passive bipedal running. *Proceedings of the Royal Society of London. B. Biological Sciences*, 240(1297):107–134, 1990.

[83] Ben Brown and Garth Zeglin. The bow leg hopping robot. In *1998 IEEE International Conference on Robotics and Automation*, pages 781–786. IEEE, 1998.

[84] Sang-Ho Hyon and Tsutomu Mita. Development of a biologically inspired hopping robot – "Kenken". In *2002 IEEE International Conference on Robotics and Automation*, pages 3984–3991. IEEE, 2002.

[85] André Seyfarth, Hartmut Geyer, and Hugh Herr. Swing-leg retraction: a simple control model for stable running. *Journal of Experimental Biology*, 206(15):2547–2555, 2003.

[86] Hae-Won Park, Koushil Sreenath, Jonathan W. Hurst, and Jessy W. Grizzle. Identification of a bipedal robot with a compliant drivetrain. *IEEE Control Systems Magazine*, 31(2):63–88, 2011.

[87] Gerardo Bledt, Patrick M. Wensing, and Sangbae Kim. Policy-regularized model predictive control to stabilize diverse quadrupedal gaits for the MIT cheetah. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4102–4109. IEEE, 2017.

[88] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1–9. IEEE, 2018.

[89] Alexander W. Winkler, Carmine D. Bellicoso, Marco Hutter, and Jonas Buchli. Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. *IEEE Robotics and Automation Letters*, 3(3):1560–1567, 2018.

[90] Octavio Villarreal, Victor Barasuol, Patrick M. Wensing, Darwin G. Caldwell, and Claudio Semini. MPC-based controller with terrain insight for dynamic legged locomotion. In *2020 IEEE International Conference on Robotics and Automation*, pages 2436–2442. IEEE, 2020.

[91] Seungwoo Hong, Joon-Ha Kim, and Hae-Won Park. Real-time constrained nonlinear model predictive control on SO(3) for dynamic legged locomotion. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3982–3989. IEEE, 2020.

[92] Yanran Ding, Abhishek Pandala, Chuanzheng Li, Young-Ha Shin, and Hae-Won Park. Representation-free model predictive control for dynamic motions in quadrupeds. *IEEE Transactions on Robotics*, 37(4):1154–1171, 2021.

[93] Junjie Shen and Dennis Hong. Convex model predictive control of single rigid body model on SO(3) for versatile dynamic legged motions. In *2022 International Conference on Robotics and Automation*, pages 6586–6592. IEEE, 2022.

[94] Gabriel García, Robert Griffin, and Jerry Pratt. MPC-based locomotion control of bipedal robots with line-feet contact using centroidal dynamics. In *2020 IEEE-RAS 20th International Conference on Humanoid Robots*, pages 276–282. IEEE, 2021.

[95] Alberto Isidori. *Nonlinear Control Systems: An Introduction*. Springer, 1985.

[96] Jessy W. Grizzle, Gabriel Abba, and Franck Plestan. Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. *IEEE Transactions on Automatic Control*, 46(1):51–64, 2001.

[97] C.K. Chow and D.H. Jacobson. Studies of human locomotion via optimal programming. *Mathematical Biosciences*, 10(3-4):239–306, 1971.

[98] P.H. Channon, S.H. Hopkins, and D.T. Pham. Derivation of optimal walking motions for a bipedal walking robot. *Robotica*, 10(2):165–172, 1992.

[99] Mostafa Rostami and Guy Bessonnet. Impactless sagittal gait of a biped robot during the single support phase. In *1998 IEEE International Conference on Robotics and Automation*, pages 1385–1391. IEEE, 1998.

[100] Christine Chevallereau, Alexander Formal'sky, and Bernard Perrin. Low energy cost reference trajectories for a biped robot. In *1998 IEEE International Conference on Robotics and Automation*, pages 1398–1404. IEEE, 1998.

[101] L. Roussel, C. Canudas-de-Wit, and A. Goswami. Generation of energy optimal complete gait cycles for biped robots. In *1998 IEEE International Conference on Robotics and Automation*, pages 2036–2041. IEEE, 1998.

[102] Joachim Denk and Gunther Schmidt. Synthesis of a walking primitive database for a humanoid robot using optimal control techniques. In *2001 IEEE-RAS 1st International Conference on Humanoid Robots*, pages 319–326, 2001.

[103] Thomas Buschmann, Sebastian Lohmeier, Heinz Ulbrich, and Friedrich Pfeiffer. Optimization based gait pattern generation for a biped robot. In *2005 IEEE-RAS 5th International Conference on Humanoid Robots*, pages 98–103. IEEE, 2005.

[104] Katja Mombaur. Using optimization to create self-stable human-like running. *Robotica*, 27(3):321–330, 2009.

[105] Jonas Koenemann, Andrea Del Prete, Yuval Tassa, Emanuel Todorov, Olivier Stasse, Maren Bennewitz, and Nicolas Mansard. Whole-body model-predictive control applied to the HRP-2 humanoid. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3346–3351. IEEE, 2015.

[106] Michael Neunert, Markus Stäuble, Markus Giftthaler, Carmine D. Bellicoso, Jan Carius, Christian Gehring, Marco Hutter, and Jonas Buchli. Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robotics and Automation Letters*, 3(3):1458–1465, 2018.

[107] Jean-Paul Laumond, Nicolas Mansard, and Jean-Bernard Lasserre. Optimality in robot motion: optimal versus optimized motion. *Communications of the ACM*, 57(9):82–89, 2014.

[108] Joachim Denk and Günther Schmidt. Synthesis of walking primitive databases for biped robots in 3D-environments. In *2003 IEEE International Conference on Robotics and Automation*, pages 1343–1349. IEEE, 2003.

[109] Pierre-Brice Wieber and Christine Chevallereau. Online adaptation of reference trajectories for the control of walking systems. *Robotics and Autonomous Systems*, 54(7):559–566, 2006.

[110] Xingye Da, Omar Harib, Ross Hartley, Brent Griffin, and Jessy W. Grizzle. From 2D design of underactuated bipedal gaits to 3D implementation: Walking with speed tracking. *IEEE Access*, 4:3469–3478, 2016.

[111] Yukai Gong, Ross Hartley, Xingye Da, Ayonga Hereid, Omar Harib, Jiunn-Kai Huang, and Jessy Grizzle. Feedback control of a Cassie bipedal robot: Walking, standing, and riding a segway. In *2019 American Control Conference*, pages 4559–4566. IEEE, 2019.

[112] Pierre-Brice Wieber, Russ Tedrake, and Scott Kuindersma. Modeling and control of legged robots. In *Springer Handbook of Robotics*, pages 1203–1234. Springer, 2016.

[113] Pierre-Brice Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *2006 IEEE-RAS 6th International Conference on Humanoid Robots*, pages 137–142. IEEE, 2006.

[114] Holger Diedam, Dimitar Dimitrov, Pierre-Brice Wieber, Katja Mombaur, and Moritz Diehl. Online walking gait generation with adaptive foot positioning through linear model predictive control. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1121–1126. IEEE, 2008.

[115] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur, and Moritz Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010.

[116] Benjamin J. Stephens and Christopher G. Atkeson. Push recovery by stepping for humanoid robots with force controlled joints. In *2010 IEEE-RAS 10th International Conference on Humanoid Robots*, pages 52–59. IEEE, 2010.

[117] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Arai, Noriho Koyachi, and Kazuo Tanie. Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, 17(3):280–289, 2001.

[118] Joao Ramos, Benjamin Katz, Meng Yee Michael Chuah, and Sangbae Kim. Facilitating model-based control through software-hardware co-design. In *2018 IEEE International Conference on Robotics and Automation*, pages 566–572. IEEE, 2018.

[119] Majid Khadiv, Alexander Herzog, S. Ali. A. Moosavian, and Ludovic Righetti. Step timing adjustment: A step toward generating robust gaits. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots*, pages 35–42. IEEE, 2016.

[120] Donghyun Kim, Steven Jens Jorgensen, Jaemin Lee, Junhyeok Ahn, Jianwen Luo, and Luis Sentis. Dynamic locomotion for passive-ankle biped robots and humanoids using whole-body locomotion control. *The International Journal of Robotics Research*, 39(8):936–956, 2020.

[121] Salman Faraji, Soha Pouya, Christopher G. Atkeson, and Auke Jan Ijspeert. Versatile and robust 3D walking with a simulated humanoid robot (Atlas): A model predictive control approach. In *2014 IEEE International Conference on Robotics and Automation*, pages 1943–1950. IEEE, 2014.

[122] Siyuan Feng, X. Xinjilefu, Christopher G. Atkeson, and Joohyung Kim. Robust dynamic walking using online foot step optimization. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5373–5378. IEEE, 2016.

[123] Igor Mordatch, Martin de Lasa, and Aaron Hertzmann. Robust physics-based locomotion using low-dimensional planning. *ACM Transactions on Graphics*, 29(3), 2010.

[124] Zohaib Aftab, Thomas Robert, and Pierre-Brice Wieber. Ankle, hip and stepping strategies for humanoid balance recovery with a single model predictive control scheme. In *2012 IEEE-RAS 12th International Conference on Humanoid Robots*, pages 159–164. IEEE, 2012.

[125] Siyuan Feng. *Online Hierarchical Optimization for Humanoid Control*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2016.

[126] Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *IEEE Transactions on Robotics*, 31(2):355–368, 2015.

[127] Robert J. Griffin, Georg Wiedebach, Sylvain Bertrand, Alexander Leonessa, and Jerry Pratt. Walking stabilization using step timing and location adjustment on the humanoid robot, Atlas. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 667–673. IEEE, 2017.

[128] Milad Shafiee, Giulio Romualdi, Stefano Dafarra, Francisco Javier Andrade Chavez, and Daniele Pucci. Online DCM trajectory generation for push recovery of torque-controlled humanoid robots. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots*, pages 671–678. IEEE, 2019.

[129] Hyobin Jeong, Inho Lee, Jaesung Oh, Kang Kyu Lee, and Jun-Ho Oh. A robust walking controller based on online optimization of ankle, hip, and stepping strategies. *IEEE Transactions on Robotics*, 35(6):1367–1386, 2019.

[130] Majid Khadiv, Alexander Herzog, S. Ali. A. Moosavian, and Ludovic Righetti. Walking control based on step timing adaptation. *IEEE Transactions on Robotics*, 36(3):629–643, 2020.

[131] Elham Daneshmand, Majid Khadiv, Felix Grimminger, and Ludovic Righetti. Variable horizon MPC with swing foot dynamics for bipedal walking control. *IEEE Robotics and Automation Letters*, 6(2):2349–2356, 2021.

[132] Mahrokh Ghoddousi Boroujeni, Elham Daneshman, Ludovic Righetti, and Majid Khadiv. A unified framework for walking and running of bipedal robots. In *2021 20th International Conference on Advanced Robotics*, pages 396–403. IEEE, 2021.

[133] Qiang Huang, Shuuji Kajita, Noriho Koyachi, Kenji Kaneko, Kazuhito Yokoi, Hirohiko Arai, Kiyoshi Komoriya, and Kazuo Tanie. A high stability, smooth walking pattern for

a biped robot. In *1999 IEEE International Conference on Robotics and Automation*, pages 65–71. IEEE, 1999.

[134] Joshua Rosenberg Hooks. *Real-Time Optimization for Control of a Multi-Modal Legged Robotic System*. University of California, Los Angeles, 2019.

[135] He Li, Robert J. Frei, and Patrick M. Wensing. Model hierarchy predictive control of robotic systems. *IEEE Robotics and Automation Letters*, 6(2):3373–3380, 2021.

[136] Charles Khazoom and Sangbae Kim. Humanoid arm motion planning for improved disturbance recovery using model hierarchy predictive control. In *2022 IEEE International Conference on Robotics and Automation*, pages 6607–6613. IEEE, 2022.

[137] Eric R. Westervelt, Gabriel Buche, and Jessy W. Grizzle. Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds. *The International Journal of Robotics Research*, 23(6):559–582, 2004.

[138] Jessy W. Grizzle, Jonathan Hurst, Benjamin Morris, Hae-Won Park, and Koushil Sreenath. MABEL, a new robotic bipedal walker and runner. In *2009 American Control Conference*, pages 2030–2036. IEEE, 2009.

[139] Jacob Reher, Wen-Loong Ma, and Aaron D. Ames. Dynamic walking with compliance on a Cassie bipedal robot. In *2019 18th European Control Conference*, pages 2589–2595. IEEE, 2019.

[140] Jacob P. Reher, Ayonga Hereid, Shishir Kolathaya, Christian M. Hubicki, and Aaron D. Ames. Algorithmic foundations of realizing multi-contact locomotion on the humanoid robot DURUS. In *Algorithmic Foundations of Robotics XII*, pages 400–415. Springer, 2020.

[141] Wen-Loong Ma, Shishir Kolathaya, Eric R. Ambrose, Christian M. Hubicki, and Aaron D. Ames. Bipedal robotic running with DURUS-2D: Bridging the gap be-

tween theory and experiment. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 265–274, 2017.

[142] Luis Sentis and Oussama Khatib. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics*, 2(04):505–518, 2005.

[143] Ludovic Righetti, Jonas Buchli, Michael Mistry, Mrinal Kalakrishnan, and Stefan Schaal. Optimal distribution of contact forces with inverse-dynamics control. *The International Journal of Robotics Research*, 32(3):280–298, 2013.

[144] Michael Mistry, Jun Nakanishi, Gordon Cheng, and Stefan Schaal. Inverse kinematics with floating base and constraints for full body humanoid robot control. In *2008 IEEE-RAS 8th International Conference on Humanoid Robots*, pages 22–27. IEEE, 2008.

[145] Oscar E. Ramos. A kinematic whole-body control system for highly redundant robots. In *2016 IEEE ANDESCON*, pages 1–4. IEEE, 2016.

[146] Siyuan Feng, Eric Whitman, X. Xinjilefu, and Christopher G. Atkeson. Optimization based full body control for the Atlas robot. In *2014 IEEE-RAS 14th International Conference on Humanoid Robots*, pages 120–127. IEEE, 2014.

[147] Michael Mistry, Jonas Buchli, and Stefan Schaal. Inverse dynamics control of floating base systems using orthogonal decomposition. In *2010 IEEE International Conference on Robotics and Automation*, pages 3406–3412. IEEE, 2010.

[148] Ludovic Righetti, Jonas Buchli, Michael Mistry, and Stefan Schaal. Control of legged robots with optimal distribution of contact forces. In *2011 IEEE-RAS 11th International Conference on Humanoid Robots*, pages 318–324. IEEE, 2011.

[149] Jaeheung Park and Oussama Khatib. Contact consistent control framework for humanoid robots. In *2006 IEEE International Conference on Robotics and Automation*, pages 1963–1969. IEEE, 2006.

[150] Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008.

[151] Luis Sentis, Jaeheung Park, and Oussama Khatib. Compliant control of multicontact and center-of-mass behaviors in humanoid robots. *IEEE Transactions on Robotics*, 26(3):483–501, 2010.

[152] Siyuan Feng, X. Xinjilefu, Weiwei Huang, and Christopher G. Atkeson. 3D walking based on online optimization. In *2013 IEEE-RAS 13th International Conference on Humanoid Robots*, pages 21–27. IEEE, 2013.

[153] Twan Koolen, Sylvain Bertrand, Gray Thomas, Tomas de Boer, Tingfan Wu, Jesper Smith, Johannes Englsberger, and Jerry Pratt. Design of a momentum-based control framework and application to the humanoid robot Atlas. *International Journal of Humanoid Robotics*, 13(01):1650007, 2016.

[154] Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. *ACM Transactions on Graphics*, 29(4):1–10, 2010.

[155] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, 33(7):1006–1028, 2014.

[156] Alexander Herzog, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti. Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid. *Autonomous Robots*, 40(3):473–491, 2016.

[157] Benjamin Katz, Jared Di Carlo, and Sangbae Kim. Mini Cheetah: A platform for pushing the limits of dynamic quadruped control. In *2019 IEEE International Conference on Robotics and Automation*, pages 6295–6301. IEEE, 2019.

[158] Xiaoguang Zhang. *Application of Proprioception Quasi-Direct Drive Actuators on Dynamic Robotic Systems*. University of California, Los Angeles, 2019.

[159] Alvin R. Tilley and Henry Dreyfuss Associates. *The Measure of Man and Woman: Human Factors in Design*. Wiley, 2001.

[160] Westwood Robotics. https://www.westwoodrobotics.io. Accessed: 2022-11-21.

[161] Yong-Jae Kim. Anthropomorphic low-inertia high-stiffness manipulator for high-speed safe interaction. *IEEE Transactions on Robotics*, 33(6):1358–1374, 2017.

[162] Kunio Kojima, Yuta Kojio, Tatsuya Ishikawa, Fumihito Sugai, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. A robot design method for weight saving aimed at dynamic motions: Design of humanoid JAXON3-P and realization of jump motions. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots*, pages 586–593. IEEE, 2019.

[163] J. Kenneth Salisbury Jr. and William T. Townsend. Compact cable transmission with cable differential, 1993. US Patent 5,207,114.

[164] Francesco Nori, Silvio Traversaro, Jorhabib Eljaik, Francesco Romano, Andrea Del Prete, and Daniele Pucci. iCub whole-body control through force regulation on rigid non-coplanar contacts. *Frontiers in Robotics and AI*, 2:6, 2015.

[165] Jochen Heinzmann and Alexander Zelinsky. The safe control of human-friendly robots. In *1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1020–1025. IEEE, 1999.

[166] Benjamin G. Katz. *A Low Cost Modular Actuator for Dynamic Robots*. PhD thesis, Massachusetts Institute of Technology, 2018.

[167] Stephen Wolfram. *The Mathematica Book*. Wolfram Media, 2003.

[168] Zhifei Zhang, Alain Sarlette, and Zhihao Ling. Integral control on lie groups. *Systems & Control Letters*, 80:9–15, 2015.

[169] Patrick M. Wensing and David E. Orin. Improved computation of the humanoid centroidal dynamics and application for whole-body control. *International Journal of Humanoid Robotics*, 13(01):1550039, 2016.

[170] Christopher G. Atkeson, Chae H. An, and John M. Hollerbach. Estimation of inertial parameters of manipulator loads and links. *The International Journal of Robotics Research*, 5(3):101–119, 1986.

[171] Wisama Khalil and Etienne Dombre. *Modeling Identification and Control of Robots*. CRC Press, 2002.

[172] Kyung-Jo Park. Fourier-based optimal excitation trajectories for the dynamic identification of robots. *Robotica*, 24(5):625–633, 2006.

[173] Taeyoon Lee, Patrick M. Wensing, and Frank C. Park. Geometric robot dynamic identification: A convex programming approach. *IEEE Transactions on Robotics*, 36(2):348–365, 2019.

[174] Shuuji Kajita, Mitsuharu Morisawa, Kanako Miura, Shin'ichiro Nakaoka, Kensuke Harada, Kenji Kaneko, Fumio Kanehiro, and Kazuhito Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496. IEEE, 2010.

[175] Georg Wiedebach, Sylvain Bertrand, Tingfan Wu, Luca Fiorio, Stephen McCrory, Robert Griffin, Francesco Nori, and Jerry Pratt. Walking on partial footholds including

line contacts with the humanoid robot Atlas. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots*, pages 1312–1319. IEEE, 2016.

[176] Cynthia R. Lee and Claire T. Farley. Determinants of the center of mass trajectory in human walking and running. *The Journal of Experimental Biology*, 201(21):2935–2944, 1998.

[177] Petr Zaytsev, S. Javad Hasaneini, and Andy Ruina. Two steps is enough: No need to plan far ahead for walking balance. In *2015 IEEE International Conference on Robotics and Automation*, pages 6295–6300. IEEE, 2015.

[178] Taylor Apgar, Patrick Clary, Kevin Green, Alan Fern, and Jonathan Hurst. Fast online trajectory optimization for the bipedal robot Cassie. In *Proceedings of Robotics: Science and Systems*, 2018.

[179] Pierre-Brice Wieber. Viability and predictive control for safe locomotion. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1103–1108. IEEE, 2008.

[180] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[181] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A LLVM-based Python JIT compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 1–6, 2015.

[182] Michael Bloesch, Marco Hutter, Mark A. Hoepflinger, Stefan Leutenegger, Christian Gehring, C. David Remy, and Roland Siegwart. State estimation for legged robots – consistent fusion of leg kinematics and IMU. *Robotics*, 17:17–24, 2013.

[183] Michael Bloesch, Christian Gehring, Péter Fankhauser, Marco Hutter, Mark A. Hoepflinger, and Roland Siegwart. State estimation for legged robots on unstable and slippery terrain. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6058–6064. IEEE, 2013.

[184] Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control, 2019.

[185] Jaemin Lee, Nicolas Mansard, and Jaeheung Park. Intermediate desired value approach for task transition of robots in kinematic control. *IEEE Transactions on Robotics*, 28(6):1260–1277, 2012.

[186] Nathan Koenig and Andrew Howard. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2149–2154. IEEE, 2004.

[187] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005.

[188] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[189] Michael Grant and Stephen Boyd. CVX: MATLAB software for disciplined convex programming, version 2.1, 2014.

[190] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[191] Jason L. Speyer and David H. Jacobson. *Primer on Optimal Control Theory*. SIAM, 2010.

[192] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[193] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*. https://underactuated.mit.edu. Accessed: 2022-08-26.

[194] Maher Moakher. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 24(1):1–16, 2002.

[195] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2016.

[196] John Stuelpnagel. On the parametrization of the three-dimensional rotation group. *SIAM Review*, 6(4):422–430, 1964.

[197] Sanjay P. Bhat and Dennis S. Bernstein. A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon. *Systems & Control Letters*, 39(1):63–70, 2000.

[198] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC press, 2017.

[199] Joan Solà. Quaternion kinematics for the error-state Kalman filter, 2017.

Appendix A

Mathematical Optimization

The control of humanoid robot system is difficult as the problem is usually too complicated and has no closed-form solution. While classic control theories are no more effective, the reliance on numerical approaches has increased among which mathematical optimization has become a popular tool in robotics, e.g., reactive controller and motion planner, thanks to the rapid development in computer hardware and solving algorithms. A general optimization problem is minimizing (or maximizing) some cost while satisfying the constraints:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & J(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{f}(\boldsymbol{x}) \leq \mathbf{0}, \\ & \boldsymbol{g}(\boldsymbol{x}) = \mathbf{0}, \end{aligned} \tag{A.1}$$

where $\boldsymbol{x}$ is the set of decision variables, $J(\boldsymbol{x})$ is the cost function, $\boldsymbol{f}(\boldsymbol{x})$ is the set of inequality constraints, and $\boldsymbol{g}(\boldsymbol{x})$ is the set of equality constraints.

Nonlinear Program

If at least one of J, $\boldsymbol{f}$, and $\boldsymbol{g}$ is nonlinear but not known to be convex, (A.1) is called a nonlinear program (NLP). NLP is the most general form of optimization and can be "solved"

using commercial solvers such as SNOPT [187] or IPOPT [188]. However, for complicated problems with high nonlinearity NLP can be computationally expensive, suffer from initial guess and local minima issues, and sometimes even end up being intractable. Therefore, it is imperative to make sure the NLP problem is conditioned appropriately to be used on the robotic system.

Convex Program

If the cost function is a convex function and the feasible set is a convex set, i.e., J is convex, $\boldsymbol{f}$ is convex, and $\boldsymbol{g}$ is affine:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & J(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{f}(\boldsymbol{x}) \leq \boldsymbol{0}, \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \end{aligned} \tag{A.2}$$

where $\boldsymbol{A}$ is a constant matrix and $\boldsymbol{b}$ is a constant vector, (A.1) is reduced to a convex program (CP). A CP problem can typically be solved much more efficiently than an NLP problem additionally with guaranteed convergence to the global optimality. CVX [189], a package for specifying and solving CPs, is readily available.

Quadratic Program

Furthermore, if the cost function is a (convex) quadratic function and all the constraint functions are affine, the problem can be formulated into a quadratic program (QP):

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{P}\boldsymbol{x} + \boldsymbol{q}^\top \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h}, \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \end{aligned} \tag{A.3}$$

where $\boldsymbol{P}$ is symmetric positive definite, $\boldsymbol{q}$ and $\boldsymbol{h}$ are constant vectors, and $\boldsymbol{G}$ is a constant matrix. Solving time can be further reduced with off-the-shelf QP solvers such as OSQP [180] and qpOASES [190]. Accordingly, if applicable people usually try to transcribe their problems into QPs which can be solved efficiently and steadily in real time.

Appendix B

Trajectory Optimization

Often the computation of robot motion is solving an optimal control problem (OCP), a control design process using mathematical optimization. The fundamental idea is to find a control over a period of time such that some measure of the system performance is optimized. Rather than trying to solve for the optimal control law for the entire state space, e.g., linear quadratic regulator (LQR), if only an open-loop trajectory is interested, e.g., the state $\boldsymbol{x}(t)$ and the control $\boldsymbol{u}(t)$ defined over a finite interval, the optimal control problem is then referred to as trajectory optimization (TO), which is widely adopted as a numerical approach.

Problem Formulation

A general TO problem is minimizing some performance index

$$J\left(\boldsymbol{u}(\cdot), \boldsymbol{x}_0\right) = \phi\left(\boldsymbol{x}(t_f), t_f\right) + \int_{t_0}^{t_f} L\left(\boldsymbol{x}(t), \boldsymbol{u}(t), t\right) \mathrm{d}t, \tag{B.1}$$

where ϕ is the endpoint cost and L is the running cost over time $t \in \left[t_0, t_f\right]$, subject to the first-order dynamics constraints

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), t\right), \tag{B.2}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{B.3}$$

and other additional constraints, e.g., path and boundary constraints [191]. As written, the problem is an optimization over continuous trajectories. In order to formulate it as a numerical optimization as introduced in Appendix A, we need to parameterize it with a finite set of variables. There are many different ways to write down this parametrization, e.g., transcription, shooting, and collocation [192].

Often times the TO formulation is an NLP problem in its original form due to the nonlinear system dynamics. To decrease the computation time, simplifications can be made to the robot model to make the optimization more tractable, e.g., the problem can be turned into a CP or even QP with a linearized dynamic model. However, an oversimplified model can limit the robot's motion capability. The challenge is thus to capture enough of the actual dynamics in the simplified model for successful control of the robot.

Model-Predictive Control

The computational efficiency of solving QP problems leads to an amazing idea: if we can optimize trajectories fast enough, we can use TO as a feedback control policy. The recipe is straightforward [193]:

1. Measure the current states.
2. Optimize the trajectory from the current states as the initial condition for a short time horizon in the future.
3. Execute the first action from the optimized trajectory.
4. Repeat for the new current states.

This recipe is known as model-predictive control (MPC) or receding horizon control as the prediction horizon keeps being shifted forward. Unlike classic control, MPC has the ability to anticipate future events and can take actions accordingly while addressing various system constraints. Although there is no closed-form representation of this policy nor general proof of long-term stability, in practice it has been proven an effective approach in robotics.

Appendix C

Special Orthogonal Group

Rigid body rotation is more complicated than translation as it cannot be represented by the common Euclidean space directly. The orientation is originally described by the rotation matrix evolving on the manifold of the special orthogonal group $\mathsf{SO}(3)$. Many researchers use Euler angles to parametrize the orientation. Despite their intuitive interpretations, using Euler angles and applying the regular techniques of Euclidean spaces are not properly invariant under the action of rigid transformations [194, 195]. Moreover, Euler angles suffer from the singularity issue. Unlike Euler angles, quaternion is a minimal globally nonsingular representation for orientation [196], but its state space of 3-sphere provides a double covering of $\mathsf{SO}(3)$ where a single orientation may correspond to two unit quaternions. This ambiguity should be carefully resolved, otherwise the unwinding phenomenon would occur where the body unnecessarily rotates through a large angle even if the initial orientation error is small [197]. Meanwhile, exploiting the manifold structure of rotation proves to be a general and decent approach in robot kinematics and control. This appendix thus collects some basic concepts [195, 198, 199].

Lie Group

The special orthogonal group is a subgroup of the general linear group, defined as

$$\mathsf{SO}(n) = \left\{ \boldsymbol{R} \in \mathsf{GL}(n, \mathbb{R}) : \boldsymbol{R}\boldsymbol{R}^\top = \mathbb{I},\ \det \boldsymbol{R} = 1 \right\}. \tag{C.1}$$

The dimension of $\mathsf{SO}(n)$ as a manifold is $n(n-1)/2$. For $n = 3$, the group $\mathsf{SO}(n)$ is also referred to as the rotation group on $\mathbb{R}^3$. The corresponding group operation is the usual matrix multiplication and the inverse is the matrix transpose.

Lie Algebra

The Lie algebra (the tangent space at the identity) of $\mathsf{SO}(3)$ is denoted as $\mathfrak{so}(3)$ which coincides with the space of 3×3 skew symmetric matrices. We can identify every skew symmetric matrix with a vector in $\mathbb{R}^3$ with the hat operator $\widehat{(\cdot)}$. If $\boldsymbol{a} = [a_1, a_2, a_3]^\top \in \mathbb{R}^3$, we have

$$\widehat{\boldsymbol{a}} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \in \mathfrak{so}(3). \tag{C.2}$$

The hat operator coincides with the vector cross product $\times$ such that

$$\widehat{\boldsymbol{a}}\boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}, \tag{C.3}$$

for all $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3$. A useful property of skew symmetric matrices is

$$\widehat{\boldsymbol{a}}\boldsymbol{b} = -\widehat{\boldsymbol{b}}\boldsymbol{a}. \tag{C.4}$$

The inverse of the hat operator is the vee operator $(\cdot)^\vee$ which maps a skew symmetric matrix to a vector in $\mathbb{R}^3$:

$$\widehat{\boldsymbol{a}}^\vee = \boldsymbol{a}. \tag{C.5}$$

Exponential Map

The exponential map (at the identity) $\exp : \mathfrak{so}(3) \to \mathsf{SO}(3)$ associates an element of the Lie algebra to a rotation, which coincides with the general matrix exponential and Rodrigues' rotation formula:

$$\exp\left(\widehat{\boldsymbol{a}}\right) = \mathbb{I} + \frac{\sin\|\boldsymbol{a}\|}{\|\boldsymbol{a}\|}\widehat{\boldsymbol{a}} + \frac{1-\cos\|\boldsymbol{a}\|}{\|\boldsymbol{a}\|^2}\widehat{\boldsymbol{a}}^2, \tag{C.6}$$

where $\|\cdot\|$ denotes the Euclidean norm, e.g., $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a}^\top \boldsymbol{a}}$. It can be shown that

$$\exp\left(\widehat{\boldsymbol{a}}\right)\boldsymbol{a} = \boldsymbol{a}, \tag{C.7}$$

i.e., the vector does not change if it is rotating along itself. Note that the exponential map reduces to $\mathbb{I}$ when $\|\boldsymbol{a}\| = 0$. For notational convenience, a capitalized $\mathrm{Exp} : \mathbb{R}^3 \to \mathsf{SO}(3)$ is usually adopted:

$$\mathrm{Exp}\left(\boldsymbol{a}\right) = \exp\left(\widehat{\boldsymbol{a}}\right). \tag{C.8}$$

A common first-order approximation of the exponential map is

$$\mathrm{Exp}\left(\boldsymbol{a}\right) \approx \mathbb{I} + \widehat{\boldsymbol{a}}. \tag{C.9}$$

Another useful property of the exponential map is

$$\boldsymbol{R}\mathrm{Exp}\left(\boldsymbol{a}\right)\boldsymbol{R}^\top = \exp\left(\boldsymbol{R}\widehat{\boldsymbol{a}}\boldsymbol{R}^\top\right) = \exp\left(\widehat{\boldsymbol{R}\boldsymbol{a}}\right) = \mathrm{Exp}\left(\boldsymbol{R}\boldsymbol{a}\right). \tag{C.10}$$

Logarithm Map

The inverse of the exponential map is the logarithm map $\log : \mathsf{SO}(3) \to \mathfrak{so}(3)$ which associates a rotation to a skew symmetric matrix:

$$\log(\boldsymbol{R}) = \frac{\theta}{2\sin\theta}\left(\boldsymbol{R} - \boldsymbol{R}^{\top}\right), \quad \theta = \arccos\left(\frac{\operatorname{tr}(\boldsymbol{R}) - 1}{2}\right), \tag{C.11}$$

where $\operatorname{tr}(\cdot)$ calculates the trace of a square matrix. Note that the logarithm map reduces to $\mathbf{0}$ when $\boldsymbol{R} = \mathbb{I}$. For notational convenience, a capitalized $\operatorname{Log} : \mathsf{SO}(3) \to \mathbb{R}^3$ is usually adopted:

$$\operatorname{Log}(\boldsymbol{R}) = \log(\boldsymbol{R})^{\vee}. \tag{C.12}$$

Right Jacobian

The right Jacobian of $\mathsf{SO}(3)$ relates an additive increment in the tangent space to a multiplicative increment on the manifold (applied on the right-hand side), defined as

$$\boldsymbol{J}_r(\boldsymbol{a}) = \frac{\partial \operatorname{Exp}(\boldsymbol{a})}{\partial \boldsymbol{a}}. \tag{C.13}$$

The right Jacobian and its inverse can be derived in closed form:

$$\boldsymbol{J}_r(\boldsymbol{a}) = \mathbb{I} - \frac{1 - \cos\|\boldsymbol{a}\|}{\|\boldsymbol{a}\|^2}\widehat{\boldsymbol{a}} + \frac{\|\boldsymbol{a}\| - \sin\|\boldsymbol{a}\|}{\|\boldsymbol{a}\|^3}\widehat{\boldsymbol{a}}^2, \tag{C.14}$$

$$\boldsymbol{J}_r^{-1}(\boldsymbol{a}) = \mathbb{I} + \frac{1}{2}\widehat{\boldsymbol{a}} + \left(\frac{1}{\|\boldsymbol{a}\|^2} - \frac{1 + \cos\|\boldsymbol{a}\|}{2\|\boldsymbol{a}\|\sin\|\boldsymbol{a}\|}\right)\widehat{\boldsymbol{a}}^2. \tag{C.15}$$

The right Jacobian has the following properties, e.g., for small $\delta\boldsymbol{a}$, we have

$$\operatorname{Exp}(\boldsymbol{a} + \delta\boldsymbol{a}) \approx \operatorname{Exp}(\boldsymbol{a})\operatorname{Exp}(\boldsymbol{J}_r(\boldsymbol{a})\,\delta\boldsymbol{a}), \tag{C.16}$$

$$\operatorname{Exp}(\boldsymbol{a})\operatorname{Exp}(\delta\boldsymbol{a}) \approx \operatorname{Exp}\left(\boldsymbol{a} + \boldsymbol{J}_r^{-1}(\boldsymbol{a})\,\delta\boldsymbol{a}\right), \tag{C.17}$$

$$\operatorname{Log}\left(\operatorname{Exp}\left(\boldsymbol{a}\right)\operatorname{Exp}\left(\delta\boldsymbol{a}\right)\right) \approx \boldsymbol{a} + \boldsymbol{J}_r^{-1}\left(\boldsymbol{a}\right)\delta\boldsymbol{a}. \tag{C.18}$$

The left Jacobian of $\mathsf{SO}(3)$ is similar.