UNIVERSITY OF CALIFORNIA

Los Angeles

Development and Real-Time Optimization-based Control of a Full-sized Humanoid for Dynamic Walking and Running

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

by

Min Sung Ahn

2023

ABSTRACT OF THE DISSERTATION

Development and Real-Time Optimization-based Control
of a Full-sized Humanoid for Dynamic Walking and Running

by

Min Sung Ahn
Doctor of Philosophy in Mechanical Engineering
University of California, Los Angeles, 2023
Professor Dennis Hong, Chair

Creating machines that resemble a human morphology have always been an aspiring goal for mankind even from the early days of engineering. Today, this effort continues to exist especially because of how much adaptive and flexible humans can be, as we can carry out a wide variety of tasks by collectively using our arms and legs. We can run, fall, get back up, and even use our arms or tools to provide additional stability when carrying out different locomotion or manipulation tasks. Imagine how much more helpful machines could be in our lives if we could build human-like machines that could do even a fraction of what we can. Traditionally however, humanoids have been big, bulky machines that moved around very slowly and were susceptible to disturbances and falling down.

Recently, with the collective advancement of key technologies, we have seen a rapid growth in the number of quadrupeds that can now dynamically walk and run in outdoor environments. Quadrupeds are inherently a stable platform, which is opposite to a biped that is inherently unstable. Yet, the underlying core principles in both hardware and software could be selectively adopted in a humanoid context to enhance their agility and robustness.

Hence, this dissertation is an extension of such ideas on a humanoid to make our human-like machines dynamically walk and run such that they can do more meaningful tasks in

the future. This requires a development of a hardware platform that uses similar design principles that were successful on quadrupeds, as well as a software and control stack that uses state-of-the-art techniques to robustly control the robot. Therefore, this dissertation introduces ARTEMIS, the first full-sized humanoid robot using proprioceptive actuators, its key design features, along with a real-time optimization-based dynamic locomotion stack that allows ARTEMIS to walk and run. Such hardware and software development allowed ARTEMIS to be the fastest walking humanoid reaching up to 2.1 m/s walking speed, be able to take aggressive pushes from all facets of its body, robustly walk without perception on debris cluttered terrain, and also be the very first humanoid fully developed in academia that can run. This opens up an exciting new chapter in the journey to developing humanoids that not only look like us, but can also robustly move and accomplish meaningful tasks.

The dissertation of Min Sung Ahn is approved.

Tyler Clites

Ankur Mehta

Paulo Tabuada

Dennis Hong, Committee Chair

University of California, Los Angeles

2023

To my family

and especially my grandmother,

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I have always prioritized doing what I enjoy and what I am passionate about. My PhD journey was no different. I freely explored various intellectual fields at my pleasure, tried to learn everything I wanted to know, and challenged myself to continue to push towards unknown directions without fear of a setback. Every experience was an opportunity to learn regardless of the outcome. This allowed me to devote the majority of my day and nights into my PhD life, as to me, this was the most enjoyable thing I could do! And it still is even as I now exit this program. I do realize this was possible because of the sacrifices others have made for me and their continued, overwhelming support. Thank you!

Dr. Dennis Hong, my advisor starting from my last quarter at UCLA as an undergraduate, throughout my Masters program, and now, even my Doctorate program. I can confidently say that my experiences as a graduate student would have been impossible without the environment and support he provided. His unconditional support for anything I pursued, the countless trips we made across the world to present in front of tens of thousands of people, and his relentless positive energy regardless of the situations we were in, gave me the necessary energy, freedom, and mental peace to do what I wanted to do at the Robotics and Mechanisms Laboratory. I am ready to embrace the fact that such support and environment will be next to impossible to come by again in life and I am immensely thankful to him for that. Not to forget the examples he led by demonstration in living life, in loving life, and also enjoying it. Thanks to him and this opportunity, I have a role model for life that I can always look up to and reference as I now live the rest of my life. Thank you so, so much.

My committee members, Dr. Paulo Tabuada, Dr. Tyler Clites, and Dr. Ankur Mehta, who played integral roles as I developed as a PhD student and as a researcher. Dr. Tabuada initially provided me and motivated me through his teachings in class, which played a crucial role in me building the confidence required to tackle such a cutting-edge and unknown field of hybrid dynamical systems such as humanoids. Many of the initial works in this dissertation

was possible because of the fundamentals he was able to provide.

Dr. Clites, with his immense biomechanics background, provided truly constructive opinions from a different point of view, which was necessary, as sometimes as a PhD student, you often get near-sighted and trapped in your field! Additionally, his emotional and mental support was also very helpful in me continuing to push forward through the ups and downs. Importantly, his views keep my pursuit for better humanoids unfinished, as he always provides room for improvement with inspiration from his background!

Dr. Mehta, who continued to push me to move out of my comfort zone and think about the bigger picture both through his classes and his advising sessions. Without his efforts, many of my ideas on where to take the next "step" with my passion such that it can be utilized for the better of other people and the greater world would not be where it is today.

To all the Robotics and Mechanisms Laboratory (RoMeLa) members since our beginning back in 2014. In no particular order, all those that I have worked with: Joshua Hooks, Jeffrey Yu, Hawkry Zhang, Hosik Chae, Alexie Pogue, Daniel Sun, Xuan Lin, Colin Togashi, Gabriel Fernandez, Fadi Rafidi, Nick Liu, Aaron Zhang, Junjie Shen, Justin Quan, Donghun Noh, Harry Nam, Yuki Shirai, Yusuke Tanaka, Alex Schperberg, Kyle Gillespie, Ji Sung Ahn, Aditya Navghare, Ohm, Ethan Hong. We've come far since then and I am thankful for all the help you have provided throughout the years. Robotics is a collective effort and I am thankful that we had a large but an extremely coherent group that we could all learn from.

All the robots in RoMeLa that I have been fortunate to work with and bringup. DARwIn-OP, THOR-RD, HEX, LARA, NABi-1, ALPHRED-1, SiLVIA, NABi-2, ALPHRED-2, BALLU, BRUCE, and ARTEMIS.

A special shoutout to Taoyuanmin Zhu, my partner in crime for the development of many of the above platforms, including ARTEMIS, as well as a great roommate for many years. Thank you for the great journey we have had so far and hoping for many more awesome years in the future!

My parents, brother, grandparents, and relatives, that sacrificed so much for me to be

able to live the worry-free life I can live today and the future I will be able to enjoy going forward. While this may just be a dissertation summing up a small portion of my PhD, your unconditional support and sacrifices were always there throughout my entire life, which is a massive privilege. Because of your efforts, we now have a PhD in our family! Especially to my mother who so dearly desired to study but could not, and my paternal grandparents and maternal grandfather who could not live to see this day come true, I hope this makes you proud. Thank you.

CURRICULUM VITAE

2015 – 2018	M.S. (Mechanical Engineering), UCLA, Los Angeles, CA, USA.
2008 – 2014	B.S. (Mechanical Engineering), UCLA, Los Angeles, CA, USA.
2015 – 2022	Graduate Student Researcher, UCLA, Los Angeles, CA, USA.

SELECT PUBLICATIONS

1. **M. S. Ahn**, T. Zhu and D. Hong. (2023). "Dynamic Walking and Running for the Humanoid Robot ARTEMIS."

2. **M. S. Ahn**, H. Chae, C. Togashi, D. Hong, J. Kim and S. Choi. (2022). "Learning-based Motion Stabilizer Leveraging Offline Temporal Optimization," in *2022 19th International Conference on Ubiquitous Robots (UR)* (pp. 129-136). IEEE.

3. H. Chae, **M. S. Ahn**, D. Noh, H. Nam and D. Hong. (2021). "BALLU2: A Safe and Affordable Buoyancy Assisted Biped." *Frontiers in Robotics and AI*, 290.

4. T. Zhu, **M. S. Ahn** and D. Hong. (2021, December). "Design and Experimental Study of BLDC Motor Immersion Cooling for Legged Robots," in *2021 20th International Conference on Advanced Robotics (ICAR)* (pp. 1137-1143). IEEE.

5. X. Lin*, **M. S. Ahn*** and D. Hong. (2021). "Designing Multi-Stage Coupled Convex Programming with Data-Driven McCormick Envelope Relaxations for Motion Planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 9957-9963). IEEE. (* equal contribution)

6. J. Hooks, **M. S. Ahn**, J. Yu, X. Zhang, T. Zhu, H. Chae and D. Hong. (2020). "Alphred: A multi-modal operations quadruped robot for package delivery applications." *IEEE Robotics and Automation Letters*, 5(4), 5409-5416.

7. **M. S. Ahn** and D. Hong. (2020). "Dynamic, Robust Locomotion for a Non-Anthropomorphic Biped," in *2020 17th International Conference on Ubiquitous Robots (UR)* (pp. 185-191). IEEE.

8. D. Hong, **M. S. Ahn**, J. Hooks and J. Yu. (2020). "Legged Robots, Design of," in *Encyclopedia of Robotics* (1-11). Springer.

9. **M. S. Ahn**, H. Chae, D. Noh, H. Nam and D. Hong. (2019). "Analysis and Noise Modeling of the Intel RealSense D435 for Mobile Robots," in *2019 16th International Conference on Ubiquitous Robots (UR)* (pp. 707-711). IEEE.

10. **M. S. Ahn**, D. Sun, H. Chae, I. Yamayoshi and D. Hong. (2019). "Lessons Learned from the Development and Deployment of a Hotel Concierge Robot to be Operated in a Real World Environment," in *2019 16th International Conference on Ubiquitous Robots (UR)* (pp. 55-60). IEEE.

11. J. Yu, J. Hooks, X. Zhang, **M. S. Ahn** and D. Hong. (2018). "A Proprioceptive, Force-Controlled, Non-Anthropomorphic Biped for Dynamic Locomotion," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)* (pp. 1-9). IEEE.

12. **M. S. Ahn**, H. Chae and D. Hong. (2018). "Stable, Autonomous, Unknown Terrain Locomotion for Quadrupeds Based on Visual Feedback and Mixed-Integer Convex Optimization," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 3791-3798). IEEE.

13. S. McGill, S. Yi, H. Yi, **M. S. Ahn**, S. Cho, K. Liu, D. Sun, B. Lee, H. Jeong, J. Huh, D. Hong and D. D. Lee. (2017). "Team THOR's Entry in the DARPA Robotics Challenge Finals 2015," in *Journal of Field Robotics*, 34(4), 775-801.

14. S. Yi, S. McGill, H. Jeong, J. Huh, M. Missura, H. Yi, **M. S. Ahn**, S. Cho, K. Liu, D. Hong and D. D. Lee. (2015). "RoboCup 2015 Humanoid AdultSize League Winner," in *RoboCup 2015: Robot World Cup XIX 19* (pp. 132-143). Springer.

CHAPTER 1

Introduction

1.1 Motivation

It is undeniable that robotics is becoming an integral part of our lives. Whether it is through the time-tested robot vacuum cleaners that are cleaning our homes or through the mobile robots that are increasingly becoming abundant in enclosed areas such as college campuses to deliver food and other necessities, it is clear that robotics is now moving beyond just factories and assembly lines, and is slowly moving out into the "wild" to play a key role in our day-to-day routines. Robots excel at executing specialized repetitive tasks that require accuracy and speed, and they will continue to find these tasks to assist parts of our daily lives.

At the same time, humanoids, possibly the most versatile and general purpose robots, are also well on their way. The most notable might be Boston Dynamics' Atlas robot, which can conduct tasks such as moving boxes from one location to another, but is also equally capable of executing parkour moves or imitating dance motions of popular K-pop stars. There also is Agility Robotics' Digit which is showing manipulation as well as locomotion capabilities in factory settings, while Tesla have recently started their development on humanoids as well. The paradigm of robots as single taskers are being broken and humanoids are at the most opposite end of that spectrum.

Humanoids' potential in our world is intriguing, especially when compared to the more traditional robot arms and wheeled mobile robots, because of their unique characteristics that stem from their anthropomorphic morphology. Even beyond just robotic arms and wheeled platforms, when compared with other legged robots such as quadrupeds and

hexapods that share a common characteristic (legs) with humanoids, there are unique advantages humanoids still provide.

- The most distinct characteristic and advantage is that humanoids are a breed of legged robots, which unlike robot manipulators that are often installed at a fixed base, can go to places! To further separate its uniqueness compared to wheeled robots which can also locomote, legged robots such as humanoids can even traverse non-continuous terrains such as stairs. By strategically moving its feet or hands that are in contact from one location to another, humanoids could traverse stepping stones to cross watercourses or even climb ladders to get to a higher platform. This gives humanoids an incomparable workspace that it can reach and conduct tasks compared to other breeds of robots, making them possibly the ultimate mobile manipulation platform.

- The amount of real-estate humanoids occupy on the ground is also significantly smaller than quadrupeds, regardless of whether they are in a cat-like shape or a sprawled shape, and comparable to mobile platforms with omnidirectional wheels. This can allow humanoids to effectively traverse extremely tight environments such as busy hallways or narrow corridors as the ground space it occupies is approximately the same whether it is turning in place, walking forwards, or walking sideways.

- Comparing specifically to other legged platforms, because humanoids predominantly locomote in a bipedal form, their workspace is relatively bigger. They can reach for higher places without the legs having to do specialized motion sequences (e.g. jump), and this bigger reachability could also help them move through far more diverse terrains. If needed, humanoids could also crouch down to conduct similar tasks a quadruped or a hexapod might do.

- Because of the bipedal form, locomotion and manipulation (i.e. locomanipulation) is also simultaneously possible. For a mobile robot, quadruped, or a hexapod, this would require an extra pair of arms to be attached.

- From a sensing perspective, a greater view of the surrounding environment is inherently

possible because there exists a higher point in the robot (i.e. head of a humanoid) compared to other robotic platforms that the cameras could be naturally mounted to. The flexibility in being able to adjust the position and orientation of the head from a high point means better field of view and superior understanding of the environment for potentially better planning algorithms to be deployed.

Beyond the unique advantages, the impact humanoids can have in our lives is diverse and the reasons to push its advancement is abundant. Its benefits could affect multiple disciplines both directly and indirectly. Just to highlight a few:

- Our human shape allows us to conduct a wide range of tasks and by mimicking this morphology, it also allows humanoids to be useful as general purpose robots that could be utilized for work as much as it could be utilized for entertainment.

- The most obvious work humanoids could directly help with is in handling the dull, dirty, and dangerous (DDD) tasks. Rather than people underutilizing their creativity by conducting "dull" jobs, or risking their health or even their lives performing dirty and dangerous duties, it naturally makes sense for robots to take on such burdens. Single tasker robots could be an option, but considering the plethora of different DDD tasks, a general purpose solution such as a humanoid whose intelligence could be tailored for a specific job seems like an attractive and effective alternative.

- With humanoids (as opposed to other robotic solutions), we do not need to recreate or retrofit the environment for it to be effective. Our world is designed for human morphology where, to name a few, doorknobs are located at a certain height, stairs have a standard height, depth, and incline per step, and furniture have standard sizes and shapes. Humanoids could immediately utilize many of the existing environment to effectively execute tasks, because their morphology is already well-suited to utilize the environment.

- Mimicking a human shape could also lower the hurdle robots have to overcome to collaborate with humans. Collaborative tools and procedures could be minimally adapted

for a human and a humanoid to collaborate. Even the most specialized, custom environments such as office spaces, hospitals, or homes could be reused.

- This suggests humanoids could also easily assist in interactive tasks where the majority of the interaction between a human and a robot is through a robot assisting a human. These tasks include pressing and unavoidable issues such as elderly care, child care, or care of those that are disabled or are in need of any help.

- Through the development of humanoids, byproducts such as a better understanding of the complexity with human motions can result in significant impacts in tangent fields. These understandings could result in more effective rehabilitation procedures for the injured or the development of better training programs for athletes, as well as the tools for the advancement of both.

- Similarly in related robotics fields such as in prosthetics and exoskeletons, the knowledge gained from humanoids could trickle down in making more effective devices for the patients.

- Humanoids could just as easily be used to represent a remote person through teleoperation. This can have significant implications in areas where an expert is needed, but may not be able to travel. Teachers could more effectively teach groups of students that are located in areas that are difficult to commute to and doctors could be able to immediately treat patients that they may not be able to get to immediately.

These dreams and promises of an automated, human-like machine helping us in our everyday lives are what fueled and will continue to fuel the humanoid sector. However, only a handful of adult-sized humanoids are now slowly making its way out of the lab, and a myriad of them are still in development or serving as predecessors to future platforms that could bring us closer to realizing the aforementioned benefits. So what is it with humanoids that are distinctively different, hindering their progress and efficacy as of today, compared to the robotic vacuum cleaner or the robotic manipulator arms? Are there other fundamental

shortcomings that needs to be resolved? Once they are resolved, how can we progress forward in the right way as quickly as possible?

It is true that humanoids do present technical challenges that are unique because of its mobile nature and anthropomorphic shape.

- By being a type of a platform that can roam the world at will, humanoids do not have a fixed contact with the world. This makes them what we call an "underactuated" system, which makes controlling them relatively a greater challenge, as some "states" can only be controlled indirectly.

- Speaking of states, not all the states of the robot are also readily available. This requires some sort of sensor fusion techniques to estimate a state that could potentially be important for control. Additionally, depending on the composition of the on-board sensors, some states may still be unobservable, requiring controllers to robustly handle situations despite the unobservable nature of the state.

- Humanoids and legged robots in general are also required to repeatedly interact with the environment. Their main form of locomotion is through breaking and making contact with the environment. As discrepancy in the shape of the actual environment and what the robot thinks is unavoidable, some form of compliance when coming in contact with the environment is necessary, whether it is through an inherent feature in the hardware or through the controller.

Along with these technical challenges, practical and engineering difficulties co-exist, which make actually deploying these systems in the real-world a significant challenge.

- From a practical sense, the accessibility to readily available and capable hardware for testing design and control approaches is still a luxury for many research groups. Not only are these hardware expensive, but they may also not exist in the first place, which would then require years of trial and error to building something that *might be* ready for use. This point is particularly important because as much as simulations are

useful, they are still inaccurate. The shortcomings of a controller, due to for example invalid assumptions or practical limitations of different sensors, could be less amplified in simulation, giving a false sense of belief that an approach may work in reality. Therefore, constant testing on hardware is imperative.

- Humanoids are also the ultimate integration of state-of-the-art hardware and software. You need capable hardware to run state-of-the-art algorithms, and you need state-of-the-art algorithms to verify the hardware's performance. Simultaneously, everything including batteries, actuators, sensors, filters, controllers, architectures, operating systems, and debugging tools to name a few, must collectively work to stabilize the system and create an effective development ecosystem. This is not an easy engineering task as seemingly everything is needed for even the most basic functionalities.

In this regard, this work is hopefully a contribution to taking another "step" (no pun intended) in the development of the next generation of humanoids in the real-world that will be able to do much more than just be a robot in a lab. And that first step, in my opinion, is being able to walk robustly and be able to get to places. Once we can get there, we will then think about the complexity of the task it needs to perform. Therefore, this work primarily focuses on the development and control of a new type of a humanoid for dynamic walking and running. My true hope is that both experienced engineers and junior graduate students or beginning researchers will be able to take something out of this, whether it is a small practical trick or a fast-paced bringup to the state of modern humanoids as of early 2023.

1.2 Background

1.2.1 A Historical Snapshot

1.2.2 Design of Humanoids and Legged Robots

Unsurprisingly, the development of humanoid robots have been very much a part of the development of legged robots in general, as platforms such as quadrupeds still share many

of the similar core technologies despite their silhouettes having distinct differences. This is because they face similar challenges mentioned in Section 1.1, such as underactuation and repeated impacts with the environment. Consequently, this background will encompass the entire field of legged robot design and how we have arrived at where we are today.

Legged robot design is a precarious balancing act that involves carefully selecting the best leg and feet configuration and type, actuators, materials, and sensors to yield a platform that is capable of achieving the robot's intended tasks. For example, in the case of legged robots designed to travel over multifaceted, unstructured terrain with discrete and isolated footholds, it is important that the robot be able to sense, coordinate, and place its end effectors (feet) at desired locations, while also being able to handle unexpected disturbances and variations in terrain. These qualities are just a few of the most rudimentary requirements for a legged robot, and yet they already imply the use of several tightly coupled subsystems.

The design of legged robots started as early as the mid 1800s with Chebyshev's "The Plantigrade Machine" and Rygg's mechanical horse [Ryg93], where well-designed linkage mechanisms with coordinated limb movements successfully made machines walk. However, these fixed gait mechanisms are incapable of traversing irregular terrain, so researchers moved towards adding more degrees of freedom (DOF) to each leg, where 3 is the minimum DOF required to move a feet to a desired position and 6 is the minimum to also define orientation, as evidenced by the GE walking truck, which was like Rygg's mechanical horse, except with 3 DOF per leg [LM68].

At a similar time, Ichiro Kato began his pioneering work on humanoids and bipedal machines, with the artificial lower limb model WL-1 in 1967. In two years, this led to WAP-1, a planar biped, which was followed by WAP-2 and WAP-3, which even walked in 3D. In 1973, he introduced the first full-scale humanoid, WABOT-1 [KOK74], followed by a number of bipedal robots with quasi-dynamic gaits in the 80s. This work initiated the field of humanoid robots, and led to the development of platforms such as Honda's P2 in 1998 [HHH98], the HRP (Humanoid Robotics Project) humanoid series [IH00], and later, the acclaimed ASIMO in 2002, Honda's humanoid that can hop, run, kick a ball, and even

dance [SWA02]. An interesting extension to the predominant humanoid research is the evolution of the robots' feet. While most have a single plate as a foot, platforms such as WABIAN-2R and LOLA have toe actuation like a human [HTH10, LBU09].

Marc Raibert started work on hopping robots in 1981, where his focus on dynamic locomotion systems led him to build a hydraulically actuated monoped that needed to continuously hop to keep balance [Rai86]. He also developed a biped and a quadruped, which were simply extensions of his hopping monoped into two and four legs. Raibert's robots embodied the idea of keeping most of the inertia at the body as opposed to the legs. Raibert went on to establish Boston Dynamics (BDI), making iconic hydraulically actuated systems such as Big Dog [RBN08], WildCat, Spot, and Atlas.

A very different legged robot design appeared in the 1990s when Tad McGeer developed a mechanically clever biped that used its passive dynamics to walk [McG90]. McGeer's passive approach contrasts Raibert's hydraulically actuated robots, but they shared the same principle of using what is commonly referred to as template models. Template models are simple models useful for studying stability during periodic motion and can capture the salient dynamics of a particular system. These results showed the use of models and fundamental dynamics' importance during locomotion, motivating researchers to make robots more like simple models which happen to be easier to analyze and control too. Section 1.2.3 will cover more on these template models.

Therefore, the current trend on platforms that demonstrate robust locomotion is lightweight legs with small feet. This effectively reduces the leg's inertia, which provides two main advantages. Firstly, this allows the legs to accelerate at high rates, while minimally influencing the dynamics of the overall robot. Secondly, small feet prevent undesirable complexities when coming in contact with the ground. The smaller the feet, the less likely an unexpected collision with the environment will occur. This also brings the physical hardware closer to the well-studied point mass template models.

To achieve this paradigm in practice, different approaches have been attempted. The option to relocate the actuators closer to the robot's body, as opposed to collocating it at

the joint is the predominant choice for its ease and most energy efficient mass distribution. To achieve this, either a belt/chain design or a pantograph mechanism is used. The former provides a greater range of motion albeit the potential maintenance required and the non-rigidity of the transmission which can potentially complicate the control, while the latter offers the opposite pros and cons. Some researchers have also created underactuated legs with passive joints to reduce the number of actuation required to begin with. All three approaches have demonstrated their capability to achieve lightweight legs.

Beyond relocating actuators, easier access to different materials and advanced manufacturing techniques have contributed to the development of robust, lightweight legs. For example, carbon fiber and different aluminum alloys are more widely used because of their attractive high strength-to-weight ratio [YHG16]. Additive manufacturing techniques are also actively used with modern optimization and analysis tools, previously considered too computationally intensive, to create durable, low inertia legs [SGM15].

However, if a legged robot is designed to primarily operate in a static equilibrium as opposed to a dynamic walking state, the aforementioned principles are modified, particularly with the feet. Especially with bipedal robots such as humanoids, since the system is inherently unstable unlike a quadruped, non-point feet with ankle actuation are necessary to stand still and maintain balance at the price of expensive torque requirements. Therefore, in the case with bipeds that are designed to be dynamically and statically stable (i.e. balancing without walking cycles), non-point feet are used. Subsequent control algorithms can utilize the ankle actuation as a damped passive joint during dynamic behaviors such that the robot behaves as if it has point feet, while the algorithms precisely control the ankle joint during static equilibrium. Additional actuation in the feet (i.e. toes) have also been investigated for their energetic and locomotion speed advantages, but are not yet a widely adopted feature [LBU09].

In recent years, the idea of embedding the prominent locomotion dynamics such as compliance into the mechanical design has continued to be the trend both in the academia and the industry. Jonathan Hurst has tried to bridge the gap between the dynamic model and

the physical robot in his bipeds ATRIAS and Cassie, which use the series elastic actuation paradigm proposed by Gill Pratt [PW95], with both platforms showing impressive locomotion capabilities [HGJ16]. Sangbae Kim has shown that a passive elastic element is not always necessary through his use of proprioceptive actuation—an idea which he brought from the field of haptics—which he proved through his quadruped MIT Cheetah [WWS17]. Raibert has emphasized that BDI focuses on the simple dynamics, while Spot and its initial prototype show that robust locomotion can be achieved regardless of whether the actuation is hydraulics or electric motors, although additional details are not readily available. These systems show that there is not yet a standard approach to compliance, and the designer needs to decide what is suitable for different scenarios.

Currently, the electric motor is the most common form of actuation in robotics. There are three common types of electric motors used in robotics: brushed DC motors, stepper motors, and BLDC motors. Both brushed DC and stepper motors are cheap and easy to use, but are bulky and inefficient. BLDCs require a motor controller making them harder to use, but they have high efficiency and high power density [Han06] making them better suited for legged robotic applications.

Traditionally, legged robots have used BLDCs combined with large gear reductions to provide high torque and low power consumption. However, such configuration is mainly suited for position control, and it has been found that legged robots that utilize force control strategies to control the GRFs are able to adapt better to their environments, producing much more robust locomotion. Many researchers have had success by adding force-torque (FT) sensors to BLDCs with high gear reductions to provide force feedback. However, FT sensors are fragile making them not ideal for dynamic locomotion. For this reason, much of the focus in recent years have been around series elastic actuators (SEA) and proprioceptive actuators.

SEAs are typically made with highly geared BLDC motors in series with a spring element whose displacement can be measured to calculate the output force [PW95]. This type of actuator retains the advantages of high torque output and efficiency provided by the large

gear reduction while also providing impact mitigation and force control. However, it suffers from reduced speeds due to the gear reduction and limited bandwidth depending on the stiffness of the elastic element.

Robots that utilize SEAs are typically designed for dynamic types of locomotion, but these actuators also allow for more static types of locomotion due to their large continuous torque outputs. Robots that utilize these actuators can be cleverly designed with reduced leg inertia for increased dynamic capabilities by moving the actuators closer to the center of mass (CoM), but this is typically not the case. The stiffness of the elastic element must go into the design of the overall robot. A "soft" spring will result in poor bandwidth, whereas a "stiff" spring will result in poor impact mitigation.

Proprioceptive actuators have been growing in usage in the legged robotic community over the last decade. These actuators comprise simply of high torque BLDC motors with little to no gear reduction, where the large motor compensates for the reduced torque amplification from the gearbox. The advantage of these actuators is that they have highly "transparent" transmission systems which allows for fairly accurate force control by monitoring the current in the motor [SWO12, KDK16]. Though this method is not as accurate as using an FT sensor, it has been shown to be sufficiently accurate for several legged robots. Furthermore, the actuator's stiffness can be changed at runtime by the motor controller without loss of bandwidth. The main drawback of these actuators is their high heat production caused by running the large BLDC motors at low speeds, meaning they are not well suited for tasks requiring high continuous torque output.

Robots that utilize proprioceptive actuators use a force control architecture for high-speed locomotion, meaning once again moving the actuators closer to the CoM to reduce the leg inertia, as well as reducing the foot to a point or using an underactuated ankle. Due to the poor continuous torque of the actuators, these robots are designed such that the actuators do not have to apply much force when in a static pose (i.e. the legs can be almost locked out when standing still).

Unlike the aforementioned electric motor based actuators, hydraulic actuators control

force or position by regulating the pressure of the hydraulic fluid through a series of valves. These types of actuators have a much larger torque density than electric actuators, but require a pump to pressurize fluid for actuation. Traditionally, the only actuators capable of sufficiently accurate force and position control were designed for the aerospace industry, making them over-designed and prohibitively expensive for legged robotics. However, BDI has made great strides to make these applicable for legged robots. Still, only a few groups other than BDI have been successful in implementing hydraulics on legged robots.

The design of a robot with hydraulic actuators requires a centralized hydraulic system, which consists of a hydraulic pump and a network of hydraulic lines and fittings. Typically, due to the pump's size and inertia, it is located near the robot's CoM. This design's advantage is that electrical power and cooling can be centralized at the pump. However, a network of hydraulic lines must run to each of the actuators and considerable care must go into the design and routing of the hydraulic lines to reduce the chance of catastrophic failure if a line is severed.

Along with compliant actuation, there have been other key results in recent years that have allowed the community to design robust legged robots. Particularly, the advancement of technology in different fields, especially in regards to fundamental components that comprise a complete system, are allowing us to unlock designs that may have once been considered impossible, now a reality.

Legged robotics is a complex field where design alone cannot solve the many challenges it poses. It is indeed the co-optimization of the robot design, along with the integration of different sensors, various control algorithms, and more that produces an effective end result. However, a well-designed system is the most fundamental basis upon which further development can be done.

1.2.3 Control of Humanoids and Legged Robots

Position control continues to be the predominant form of control in robotics. This is driven by the tasks that many of the robots are engineered to conduct, which require high positional

accuracy. For example, robot arms in an assembly line need to precisely move components from one place to another. Additionally, position control schemes do have the benefit of being easier to visualize how the robot will behave in an environment. We can also easily verify while the robot is playing through its trajectories whether it is behaving as intended. Position control does have its shortcomings in its lack of adaptability to the environmental uncertainties. For example, if the robot is trying to reach a certain position but is denied because an obstacle is in the way, the robot could end up applying a huge amount of force to the environment. On the other hand, if an interaction with an object is desired but the object is further away than originally expected, the robot could prematurely stop its motion before it comes in contact with the object.

For humanoids, interaction with imprecise environment information is unavoidable so having an adaptable control approach is beneficial. In that regard, torque control is an attractive option and has increasingly become a popular approach for the humanoid community as well as the legged robotics community in general. It allows the quick adaptability we desire on our systems and also resembles the low impedance behavior seen in nature [Pra02]. If an unknown force is experienced by the system, the system should react to it rather than be resistant to it.

Torque control is not a completely new concept, but the paradigm has increasingly been used since the 2010's. In the context of humanoids, tracking multiple objectives under the inverse dynamics framework has been actively researched. Also known as whole-body control, Khatib pioneered the field where the torques to be sent to the humanoid were found under pre-defined desired motion tasks and dynamic constraints by using the null-space projection techniques [KSP04, SK06]. Whole-body control has also been formulated to be contained in convex optimization formulations. The decision variables are the torques to be applied to the system with the cost function designed to minimize the different tracking errors for tasks such as the center of mass position, joint position, end-effector pose, and momentum. These approaches are general enough to be used for manipulation, locomotion, or any multi-task during multi-contact problems such as locomanipulation (manipulation during locomotion).

The exact details on these fundamental formulations are further described in Section 3.2.6.

The question then is where do these reference motions come from. Usually there exists a higher-level motion planner with feedback from the current state of the platform. This high-level planner for humanoids, and also generally for legged robots, computes when and how to make contact with the environment, whether it is with the ground or some object that it needs to interact with. For a planner to plan these contacts, a representation of the robot model is required. This model can be the complete dynamic or full-order model of the robot, which gives the planner the benefit of being able to leverage the minute dynamic details present in the model. It can also be a reduced-order model that captures the salient dynamics of the complete dynamic model. Many times the details from the full-order model may not be required. Reduced-order models provide computational efficiency because of their simplicity, but at the cost of inaccuracy. Reduced-order models can also make interpreting and analyzing the results easier.

If we were to plan using the full-order model, the complete equations of motion for a dynamical system is used as follows:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}\mathbf{g}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^\top \boldsymbol{\tau} + \mathbf{J}^\top \mathbf{f}. \tag{1.1}$$

The generalized coordinates $\mathbf{q}, \dot{\mathbf{q}}$ contain not only the actuated joints' positions and velocities that determine robot's posture and its change over time, it also contains the robot's base (i.e. first) frame with respect to a global fixed origin. For the generalized positions $\mathbf{q}$, this would be the position and orientation of the frame with respect to the world in the world's frame, and for the generalized velocities $\dot{\mathbf{q}}$, this would be the linear and angular velocity of the base frame.

To simplify Equation (1.1), the first assumption we can make is the sufficient torque assumption which states that the robot has sufficient torque to actuate the joints. This allows us to focus only on the floating base dynamics or the unactuated joints from the full order model:

$$\begin{bmatrix} m(\ddot{\mathbf{p}}_G - \mathbf{g}) \\ \dot{\mathbf{L}}_G \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\tau}_G \end{bmatrix} \tag{1.2}$$

where m is the mass of the robot, $\ddot{\mathbf{p}}_G$ is the position of the center of mass with respect to the world, $\mathbf{g}$ is the gravity vector, $\mathbf{L}_G$ is the total angular momentum about the center of mass, $\mathbf{f}$ is the total external force from contact, and $\boldsymbol{\tau}_G$ is the moment from the contact forces about the center of mass. This is equivalent to the Newton (linear) Euler (angular) equations and is known as the *centroidal dynamics model*.

This can further be simplified with additional assumptions to obtain a linear model. The nonlinearity is introduced through the angular momentum and the change in the height of the center of mass as seen by its definition

$$\dot{\mathbf{L}}_G = \Sigma_i(\mathbf{p}_i - \mathbf{p}_G) \times \mathbf{f}_i + \boldsymbol{\tau}_i.$$

where i corresponds to the i-th contact. If we assume zero angular momentum about the center of mass (i.e. $\dot{\mathbf{L}} = 0$) as well as a constant height of the center of mass $\mathbf{p}_G = H$, which are both reasonable assumptions [HP08], we can derive a linear model known as the *linear inverted pendulum model (LIPM)*. This model assumes the robot to be a point mass at a constant height supported by a massless leg in contact with the ground. LIPM is often referred to as the model for walking [Ale76, KKK01].

There have been many variations to this locomotion model to overcome the limitations that the aforementioned assumptions enforce. For example, our center of mass height with respect to the contact position on the ground is not strictly a constant value, and to overcome this assumption, variable height models have been investigated [CEL19, PD07]. Efforts to consider a non-zero angular momentum about the center of mass also exist, where the point mass can be replaced with a flywheel [PCD06].

A similar simplified model that replaces the constant stance leg with a compliant model is the Spring Loaded Inverted Pendulum (SLIP) model [Bli89, MC90]. Extensively explored in both legged robotics and biomechanics, the SLIP model originates from studies in biology where many animals demonstrated a center of mass trajectory resembling the SLIP's dynamics [FK99, DFF00]. This has led to not only designs that try to replicate the SLIP model [HGJ16], but also controllers that try to adhere to it [RHJ15]. SLIP models are versatile in that they can be used for walking as well as running, but its use can be limited

because it lack an analytical representation because of its nonlinearity [YGD20]. There are also variations to the SLIP model such as an Asymmetric Spring Loaded Inverted Pendulum (ASLIP) which considers the orientation dynamics of the robot's body (which a point mass does not model) [PG09]. A flywheel has also been attached to the SLIP model (Flywheel-SLIP) to realize motions such as somersault on a bipedal robot [XA20]. A dual SLIP model that appends the point mass with two springs has also been used as the template model for locomotion [LWO15, LWS16, AH20].

Efforts to include angular momentum or orientation suggests that a middle ground between full-order models and reduced-order models using only a point mass could be useful. A recently popular approach in the quadruped community is the *Single Rigid Body Model (SRBM)* and it assumes a point mass with a fixed moment of inertia about the center of mass. These assumptions are valid if the legs have negligible inertia compared to the body, as the legs' motions will have minimal effect on the torso [BPK18,DWK18,HAY20,KDK19b, DPL21]. Note how this is in parallel with the design efforts in Section 1.2.2 which tries to minimize the distal mass and leg inertia. For practical reasons with a humanoid only having two legs compared to a quadruped with four and the limited number of legs requiring more degrees of freedom (i.e. more actuators) to properly position and orient its end-effector (i.e. foot), the lightweight leg assumption is usually less accurate.

To plan a sequence of contacts using these different dynamic models, different strategies can be deployed. In the context of locomotion, different stability measures exist and many of the contact planning strategies revolve around incorporating these stability measures. The most notable is the concept of the Zero Moment Point (ZMP) [VS72], which is a contact point on a planar surface where there are no moments in the horizontal direction. This led to one of the most popular stability criterions that states that if the ZMP is within the convex hull (i.e. support polygon) of the foot, the robot will remain stable. Another important stability measure is the Capture Point (CP) [PCD06] or capturability, which is an effective measure when the planar contact surface assumption is no longer valid. This is a point on the ground which if the robot places its foot, it will be able to come to a complete stop. If

the CP is within the support polygon, the robot will be able to maintain its stability without taking a step, whereas if it exists outside the support polygon, a step will be required to recover from falling.

With these dynamic models and some approach to measure the "stability" of a robot, there are seminal works such as the ZMP preview controller [KKK03], which plans footsteps and ZMP trajectories to smoothly generate a center of mass trajectory to follow. More recently however, to be more robust and reactive to unexpected disturbances, many control approaches are using model-based optimization and have enjoyed much success. Simplifications to the aforementioned dynamic model to make the problem convex and solve for torques to apply to the system are an extremely popular approach at this time [DWK18, HAY20, DPL21, GGP19], while having a slower optimization loop to seed a faster optimization is also a framework we are now starting to see [KDK19b]. Many of these approaches use some sort of a Model Predictive Control (MPC) to provide the controller with some horizon information, which has shown to allow the system to react more robustly yet less aggressively. Most recently tho, researchers are investigating going beyond convex problems and solving nonlinear problems that include the complete dynamics of the system over a given horizon, which results in a nonlinear MPC. This is a difficult problem to solve online because of the nonlinearity and high-dimensionality of the problem and therefore is an active area of research [DVT14, FJS17, GFR19, RBF21].

1.3 Overview

The overview of this dissertation will be as follows. In Chapter 2, a humanoid robot developed based on the findings in Section 1.2.2 are presented. Some of the development decisions that will be beneficial later on to the control will be highlighted. Afterwards, Chapter 3 details the software stack on the robot as well as its current dynamic locomotion stack to get a modern humanoid platform to robustly walk and even run. Chapter 4 then explains the implementation details that allowed state-of-the-art optimization-based approaches to be effectively run online on the system, as well as results to verify the robot's locomotion

performance. Now that the robot is capable of robustly walking outdoors to get to places, some ideas on direct improvements to the software stack as well as work in less pertinent fields based off of this work is presented in Chapter 5. The dissertation is concluded with Chapter 6.

CHAPTER 2

ARTEMIS: A Robot for Dynamic Locomotion and Hyper-Dynamic Motions

2.1 Introduction

Robots that resemble a human appearance have been around for a long time, but only in the last century are we seeing humanoids that can do the following:

- Sense: Whether it is through exteroceptive sensors or proprioceptive sensors, the robots will *sense* some form of signals to later process for planning.
- Plan: Based on the provided information from the sensors, the robot will digest the information and *plan* an action to take.
- Act: From the plan, the robot will execute some physical *action* in the world.

For example, a walking humanoid robot equipped with force-torque sensors will first *sense* whether its foot is in contact with the ground. Based on the contact information, it will *plan* whether it should start shifting its body towards the foot that is in contact. After the plan, the robot will *act* it out in reality.

Despite the near 100 year history of humanoids, only recently in the last decade are we seeing humanoids moving out of its controlled laboratory environments and out into the wild. As introduced in Section 1.2.2, a combination of start-of-the-art approaches in both design and control are what is enabling this endeavor. Many of these "modern" humanoids adopt these key design elements (actuation, leg design, sensors), which allows them to seamlessly

Figure 2.1: ARTEMIS, a next-generation humanoid designed for dynamic and agile motions.

coalesce with state-of-the-art control approaches and step out into the outside world. This section introduces ARTEMIS (Advanced Robotic Technology for Enhanced Mobility and Improved Stability), a next-generation full-sized humanoid robot platform designed for hyper dynamic motion, as well as how it differs from other recent humanoid platforms that share similar design elements.

2.1.1 Existing Platforms

To highlight the distinct characteristics of ARTEMIS, I will first introduce other existing humanoid platforms that share similar, yet different, principles that are helping them take

steps to robustly performing their desired tasks. For example, even if the actuation type is the same for two robots, their core designs may slightly differ because of the tasks they desire to do.

Boston Dynamics' Atlas is the most well-known humanoid platform that has gone through many iterations to become what it is today. Based on observation of the available media, the most recent version of Atlas has 6 DoF per leg and 7 DoF arm, with no actuated neck. It is powered using hydraulic actuation, which gives it the best power density among the modern humanoids. The leg design and manufacturing is seemingly based on additive manufacturing where the hydraulic components are part of the structural components of the leg. Otherwise, not much information about Atlas is readily available for the general public.

A relative veteran in this group, TORO is a torque-controlled humanoid robot that is 1.74 m tall and weighs 76.4 kg [EWO14]. It has 27 joints in total, with 25 of them being torque controlled and the remaining 2 being position controlled. The legs and arms each have 6 joints (totaling 24) and the remaining three joints are located at the waist and the neck. The modular torque controlled joints consist of a torque sensor with a Harmonic drive gear unit, which makes TORO unsuitable for running or jumping, but still capable of investigating torque-based algorithms using it. The hardware demonstrated multi-contact balancing, whole body locomanipulation tasks, as well as human robot interaction tasks.

LOLA is also a relatively older robot from this group as well and is 1.8m tall and weighs 68 kg [LBU09, SWS21]. It has 25 DoF with a redundant kinematic configuration at the legs. The legs consist of 7 DoF each (rather than the more conventional 6 DoF) with the additional DoF existing at the toe, while the remaining 11 DoF come from the arms (3 DoF each), the pelvis (2 DoF), and the head (3 DoF). LOLA's actuators also use Harmonic drive gears and planetary roller screws. It has a force/torque sensor at its feet to measure the wrenches at the feet. This enables platforms like LOLA to control the force it wants to apply to its surrounding environment for tasks such as locomotion.

HRP-5P is a relatively modern humanoid built for practical use in industrial sites [KKS19]. Because of its use case, it has 37 DoF, which is considerably more than other platforms.

Specifically the arm has 8 DoF with an additional 2 DoF in the hand, while the trunk has 3 DoF, which gives the robot greater freedom when handling objects such as plasterboards in a construction site. It is also 1.83 m tall and weighs 101 kg [KMS19] such that it can directly replace a human worker. HRP-5P also has force/torque sensors at its wrist and ankles to control the force it should apply to the environment during locomotion and manipulation.

THOR is a 34 DoF humanoid designed for disaster response in place of or in collaboration with humans [Lee14, HLL16]. Thus, it has a similar profile to a human, as it stands 1.78 m tall and weighs 65 kg. THOR uses a combination of custom compliant linear series elastic actuators in the lower body and stiff rotary actuators in the upper body to control the contact forces during locomotion and manipulation. This enables the platform to adapt to the uncertainties in the environment.

WALK-MAN is also a humanoid tailored for disaster response [TCN17]. The creators focused on reducing the leg inertia, improving the power-to-weight ratio, as well as the increasing the range of motion, to strive for human like dynamic motions. In total, the robot has 29 DoF, with 12 DoF in the lowerbody and 17 in the upperbody. WALK-MAN uses a combination of a Harmonic drive gearbox and a flexible element inside a custom actuation unit, which enables the torque sensing at the joint level. It also has a 6 DoF force/torque sensor integrated in the wrists and the ankles.

Digit from Agility Robotics might be the most drastically different humanoid compared to the rest, while the most similar to ARTEMIS. It is 1.6 m tall, weighs 48 kg, and has 20 actuated joints. There are 4 passive joints in the lowerbody, originating from leafspring four bar linkages, which gives Digit room for compliance when interacting with the environment. Digit also follows the lightweight leg design ideology, where distal mass with respect to the body are minimized.

Compared to the above platforms, ARTEMIS has many unique characteristics that differentiate it from the rest. One of the key differences is the use of custom proprioceptive actuators. The gear ratios for the different actuators for the different joints are shown in Table 2.1. This design choice is partly driven by past experience in integrating them into

Joint Name	Gear Ratio
Hip Yaw / Roll	14.46154
Hip / Knee Pitch	5.93103
Ankle Pitch	50.28099
Arms	19.1916667
Neck	9

Table 2.1: Gear ratios of the custom proprioceptive actuators used on ARTEMIS.

quadrupeds [HAY20, Zha19] and non-anthropomorphic bipeds [YHG16, AH20, Yu20] as well as from the immense success the quadruped community has recently been experiencing by using them. They provide a good balance of advantages that are beneficial for legged robots that must adapt to the uncertainties presented by the outside world. In fact, to the best of our knowledge, it is the first full-sized (i.e. a standing size comparable to an adult) humanoid to use proprioceptive actuators, which also has its disadvantages especially as the platform's size starts to scale up. Furthermore, careful mass distribution and additional exteroceptive sensors are strategically planned and embedded to assist control.

In the subsequent sections of this chapter, we introduce the platform and its unique features compared to other platforms in-depth. Afterwards, the mathematical modeling of ARTEMIS is done. This includes how we define the frames, the platform's kinematics, as well as its dynamics.

2.2 Design Overview

ARTEMIS (Advanced Robotic Technology for Enhanced Mobility and Improved Stability) as seen in Figure 2.1, is a next-generation full-sized humanoid robot platform designed for dynamic locomotion and agile motions. It is also the first full-sized humanoid in the world that makes use of proprioceptive actuators and is designed to dynamically interact with the outside environment. It was originally crowdfunded in late 2018 as seen in Figure 2.2, and was fully built in the summer of 2022.

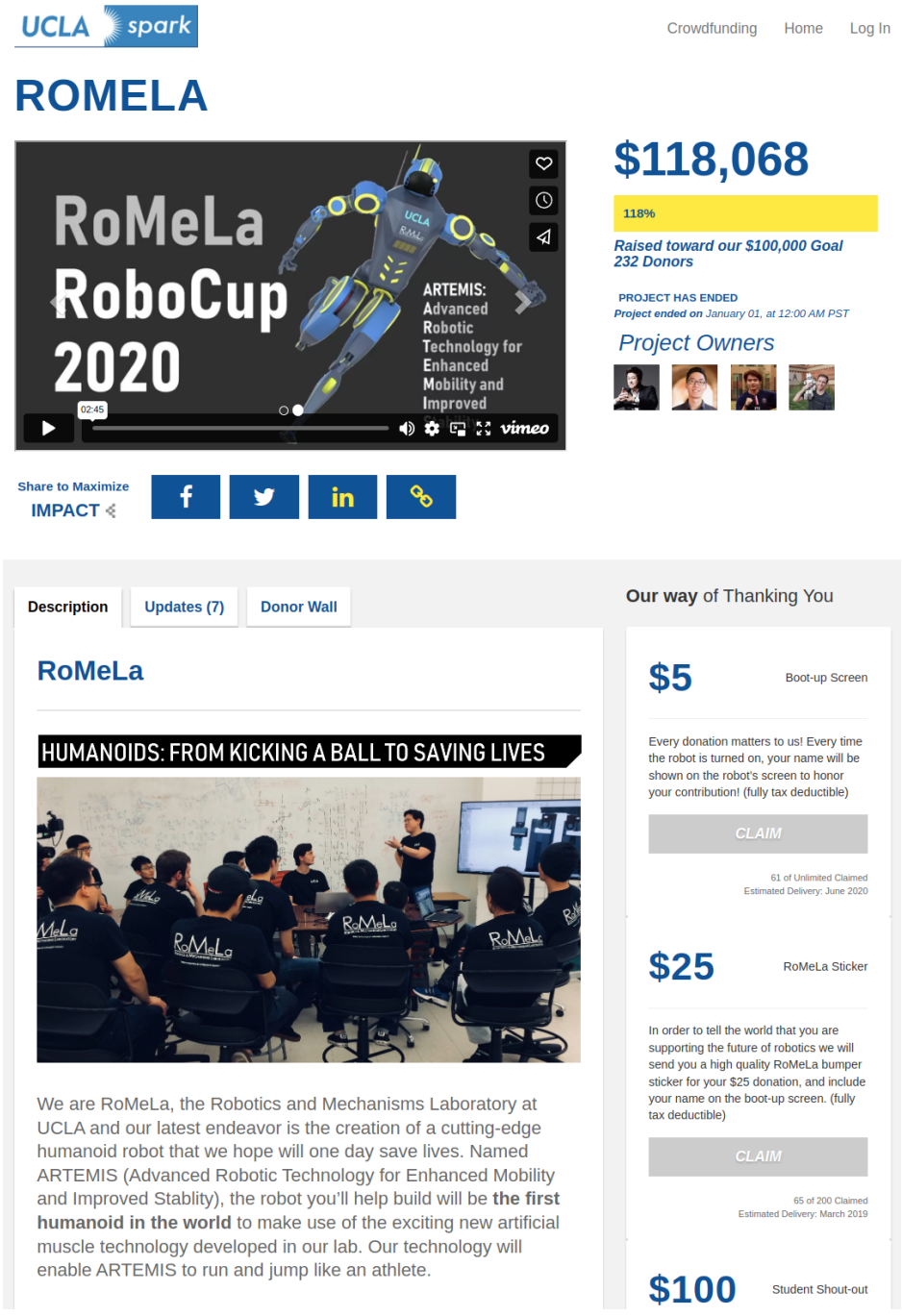

Figure 2.2: Crowdfunding through UCLA Spark to fund the development of ARTEMIS.

2.2.1 Leg Design Ideology

ARTEMIS is equipped with a suite of custom actuators and sensors that make it suitable for exciting new research in dynamic and agile locomotion, locomanipulation, robot dancing, and motion retargeting to name just a few of the many research fields it is suitable for. Overall, the robot is actuated using 20 custom BEAR (Back-drivable Electric Actuator for Robotics) actuator modules [ZHH19, ZAH21]. The lowerbody has 10 actuators (5 per leg) while the upperbody has the remaining 10 actuators (4 per arm, 2 for neck).

Recalling recent trends from Section 1.2.2, ARTEMIS follows a similar design ideology. It is designed with the goal of reducing as much distal mass as possible. This effort can be seen primarily in three different areas of the design.

The first is through the linkage design. A knee and an ankle are essential components for a human, as well as for a humanoid robot. Especially a knee is needed for any form of motion, whether it is running, jumping, stepping down a step, or stepping up a ladder. The ability to retract our legs closer to our body gives us the flexibility to traverse various terrains. Similarly, the ankle is also important in providing stability during any sort of motion because without it (i.e. a point feet), we would have to continuously take steps or significantly use our upperbody to stay balanced. Therefore, an actuator that controls the knee and the ankle are a necessity. The most simple approach would be to have the actuators exactly where the joint is located, but as mentioned in Section 1.2.2, there are different viable approaches to actuating the joint using an actuator mounted elsewhere. In the case with ARTEMIS, it uses a linkage mechanism to actuate the knee joint from an actuator at the hip, and the ankle joint from an actuator at the knee. The knee joint being actuated through a linkage is particularly important as it is often the joint that requires the most amount of torque, which then requires the biggest (heaviest) actuator.

The second is by embedding the actuators into the structural design of the system. There are benefits of using a modular, off-the-shelf actuation module because of its ease of use and serviceability [MYY17, YHZ18, HAY20]. It does come at the cost of adding unnecessary mass which is usually undesirable for mobile systems such as a legged robot. As

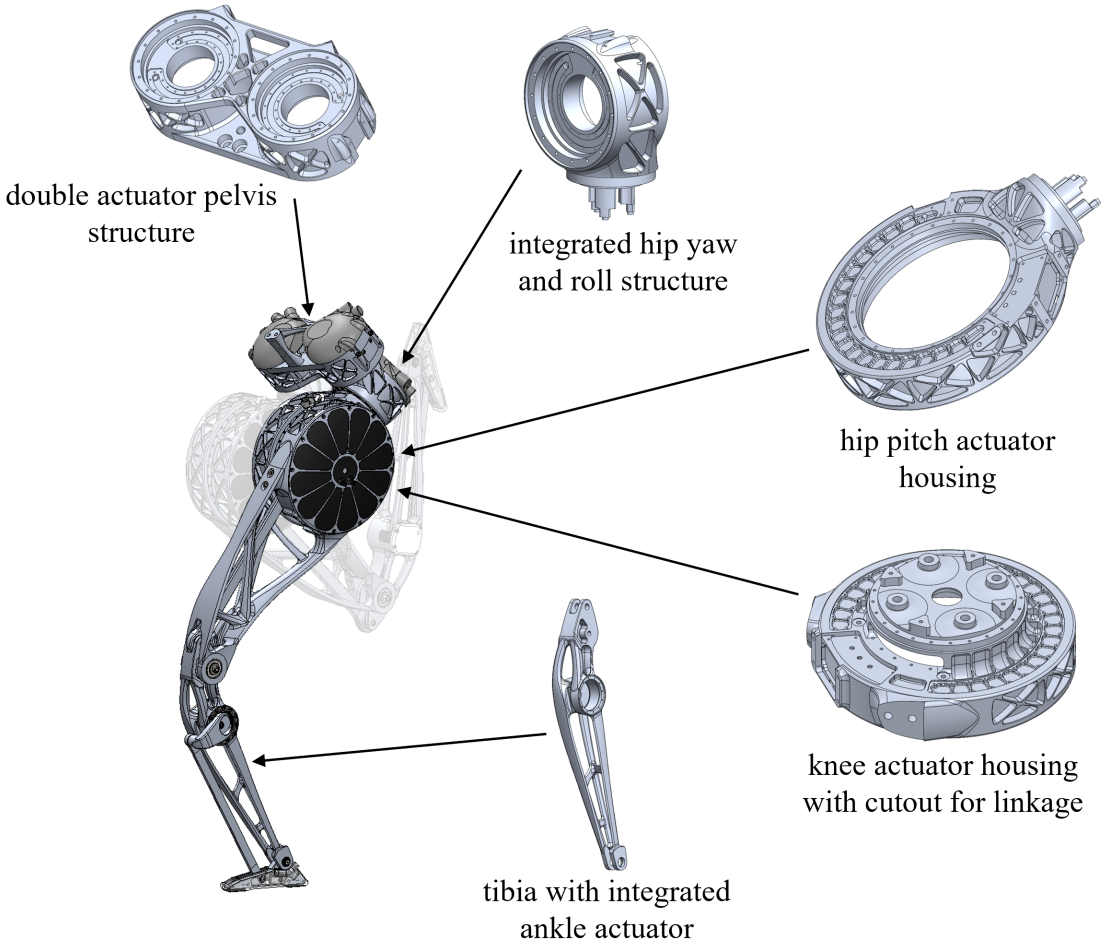

Figure 2.3: Closeup of the tightly packaged hip configuration in ARTEMIS.

ARTEMIS was designed from scratch with modern design principles in mind, it embeds the custom actuation modules into the robot's structural design. This allows tighter packaging of multiple actuators into the hip area of the robot as seen in Figure 2.3 which reduces unnecessary mass that would have been added from the extra housing that modular actuation units come with.

Lastly, ARTEMIS' structural design uses modern optimization-based design techniques [HAH20]. Given a set of loads and constraints on a simpler leg design, topology optimization was done. This resulted in lighter structural components throughout the leg while maintaining the strength required to withstand impacts during dynamic motions.

This makes ARTEMIS a one-of-a-kind platform in terms of design principles combined

Parameters	ARTEMIS	Atlas	TORO	LOLA	HRP-5P	THOR	Walkman	Digit
Leg DoF	5	6	6	7	6	6	6	6
Arm DoF	4	7	6	3	8	8	7	4
Head DoF	2	0	2	3	2	2	2	0
Mass [kg]	36.8	85	76.4	68	101	65	102	48
Height [m]	1.42	1.5	1.74	1.83	1.83	1.78	1.85	1.6
Actuation Type	P	H	M	M	M	S	S	S

Table 2.2: Comparison of the different modern humanoids unveiled in the last decade. Leg DoF and Arm DoF are per each leg and arm respectively. Height is based on the reported heights from publications. In "Actuation Type," P is for proprioceptive actuation, S is for series elastic actuation or if any elastic component is a part of the design, H is hydraulic actuation, and M is for harmonic drives.

with actuation technology. A summary of the degrees of freedom and actuation differences compared to traditional and modern humanoids from Section 2.1 are shown in Section 2.2.1.

2.2.2 Sensors

Beyond the proprioceptive actuators, ARTEMIS is also equipped with inertial measurement units (IMU), force sensors, and exteroceptive sensors to assist with understanding the state of the robot, as seen in Figure 2.4. There is a tactical-grade IMU located at the pelvis as well as lower cost alternatives (LSM6DSO32) integrated at each foot providing linear acceleration and angular rate measurements with respect to each sensor's coordinate frames. The selection of the IMU is particularly important as it is used in the propagation step of estimating the state of the robot. We evaluated using VectorNav VN-100, Microstrain 3DM-GX5-AHRS, and Microstrain 3DM-CV7-AHRS as the main IMU to be installed at the robot's pelvis. Except for the latter, the first two are frequently used in the legged robotics community [BPK18, YHZ18, KDK19a, HAY20]. However, in our case, we eventually chose to go with the latter because of its superior performance as will be further detailed in 4.1.

Figure 2.4: Proprioceptive, exteroceptive, and IMU sensors located on ARTEMIS.

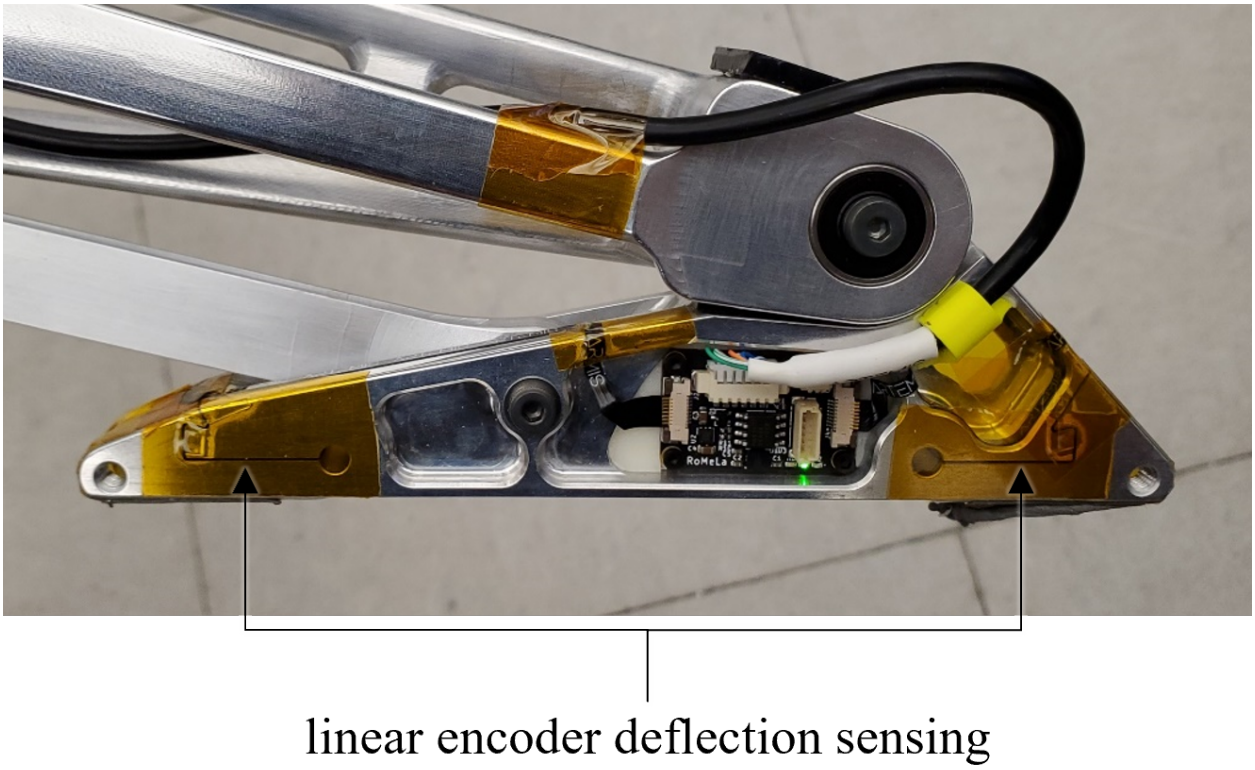

Figure 2.5: Contact sensors located at the toe and heel of ARTEMIS foot which works by using a linear encoder sensing deflection.

ARTEMIS also has 1-D force sensors located at the toe and heel to read force values that are perpendicular to the bottom of the foot, as seen in Figure 2.5. It is currently primarily used as a contact sensor where if the force reading goes beyond a pre-defined threshold, the system decides that contact has been made. However it also serves as a useful visualization tool for debugging to observe the center of pressure's location and force distribution at the feet during dynamic motions. For visualization purposes, the foot also has light-emitting diodes installed, where the intensity of the light reflects the magnitude of the force values.

Lastly, ARTEMIS also has multiple exteroceptive sensors to assist with localization, navigation, and state estimation. At the head is the ZED 2 stereo camera that can be used for localizing the robot in a known environment such as an indoor building or a soccer field which could later be used for path planning purposes. At the front and back of the pelvis

are Intel RealSense D435i depth cameras angled 45° with the ground [ACN19]. The position of the feet and the distance to the ground can be captured by the sensors, which will be used to remove the state estimation drift later discussed in Section 4.1. It can also be used for precise footstep planning to walk up stairs and foot collision avoidance in the future.

2.3 Modeling

2.3.1 Definition

Prior to conducting any kinematics and dynamics operations, it is necessary to define the coordinate axes for the links and the joints. Humanoids like ARTEMIS are a multi-rigid body system that can be represented with a rigid-body tree structure. As is standard with humanoids, ARTEMIS has five rigid-body chains when starting from its body frame. These are the two rigid-body chains for the legs, two for the arms, and one for the head. Their names and offsets from their parent links are shown in Tables 2.3 to 2.6. The coordinate axes are shown on a figure of ARTEMIS in Figure 2.6. For reference purposes, inertial parameters based on these definitions can be found in Appendix B.

2.3.2 Kinematics

Kinematics is solely the study of motion (i.e. position, velocity, and acceleration) without considering why something is moving in a certain way. Therefore, by knowing how the joints in a robot are moving, we can only know how the rigid bodies that make up the robot are moving. This section explains the fundamental kinematics operations done specifically for ARTEMIS.

2.3.2.1 Forward Kinematics

Forward kinematics is the computation of the positions and orientations of the frames attached to the rigid bodies of the robot using the joint positions of the kinematic chain. Often

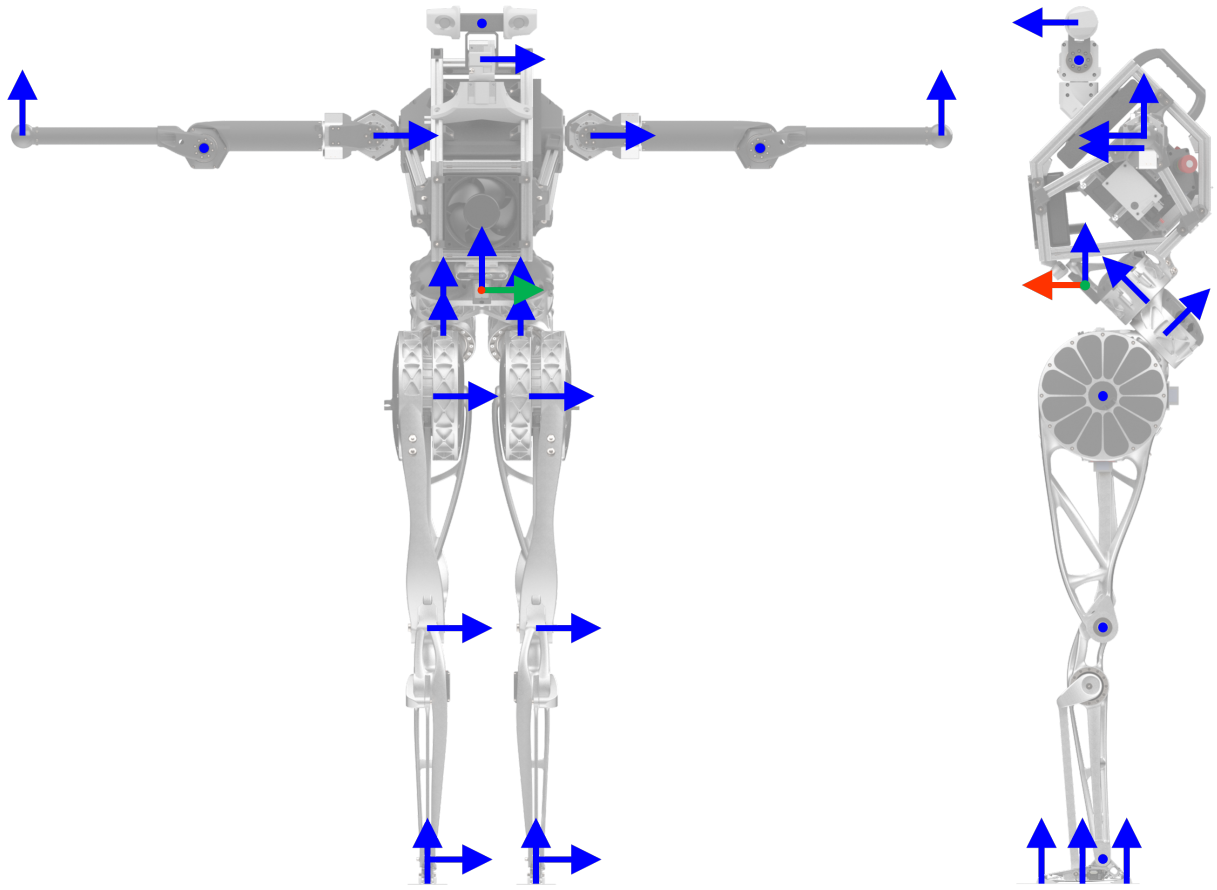

Figure 2.6: Definitions for the frames ARTEMIS uses in its kinematics, dynamics, and state estimation. The Z axis for each joints are shown and the X and Y are hidden for clarity.

times this frame of interest is the frame representing the last joint in the kinematic chain or any point that may interact with the environment. For example, in the case with ARTEMIS, our primary frames of interest for the feet are the mid-point along the line foot (Left/Right Sole from Table 2.3) and its two tips (Left/Right Toe and Heel from Table 2.3).

Using the standard forward kinematics by representing the joint offsets as DH parameters [Cra06] or in exponential coordinates [LP17], we can find the position and orientation of each of the frame with respect to the body. However, because ARTEMIS is a mobile robot, it is not a fixed-base robot but a floating base robot. Therefore, an additional transformation representing the robot body frame with respect to the world/inertial frame can be applied to

Frame Name	Parent Frame	Offset	
		Translational [m]	**Rotational [rad]**
Hip	Body	[-0.1, ±0.0625, -0.025]	[0, 0.7854, 0]
Adductor	Hip	[0.0135, 0, -0.054]	[0, -1.5708, 0]
Femur	Adductor	[0, ±0.008, -0.1455]	[1.5708, 0.7854, ∓3.1416]
Tibia	Femur	[0.375, 0, ±0.01735]	[0, 0, 0]
Foot	Tibia	[0.375, 0, 0]	[0, 0, 0]
Sole	Foot	[0.04, -0.03, 0]	[1.5708, 0, -1.5708]
Toe	Sole	[0.07, 0, 0]	[0, 0, 0]
Heel	Sole	[-0.07, 0, 0]	[0, 0, 0]

Table 2.3: Frame names and offsets for the left and right leg. Any number with ± or ∓ corresponds to the left and right leg respectively.

the existing frames on the robot to represent all these frames with respect to the world. This is particularly important and is later useful as all the planning and control is done in the world frame. The transformation from the world frame to the robot body frame is provided by an estimator that will be later discussed in Section 4.1.

2.3.2.2 Time Variation of Kinematics

Provided forward kinematics, it is also possible to compute the change in the position of frames of interest given the joint positions $\mathbf{q} \in \mathbb{R}^{n_u}$ and the joint velocities $\dot{\mathbf{q}} \in \mathbb{R}^{n_u}$. Recalling the forward kinematics

$$\mathbf{x}(t) = \texttt{FK(q)},$$

Frame Name	Parent Frame	Offset	
		Translational [m]	**Rotational [rad]**
Clavicle	Body	[-0.092, 0.175, 0.245]	[-1.5708, 0, 0]
Scapula	Clavicle	[0, 0, 0]	[3.1416, -1.5708, 0]
Upper Arm	Scapula	[0, 0, 0]	[-3.1416, -1.5708, 0]
Fore Arm	Upper Arm	[0, 0.02, 0.275]	[3.1416, -1.5708, 0]
Hand	Fore Arm	[0.292, 0.02, 0]	[0, -1.5708, -1.5708]

Table 2.4: Frame names and offsets for the left arm.

Frame Name	Parent Frame	Offset	
		Translational [m]	**Rotational [rad]**
Clavicle	Body	[-0.092, -0.175, 0.245]	[-1.5708, 0, 0]
Scapula	Clavicle	[0, 0, 0]	[3.1416, -1.5708, 0]
Upper Arm	Scapula	[0, 0, 0]	[-3.1416, -1.5708, 0]
Fore Arm	Upper Arm	[0, 0.02, -0.275]	[0, 1.5708, 0]
Hand	Fore Arm	[0.292, -0.02, 0]	[0, -1.5708, 1.5708]

Table 2.5: Frame names and offsets for the right arm.

using the chain rule, we can find the time variation of `FK()` as follows.

$$\begin{aligned}\dot{\mathbf{x}} &= \frac{\partial \texttt{FK}(\mathbf{q})}{\partial \mathbf{q}} \frac{d\mathbf{q}}{dt} \\ &= \frac{\partial \texttt{FK}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}} \\ &= J(\mathbf{q})\dot{\mathbf{q}}.\end{aligned}$$

From the above, we can see that $J(\mathbf{q}) \in \mathbb{R}^{6\times n_u}$ is the Jacobian, a function of joint positions $\mathbf{q}$, mapping the joint velocities $\dot{\mathbf{q}}$ to a frame of interest's linear and angular velocities. n_u is the number of actuated joints. If we were to include the floating base information as well, $J(\mathbf{q}) \in \mathbb{R}^{6\times n_v}$ where n_v is the sum of the number of actuated joints and the dimension of the linear and angular velocity (i.e. 6).

Frame Name	Parent Frame	Offset	
		Translational [m]	Rotational [rad]
Neck Bottom	Body	[0.0127, 0. 0.368]	[0, 0, 0]
Neck Top	Neck Bottom	[0, 0, 0]	[-1.5708, 0, 0]
Head	Neck Top	[0, -0.06, 0]	[0, 1.5708, 0]

Table 2.6: Frame names and offsets for the neck.

2.3.2.3 Inverse Kinematics

Inverse kinematics, as the nomenclature suggests, is the reverse of forward kinematics and is the problem of finding the joint positions $\mathbf{q}$ given a desired frame of interest's transformation $\mathbf{T}(\mathbf{q}) \in SE(3)$. This problem can sometimes be straightforward, as well as a challenge, depending on the redundancy of the kinematic chain and the ordering of the joints, as there may be infinitely many solutions and no analytical solution.

In the case with ARTEMIS, because the legs have 5 DoF and the arms have 4 DoF, it is not straightforward to find the joint positions. So rather than using an analytical solution, we use a numerical approach when trying to find the joint positions of the leg/arm given a desired $\mathbf{T}^d_{\texttt{BE}}$, where `B` is the body frame and `E` is the end-effector (i.e. foot or arm) frame. The numerical approach we use is a variation of the damped least squares (DLS) method. The pseudocode is shown in Algorithm 1.

2.3.3 Dynamics

Dynamics gives us insight into why things are moving as they are, under the influence of forces and torques. For a platform like ARTEMIS, controlling the robot using only its kinematics is a challenge as even for walking, the lack of a single DoF in the legs makes balancing a challenge. It cannot maintain balance purely using its ankles, but must use its entire body to stay balanced about the foot's roll axis.

Pseudocode 1 `artemis_model::get_single_chain_numerical_ik()`

Require: `E`, $\mathbf{T}_{\texttt{BE}}^{d}$, $\mathbf{q}_{\text{initial}}$, $\mathbf{W}$, n_{iter}, `dt`, ϵ, $\boldsymbol{\lambda}$

$\mathbf{q} \leftarrow \mathbf{q}_{\text{initial}}$
while `i` $< n_{\text{iter}}$ **do**
    $\mathbf{T}_{\texttt{BE}} \leftarrow \texttt{FK}(\mathbf{q}, \texttt{E})$
    `err` $\leftarrow \texttt{log6}(\mathbf{T}_{\texttt{BE}}, \mathbf{T}_{\texttt{BE}}^{d})$
    if `err` $< \epsilon$ **then**
        `success_stat` $\leftarrow$ `success`
        `break`
    end if
    if `i` $> n_{\text{iter}}$ **then**
        `success_stat` $\leftarrow$ `fail`
        `break`
    end if
    $\mathbf{J} \leftarrow \texttt{get\_jacobian}(\mathbf{q}, \texttt{E})$
    $\mathbf{JJT} \leftarrow \mathbf{WJJ}^{\top}\mathbf{W}^{\top} + \boldsymbol{\lambda}$
    $\mathbf{q} \leftarrow \mathbf{q} - \mathbf{WJ}^{\top}\mathbf{JJT}^{-1}(\mathbf{W}\texttt{err})\texttt{dt}$
    `i++`
end while
return $\mathbf{q}$, `success_stat`

We can define the well-known equations of motion or the manipulator equations as follows

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{Cg}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^{\top}\boldsymbol{\tau} + \mathbf{J}_{c}(\mathbf{q})^{\top}\mathbf{f} \tag{2.1}$$

but special attention must be given to the size of the generalized position $\mathbf{q} \in \mathbb{R}^{n_q}$, generalized velocity $\dot{\mathbf{q}} \in \mathbb{R}^{n_v}$, and generalized acceleration $\ddot{\mathbf{q}} \in \mathbb{R}^{n_v}$. This is because as a floating base system, the multi-body dynamics must also include the pose and velocities of the robot's body frame with respect to the world frame, just like how it was included for the kinematics. Then, Equation (2.1) consists of the following elements:

$\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n_v \times n_v}$: Mass matrix.

$\mathbf{Cg}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n_v}$: Nonlinear vector which is the sum of the centrifugal, Coriolis, and gravitational terms.

$\boldsymbol{\tau} \in \mathbb{R}^{n_v}$: Torques applied at the joints.

$\mathbf{J}_c(\mathbf{q}) \in \mathbb{R}^{(\dim \mathbf{f} \times n_c) \times n_v}$: Contact (Geometric) Jacobian.

$\mathbf{f} \in \mathbb{R}^{\dim \mathbf{f} \times n_c}$: External forces due to contact with the environment.

$\mathbf{S} \in \mathbb{R}^{n_u \times n_v}$: Selection matrix to select the actuated torques.

The generalized coordinates of a floating base system consist of actuated joints $\mathbf{q}_a \in \mathbb{R}^{n_u}$ and unactuated joints $\mathbf{q}_u \in \mathbb{R}^6$. In the case with ARTEMIS, $\dot{\mathbf{q}}_a \in \mathbb{R}^{20}$, and the actuated joints affect Equation (2.1) through the selection matrix which is of the form:

$$\mathbf{S} := \begin{bmatrix} \mathbf{0}_{6\times 6} & \mathbf{I}_{n_v - 6} \end{bmatrix}.$$

To have control authority over the floating base joint (i.e. base link) of the robot, which is often the case with legged robots, reaction forces $\mathbf{f}$ are required. The dimension of this force ($\dim \mathbf{f}$) depends on the chosen contact model. In the case of a rectangular foot, this could be a 6 dimensional vector accounting for the reaction forces and moments at a single point or it could be four 3 dimensional forces at the corners of the rectangle. In the case with ARTEMIS which has a line foot and 5 DoF in the leg, the reaction force could be represented using three forces and two moments at a single point on the foot or with two 3 dimensional forces with one at the front (toe) and the other at the back (heel). The two contact models applicable to ARTEMIS are shown in Figure 2.7.

Naturally, the chosen contact representation decides the contact Jacobian. In the case the 6 dimensional force and moment representation is used, the complete Jacobian mapping the joint velocities to the linear and angular velocities are used for contact i as shown in Equation (2.2). Note that in literature, there are a lot of different ways to refer to the

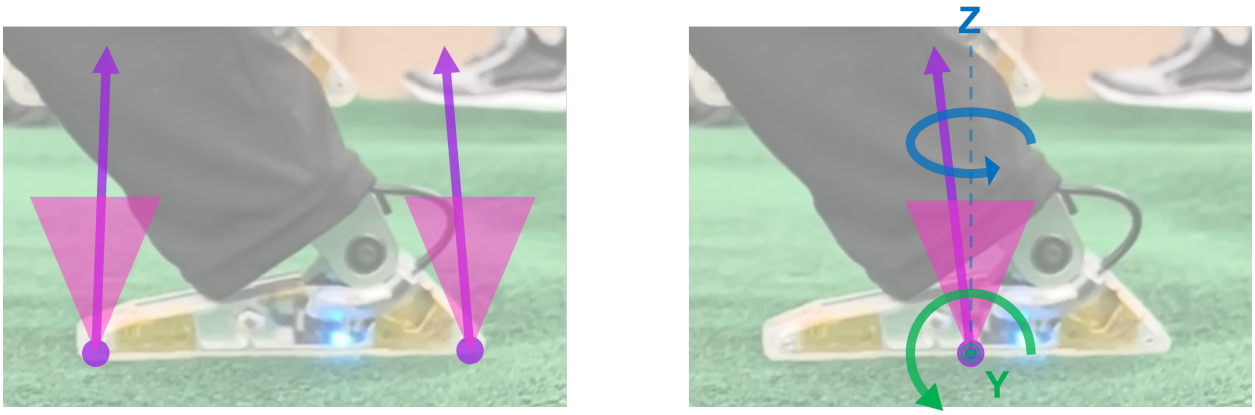

Figure 2.7: Potential contact models for ARTEMIS. Left: A three dimensional force applied at the toe and the heel. Right: A three dimensional force applied at the middle of the foot and a moment applied about the Y and the Z axis at the foot's frame.

same Jacobian. To clarify, the contact Jacobian is identical to the Jacobian used earlier in Section 2.3.2.2.

$$\mathbf{J}_{c,i} = \begin{bmatrix} \mathbf{J}_{v,i} \\ \mathbf{J}_{\omega,i} \end{bmatrix} \tag{2.2}$$

In most cases however, the Jacobian corresponding to the linear velocities in the world frame are used. Also in this dissertation, we are using 3 dimensional forces at the toe and the heel of the foot, which then, the contact Jacobian for the left foot and the right foot look like the following:

$$\begin{aligned} \mathbf{J}_{c,\mathtt{LF}} &= \begin{bmatrix} \mathbf{J}_{v,\mathtt{LF\_TOE}} \\ \mathbf{J}_{v,\mathtt{LF\_HEEL}} \end{bmatrix} \\ &= \mathbf{R}_{\mathtt{WB}} \begin{bmatrix} -\widehat{\mathbf{p}_{\mathtt{B,LF\_TOE}}} & \mathbf{I} & \mathbf{J}_{v,\mathtt{B,LF\_TOE}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\widehat{\mathbf{p}_{\mathtt{B,LF\_HEEL}}} & \mathbf{I} & \mathbf{J}_{v,\mathtt{B,LF\_HEEL}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned}$$

$$\mathbf{J}_{c,\mathtt{RF}} = \begin{bmatrix} \mathbf{J}_{v,\mathtt{RF\_TOE}} \\ \mathbf{J}_{v,\mathtt{RF\_HEEL}} \end{bmatrix}$$
$$= \mathbf{R}_{\mathtt{WB}} \begin{bmatrix} -\widehat{\mathbf{p}_{\mathtt{B,RF\_TOE}}} & \mathbf{I} & \mathbf{0} & \mathbf{J}_{v,\mathtt{B,RF\_TOE}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\widehat{\mathbf{p}_{\mathtt{B,RF\_HEEL}}} & \mathbf{I} & \mathbf{0} & \mathbf{J}_{v,\mathtt{B,RF\_HEEL}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Additionally, the component of the Jacobian mapping the joint velocities to the angular velocities are as follows, except in the foot frame instead of in the world frame:

$$\mathbf{J}_{\omega,\mathtt{LF}} = \mathbf{R}_{\mathtt{LF,B}} \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{J}_{\omega,\mathtt{B,LF}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{J}_{\omega,\mathtt{RF}} = \mathbf{R}_{\mathtt{RF,B}} \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{J}_{\omega,\mathtt{B,RF}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Lastly, to verify the correctness of the dynamics, simple sanity checks on the resulting equations of motion can be conducted. For floating base systems, one way to check for the correctness of the dynamics is by checking the relationship between the position of the system's CoM and the mass matrix. Depending on the reference frame and order of the linear and angular velocities, the exact elements in the inertia matrix to compare may change. However, if following the convention in this dissertation where the linear velocity comes before the angular velocity, and they are both in the body frame [Fea14], the center of mass can be calculated from the inertia matrix as follows

$$\mathbf{p}_{\mathtt{BRcm}} = \begin{bmatrix} \frac{\mathbf{M}(5,3)}{\mathbf{m}} & \frac{\mathbf{M}(6,1)}{\mathbf{m}} & \frac{\mathbf{M}(4,2)}{\mathbf{m}} \end{bmatrix}$$
$$\mathbf{p}_{\mathtt{WRcm}} = \mathbf{R}_{\mathtt{WB}}\mathbf{p}_{\mathtt{BRcm}} + \mathbf{p}_{\mathtt{WB}}$$

where $\mathbf{m}$ is the mass of the robot. A quick, manual computation of the center of mass and comparing it with the above with respect to the world frame is a good sanity check.

CHAPTER 3

ARTEMIS Software and Control

As the ultimate integration challenge, humanoids require a strong combination of hardware and software to realize the desired behaviors on the physical robot. This chapter explains the software and locomotion control stack behind ARTEMIS. Because of the complexity of a system like a humanoid, it is even more important for the software stack to be modular, reusable, and have clear interfaces to be extendable. This allows components to be more easily replaced with different approaches in the future, simplifies creating different combinations of controllers, and allows incremental testing of the system from component level to the entire stack. Then, we should be able to more easily generate different motions and behaviors while being able to conveniently reuse many of the existing functionalities that are integral to running a humanoid robot. An end-user integrating their specific functionality can focus on their contribution and extensively test it without concerns of a potential issue in a component other than theirs. With that in mind, the ARTEMIS software stack is architected such that different hardware and controllers can be easily swapped.

3.1 ARTEMIS Software Suite

The software stack follows a similar approach to those deployed in [YHZ18, AH20, HAY20] where the same stack was used in both simulation and on the actual hardware. An overview of it is shown in Figure 3.1. It is primarily divided up into a hardware/simulation interface, the camera interface, the controller interface, and the safety interface. These different components need to be run concurrently making multithreading an integral feature in the software stack for algorithmic reasons as well as to ensure a distributed load across the com-

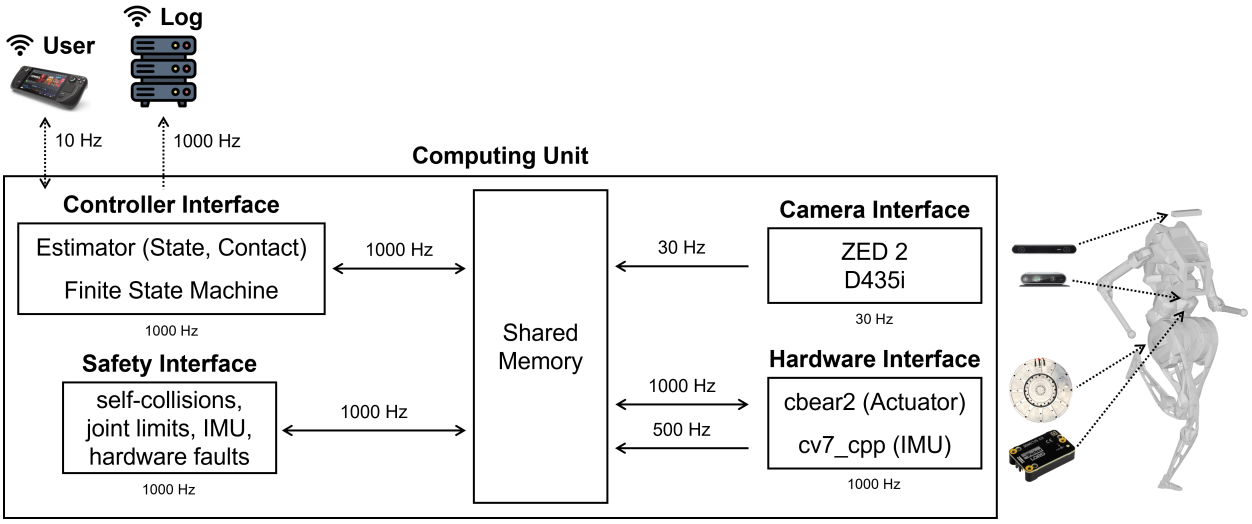

Figure 3.1: An overview of the software architecture running on ARTEMIS.

puter's central processing unit (CPU). Consequently, shared memory and semaphores are largely used to manage the data across the different concurrently running processes. All data are logged to an off-board computer for analysis.

3.1.1 Hardware Interface

3.1.1.1 Actuators

The hardware interface is composed of the different objects that expose communication with the hardware and are essential to the operation of the robot. This includes the actuators, the inertial measurement unit, and the contact sensors. A custom Software Development Kit (SDK) for communicating with the actuators is used to send commands over two RS-485 chains [Ahna]. One chain is used to communicate with the lower body actuators (10 actuators) and the contact sensor (one per foot), and another chain is used to communicate with the upper body actuators (8 from the arms and 2 at the neck). These two chains are concurrently run to achieve 1,000 Hz communication frequency which is important for stable torque control. All commands to both communication chains consist of desired joint

positions, velocities, and torques, except for the contact sensor which does not receive any commands. The returned packet contains current joint position, velocity, and torque information, as well as an error status flag that contains useful information about the actuators (e.g. joint limit violation on the firmware side, overheating). The contact sensors' readings at the toe and the heel are mapped to the joint position and velocity registrars to enable convenient reading of the same registrars across the entire RS-485 chain.

To summarize, the actuator hardware interface reads joint commands in the shared memory, which are updated by the controller interface. It then sends those commands to the actuators through the custom SDK. The returned values (current joint state and contact sensor readings) are written in the shared memory by the hardware interface and that information is consumed by the controller interface, the safe interfaces, and the loggers.

3.1.1.2 Inertial Measurement Unit

The interface with the inertial measurement unit also works in a similar manner where a separate thread calls on a custom IMU object built around Microstrain's official SDK to minimize overheads [Ahnb]. The 3DM-CV7-AHRS is capable of providing raw (sensed) acceleration and angular rates from the accelerometer and the gyroscope at 1,000 Hz, but we instead sample the filtered acceleration and angular rates from 3DM-CV7-AHRS's built-in estimation filter at 500 Hz. This is to allow the robot state estimator to not have to explicitly estimate the acceleration and angular rate biases, which will be further explained in Section 4.1. These acceleration and angular rate data get stored at its corresponding shared memory segments to be consumed by the robot's state estimator in the controller interface as well as by the safety triggers in the safety interface.

3.1.2 Simulation Interface

The simulation interface tries to mimic the functionalities in the actuator interface and the IMU interface. Desired position, velocity, and torque commands can be sent to the simulation environment in an identical fashion to the physical hardware. The IMU's noise parameters

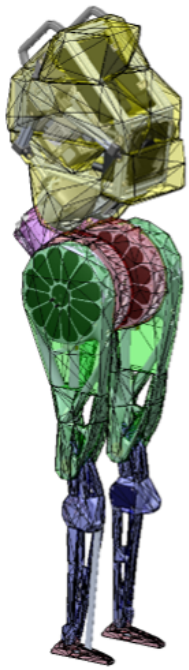

Figure 3.2: A convex decomposed collision and visual model overlayed on top of the detailed robot model.

are based off of the physical IMU's datasheet such that the same parameters for the floating base state estimator can be used in both simulation and hardware. Currently, the supported simulators are Gazebo [KH04], MuJoCo [TET12], and Pybullet [CB19], which the user can pick based on their preference as well as to test across different environments prior to testing on the hardware. The robot model uses the inertial parameters from Appendix B.

To reduce the complexity of the simulation, we refrain from using the detailed model exported directly from CAD but also avoid primitive shapes such as cylinders, capsules, or even convex hulls, because of ARTEMIS' unique design generated by the topology optimization. Instead, we use an approximate convex decomposition using V-HACD [MLP16] as seen in Figure 3.2 (full model) and Figure 3.3 (femur) for both visualization and collision, which is a nice middle ground between the detailed model and the convex hull. More explanation on the decomposition's role in collision detection is available in Section 3.1.4. Additionally, unlike in reality, the user can step through the simulation and the controller at a desired step time to assist with debugging.

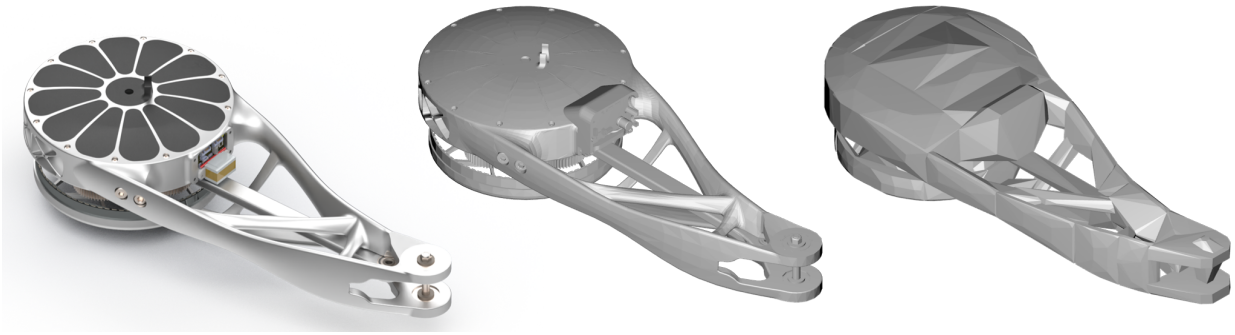

Figure 3.3: Left: Original model from the CAD. Middle: A simplified model using Meshlab [CCC08]. Right: A convex decomposed model using V-HACD. [MLP16]

3.1.3 Controller Interface

The controller interface is responsible for reading the current joint states and IMU readings from shared memory and updating the contact detector and the state estimator which will be consumed by controllers such as the locomotion controller. Estimating the floating base's pose and velocities are an essential part of legged robot control as explained in Section 2.3. This requires the robot's frames that are in contact (from the contact detector) and the corresponding kinematics information (derived using the joint states). Once the estimator predicts the floating base information, it is concatenated with the joint states to be passed to the finite state machine's current state. Custom user controllers reside in their own custom state and if the state is the current state in the finite state machine, it can consume the generalized position and velocities to compute the desired joint states to be sent back to the hardware interface. The robot state estimator and the user controller runs on a separate thread to the main controller interface which updates the finite state machine at 1000 Hz. The pseudocode of the controller interface is shown in Algorithm 2.

Pseudocode 2 `artemis_controller_interface::update()`

```
1: q, qdot ← get_joint_states()
2: contacts ← get_contact_status(q, qdot, tau)
3: update_state_estimator(q, qdot, contacts)
4: q_desired, qdot_desired, tau_desired ← update_fsm(args)
5: send_joint_commands_to_hardware(q_desired, qdot_desired, tau_desired)
```

3.1.4 Safety Interface

Lastly, while the hardware interface is communicating with the actuators, a safety interface runs on a separate process to shutdown the robot in the case that any erroneous behaviors are detected. A practical difficulty with adult-sized torque controlled humanoids using proprioceptive actuators is that because their joint velocities can be very high, it is difficult for a human to react fast enough and intervene before the robot does any damage to its surroundings or itself. Therefore, a safety interface was designed to run at 1,000 Hz, checking the following to see if any thresholds are triggered to stop the robot:

- **Self-collision**

  Since the joint positions dictate the robot's posture, by checking the joint positions at every hardware communication, we can detect whether a self-collision has occurred. This is achieved by using a simplified convex decomposition of the collision mesh of the robot and the Gilbert-Johnson-Keerthi distance algorithm [GJK88]. As powerful as modern computers are, checking every collision combination between one link against another is computationally intensive and also unnecessary. For example, the left foot will likely never come in contact with the right femur, although it may come in contact with the right foot or the right tibia. In this case we remove the pair left foot, right femur from collision checks. For the complete list of collision combinations that are ignored, refer to Appendix A.

- **Joint Velocity Limits**

  From the motion that the robot will execute, we know the joint velocities that the

joints should be moving at. For example, the joint velocities during regular walking is drastically different to when the robot is trying to do a jump with significant air time. Based on the desired joint velocities, if a joint is moving faster than the desired velocities by a pre-defined percentage, or if it exceeds a preset velocity threshold, the robot is also assumed to be erroneously behaving and is halted for safety. Other joint related limits include joint position limits and joint torque limits. They also can be triggered when violated, but are set in the actuators' firmware, thus they are not explicitly a part of the safety interface. Their values can be seen in Appendix C.

- **Acceleration and Angular Rates**

  A similar logic is applied to halt the robot in the case that acceleration or angular rate values directly from the IMU experience erroneous values and reach a predefined threshold. This is because there could be legitimate motions that require high acceleration or angular rates, such as a turning jump. Otherwise, anytime the robot suddenly jumps up, tugs on the gantry, or even jerks in place (usually a result of extremely high torques being commanded), this portion of the safety interface should catch this erroneous behavior and stop the robot (usually a result of extremely high torques being commanded).

- **Joint Oscillations** An unstable input will often result in high frequency oscillations of the system, which is undesirable for the hardware to repeatedly experience. The vibrations can result in the general wear and tear of the hardware over time which can further induce unexpected slips in the system. To prevent this, we repeatedly check for high frequency oscillations in the joint states (position, velocity, torque) and halt the robot in the case that any of the oscillations are above a pre-defined threshold.

- **Error Code Checks**

  The actuators constantly append an error status at the end of the return packet. These errors can signify `NO_ERROR`, but they can also have a non-zero value which could signal erros such as joint limits and overheating. If any of the actuators return an unexpected packet, the entire robot is forced into a halt state.

If any of the safety checks are triggered, the robot's state machine is set to the `HALT_DAMPING` state which switches the actuators to a damping mode where every joint of the actuator behaves as if it was extremely damped.

Using this software architecture, the next topic to be presented is the dynamic locomotion stack that controls ARTEMIS. This controller resides in a single state machine within the architecture's finite state machine.

3.2 Dynamic Locomotion Stack

For dynamic locomotion in outdoor environments, which could have uneven and discontinuous terrains, it is important for the locomotion controller to be able to adapt its plans and trajectories immediately (i.e. in real-time). To make these strategic decisions on-the-fly, the robot must be able to:

1. Sense its contact status with the environment.
2. Estimate its base frame's pose and velocities.
3. Plan *when* to move its end-effectors.
4. Plan *where* to move its end-effectors.
5. Plan *how* to move its end-effectors, center of mass, and body.

The overall scheme of the locomotion framework is shown in Figure 3.4. In this chapter, the above topics are presented to explain the integrated stack that will test the dynamic locomotion capabilities of ARTEMIS.

3.2.1 Contact Estimator

In a perfect environment where the exact dimensions of the world are known and the robot's relative position and control is also perfect, upcoming collisions with the environment as the

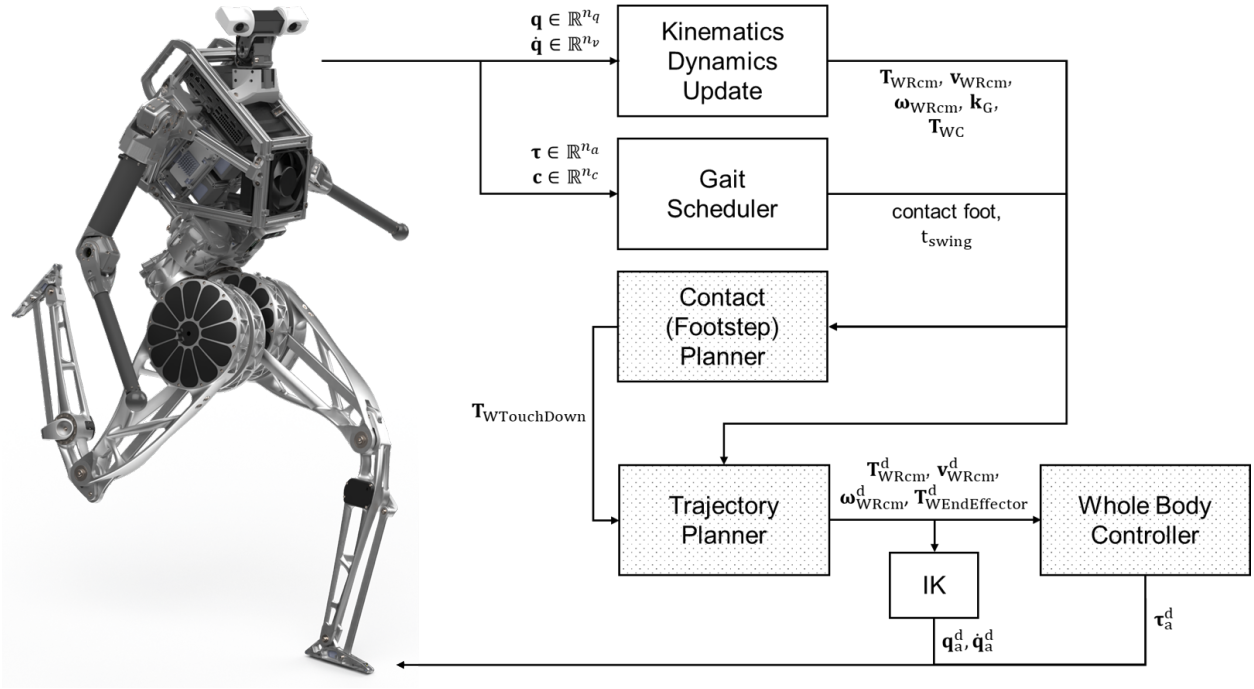

Figure 3.4: Overview of the locomotion framework. The locomotion controller takes in the current generalized coordinates $(\mathbf{q}, \dot{\mathbf{q}})$, torques $(\boldsymbol{\tau})$ and contact statuses $(\mathbf{c})$, and computes a desired feedforward torque $(\boldsymbol{\tau})$ and joint PD. Gait Scheduler plans *when* to move the end–effectors. Contact (Footstep) Planner decides *where* to move the end-effectors. Trajectory Planner, IK, and Whole Body Controller decides *how* to move the end-effectors, center of mass, and body.

robot moves can be accurately predicted. However this is an inappropriate assumption as joint control can be imperfect, the state of the robot can be inaccurate, but more importantly the environment cannot be perfectly modeled beforehand. The environment a robot encounters at this current instance may be altered even just seconds later.

As humans, we use a combination of different senses (e.g. touch and perception) to navigate around uncertainties in the environment. In the case of walking on flat ground, even without perception data, we expect roughly when contact will occur based on our previous walking cadence and shift our body mass to the point of contact once contact has been made. However, we all have experienced moments where we thought contact was going to happen, but it doesn't. If we shift our mass when support has not been established, we fall or stumble.

In the case with ARTEMIS, the force sensor on the foot as seen in Figure 2.5 can act as a contact sensor to indicate that contact has been made at the foot. While this is applicable for the foot and can be used, we also use a contact estimator for the following reasons:

1. A foot sensor is a physical component that is one of the most susceptible points of failure as it is closest to the part of the robot that comes in repeated impact. This can result in the sensor returning erroneous values over time.

2. It is a practical challenge to embed force/contact sensors everywhere that a contact could occur. If the robot wanted to detect contact at the knees in scenarios such as during a standup sequence where the robot could support itself through contact at its toes and the knees, it would be required to embed another sensor at the knees. This is not just limited to locomotion, but even for manipulation to touch and detect collision between the hands and an object or an environment, embedding sensors in multiple points of contact can be a practical design challenge.

The overview of the contact estimator on ARTEMIS is as follows. On ARTEMIS, we calculate the residual contact force by using the current torques and the expected torques. Afterwards, a probabilistic function is used as a threshold to determine whether or not a con-

tact has been made. Additionally, depending on the "phase" of the motion, the probabilistic function is modified such that contact can be estimated as early as possible.

The current torque readings are readily available from the actuators, but the expected torques are found using the inverse dynamics (i.e. Recursive Newton Euler Algorithm). In the computation of the inverse dynamics, we consider the full floating base dynamics of the robot rather than just the leg dynamics. The product of the error in the torques and the contact Jacobian is used to compute the residual force

$$\mathbf{F}_{\text{res}} = -(\mathbf{J}_c^\top)^{-1}(\boldsymbol{\tau} - \widehat{\boldsymbol{\tau}}) \tag{3.1}$$

where $\boldsymbol{\tau}$ is the vector of current joint torques, $\widehat{\boldsymbol{\tau}}$ is the vector of estimated joint torques from inverse dynamics, and $\mathbf{J}_c$ is the contact (geometric) Jacobian to the frames of interest (e.g. toe/heel of the foot, knee, hands), as defined in Equation (2.2). Note the dimensions of the vectors and the matrices. The torques $\boldsymbol{\tau} \in \mathbb{R}^{n_j'}$ where n_j' is the number of actuated joints to the frame of interest, $\mathbf{J}_c \in \mathbb{R}^{3\times n_j'}$ and $\mathbf{F}_{\text{res}} \in \mathbb{R}^3$. For example, to detect contact on the bottom of the left foot, the current and estimated torques are $\boldsymbol{\tau} \in \mathbb{R}^5$ and $\widehat{\boldsymbol{\tau}} \in \mathbb{R}^5$, the contact Jacobian is $\mathbf{J}_c \in \mathbb{R}^{3\times 5}$, and $\mathbf{F}_{\text{res}} \in \mathbb{R}^3$.

Afterwards, a residual force along the perpendicular direction to the contact surface is computed

$$F_{\text{res},\widehat{\mathbf{n}}} = \mathbf{F}_{\text{res}} \cdot \widehat{\mathbf{n}}_c \tag{3.2}$$

where $\widehat{\mathbf{n}}_c \in \mathbb{R}^3$ is the unit vector perpendicular to the contact surface. If walking on flat ground, $\widehat{\mathbf{n}}_c = \mathbf{e}_z^{\mathtt{W}}$, while for other surfaces, this information could come from perception data.

Finally, the estimated normal force is evaluated in an adaptive logistic function to determine whether contact has been made. The adaptive logistic function uses the general logistic function $p(x) : \mathbb{R} \to (0, 1)$:

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \tag{3.3}$$

except the explanatory variables β_0 and β_1 are modulated based on a user-input. In the case of locomotion, we modify the explanatory variables based on the phase of walking. When the swing leg is lifting off the ground or is imminent of touchdown, we want the estimator

to be relatively more sensitive to the residual normal contact force compared to when it is midway through the swing phase, where a contact is less likely. To embed this behavior, we use the following heuristic to modify the explanatory variables.

$$\beta_0 = \beta_{0,\text{nom}} + \gamma_0(1 + \sin(\pi(s+1))) \tag{3.4}$$

$$\beta_1 = \beta_{1,\text{nom}} + \gamma_1(1 + \sin(\pi(s+1))) \tag{3.5}$$

where $\beta_{0,\text{nom}}$ and $\beta_{1,\text{nom}}$ are identified nominal values, γ_0 and γ_1 are non-negative parameters where the greater the values, the more sensitive the estimator will be near lift-off and touch-down, and $s : \mathbb{R} \to [0, 1]$ is the phase variable where when $s = 0$, the foot is just starting its swing sequence and when $s = 1$, the foot is just ending its swing sequence. If γ_0 or γ_1 is set to 0, the adaptive behavior can be turned off. Finally, when $p(x)$ goes beyond a pre-defined threshold c, contact is assumed to be made. Figure 3.5 shows a comparison of the different β_0, β_1 values depending on the phase s of the swing trajectory. The contact is triggered at a lower threshold when closer to the start or end of the phase.

The approach is run concurrently for all frames that contact is expected to potentially happen. Its steps are detailed in Algorithm 3.

Pseudocode 3 `get_estimated_contact_status()`

Require: $\boldsymbol{\tau}$, $\widehat{\boldsymbol{\tau}}$, $\mathbf{J}_c$, $\widehat{\mathbf{n}}_c$, s, c

$F_{\text{res},\widehat{\mathbf{n}}} \leftarrow -(\mathbf{J}_c^\top)^{-1}(\boldsymbol{\tau} - \widehat{\boldsymbol{\tau}}) \cdot \widehat{\mathbf{n}}_c$

$p(x) \leftarrow$ `adaptive_logistic_regression`(F_{res})

if $p(x) \geq c$ **then**

    return `TRUE` ▷ Contact.

else

    return `FALSE` ▷ No contact.

end if

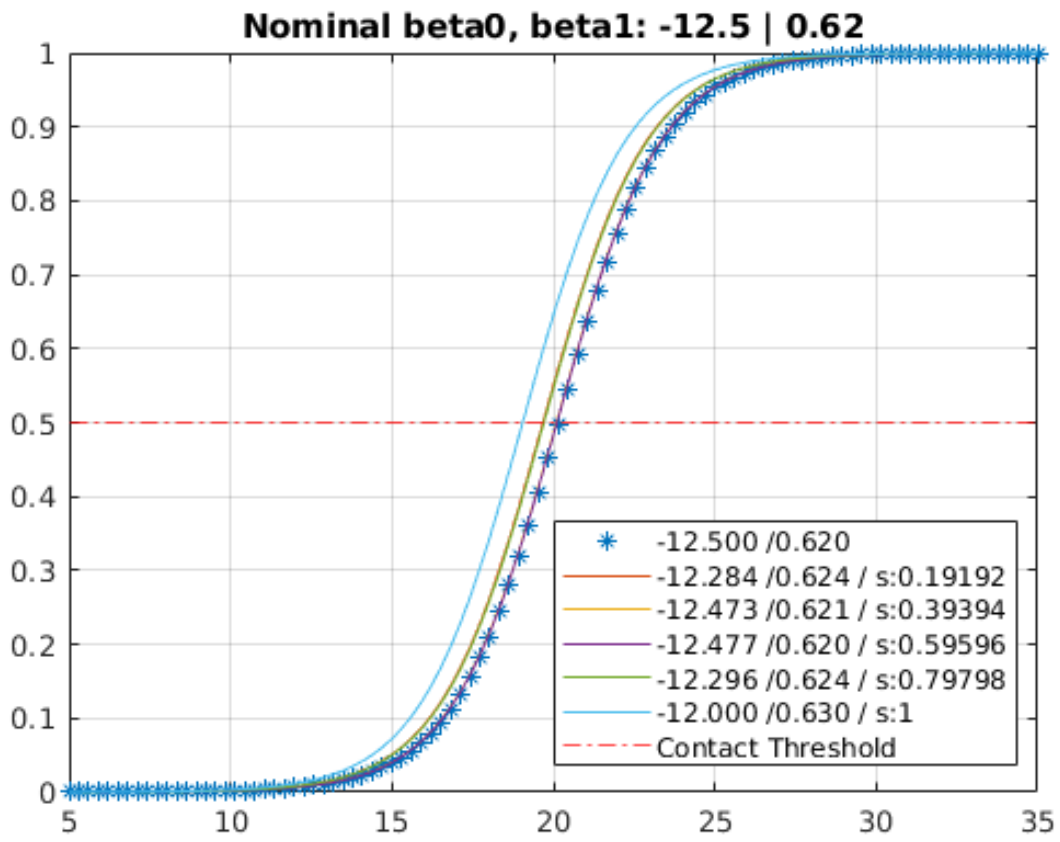

Figure 3.5: A comparison of how the contact is assumed depending on the phase of the swing trajectory.

3.2.2 Gait for Walking and Running

The first important component of the dynamic locomotion stack is choosing not only when to move a foot, but also how to coordinate their movements. This can easily be done by defining a "step" in a non-conventional way, assuming that there are more than a single foot on the robot.

Often a "step" is defined as a consecutive sequence of a stance (contact) phase followed by a swing (no contact) phase. A "phase" in this context is defined as a value from $[0, 1]$ and is indicative of how much percentage a stance or a swing sequence has elapsed. A stance

time and a swing time defines the timings of a step. For example, if we desired a swing time of 0.4 s and 0.2 s elapsed since the start of the swing time:

1. The swing phase would be 0.5 since $\frac{0.2}{0.4} = 0.5$.
2. The stance phase would be 1.0 since swing phase is greater than 0.0 and by definition a swing sequence always follows a stance sequence.

This definition is universal and applies even if you have a monopod hopper.

If we assume at least two legs as in the case with a humanoid, we can define a "step timing" in a non-conventional way to also naturally introduce flight phases which are required for running. Rather than defining the step timings as a stance time and a swing time, we could also define it with:

1. Swing Time: The amount of time the foot should be in the air for assuming ideal conditions.
2. Lift-Off Percentage: The percentage of Leg B's swing phase when Leg A should transition from stance to swing.

The above change in the definition when there are at least two legs can be better understood through a few examples.

1. Swing Time 0.4, Lift-off Percentage 1.0: When the lift-off percentage is set to 1.0, it implies that the leg that is in stance should only move to swing phase once the current swing leg reaches the end of its swing phase. This results in a desired gait where there only is a single support phase at all times and no double support phase or flight phase. Figure 3.6 shows a figurative representation of the desired stance and swing timings of the left and the right foot with this swing time and lift-off percentage. It is noticeable that there are no times when it is desirable for both feet to be off the ground and in swing phase.

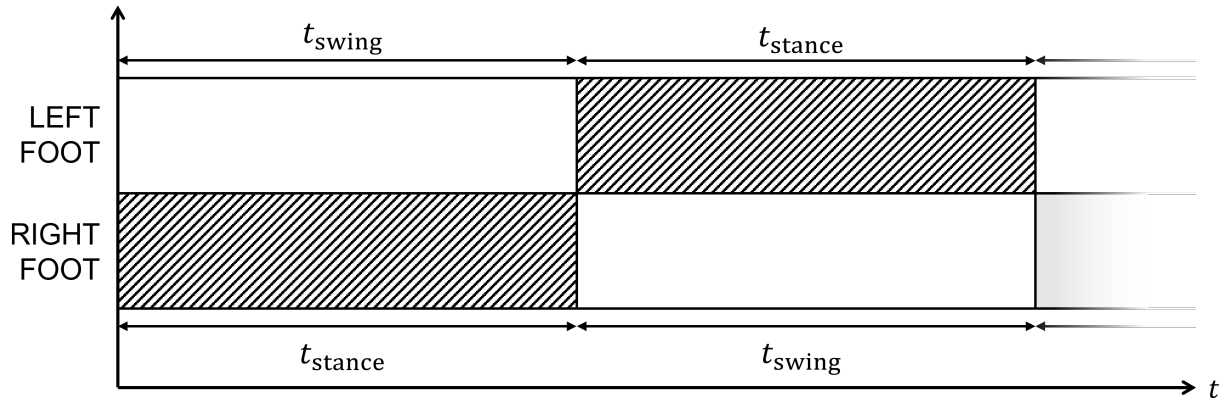

Figure 3.6: Desired stance and swing timings for the left and right foot when the lift-off percentage is 100%. The horizontal axis is time and the vertical axis indicates which foot. The lined intervals indicate stance and the empty intervals indicate swing.

2. Swing Time 0.4, Lift-off Percentage 0.7: When the lift-off percentage is set to 0.7, it means that the current leg that is in stance will transition from stance to swing when the current swing leg has been swinging for $0.7 \times 0.4 = 0.21$ seconds. Figure 3.7 shows a diagram of the desired stance and swing timings of the left and the right foot. Compared to Figure 3.6, we can notice that there are time intervals (highlighted in green) when both feet are desired to be in swing phase, which would result in wanting the robot to be in some sort of a flight phase. The desired flight duration is then given by the following equation:

$$T_{\text{flight}} = T_{\text{swing}}(1.0 - LOP) \tag{3.6}$$

where T_{swing} is the desired swing time and LOP is the lift-off percentage. So for our example, the desired flight time would be 0.12 s. Therefore, this informs us that any lift-off percentage below 1.0 will result in some desired non-zero flight time.

Provided this *when* contact should happen information, corresponding *how* information on the center of mass trajectory along the Z direction will be generated. Therefore, by

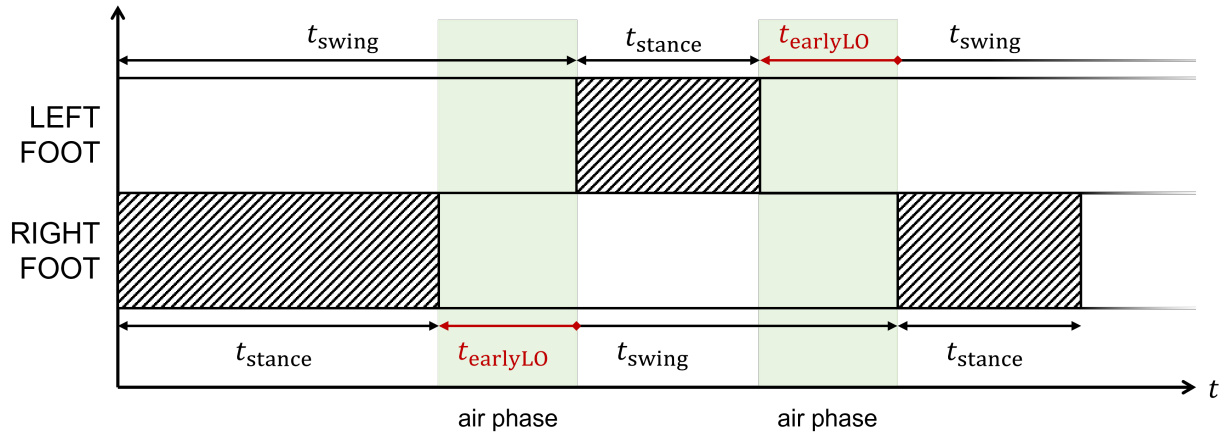

Figure 3.7: Desired stance and swing timings for the left and right foot when the lift-off percentage is 70%. The horizontal axis is time and the vertical axis indicates which foot. The lined intervals indicate stance, the empty intervals indicate swing, and the green highlighted intervals indicate flight phases.

simply modifying the lift-off percentage from 1.0 to some positive value less than 1.0, we can transition from desired contact timings that would correspond to walking to those for running.

3.2.3 Footstep Planner

Recall that for an underactuated platform like ARTEMIS which has 5 DoF per leg, maintaining balance on a single line foot is a non-trivial task. Previous position-controlled humanoids using approaches such as ZMP-based preview control [KKK03] are able to demonstrate static walking as the legs are fully actuated (i.e. 6 DoF) and their feet create a two-dimensional support polygon for the robot to maintain its center of mass inside it as demonstrated in Figure 3.8.

When a support polygon does not exist, the only way for a robot to not fall down is strategically positioning its feet at locations that it can "catch" itself from falling down. In this regard, the robot is dynamically staying balanced and if the actuators were to suddenly

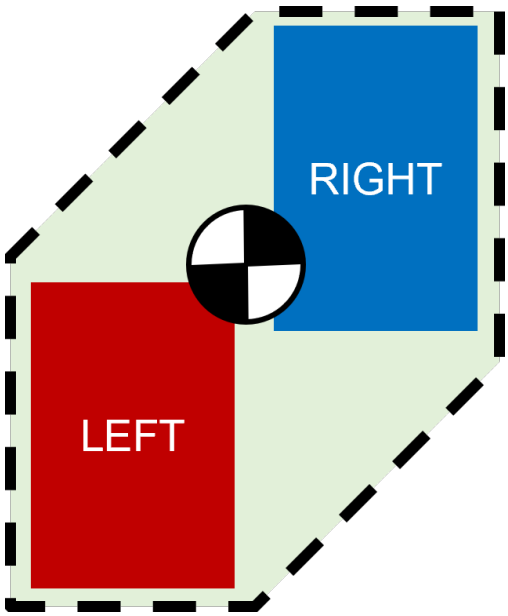

Figure 3.8: If the center of mass is inside the support polygon (highlighted in green) defined as the convex hull of all feet in contact, the robot will not fall down.

stop moving (i.e. maintain its position), the robot would fall down as the center of mass is outside its support polygon. Different approaches exist, but the more widely used approaches are Capture Point [PCD06] based approaches and Raibert heuristics [RBC84, Rai86].

In the biped and humanoid community, footstep planning based on Capture Point and Divergent Component of Motion have been actively researched and successfully used. The idea has stood the test of time as since its initial introduction [PCD06], the concept is still actively used, whether it is to come to an immediate step or to analyze a controller's ability to come to a stop in N-steps [KDR12, She22].

Raibert heuristics has especially been popular in the quadruped community to demonstrate highly dynamic locomotion [KDK19b, BPK18, HAY20], but has not been as much in the humanoid community [AH20]. It was originally developed as part of a controller for a single legged hopping machine. This controller can be decomposed into three parts where one of them was the footstep planner which controlled the forward velocity of the robot. By strategically placing the foot with respect to the center of mass projected on to the contact surface, the hopping machine could control its forward velocity. Recall how a robot, when

its foot is in contact with the ground, behaves like an inverted pendulum. From everyday experience, we know that we can keep the pendulum balanced by manipulating the position of the base with respect to the center of mass. If the base position is directly below the center of mass, the pendulum will stay balanced. If it strays away, it will start to accelerate. Similarly, for a legged robot in contact with the ground, if its contact point is not located directly below its center of mass, the robot will either accelerate or decelerate. Raibert recognized that acceleration is a function of the displacement between the center of mass and the contact point. Therefore, if the center of mass were to pass over the contact point midway through the stance phase, the accelerations (i.e. the moments induced by the mass away from the contact point) before the midpoint and after it would cancel each other out. If the symmetry is perfect, no acceleration would be produced and the robot would travel at a constant velocity. The body will accelerate if the center of mass spends more time in front of the contact point and will decelerate when it is more behind the contact point. Therefore, an error between the desired and current velocity was used to adjust the position away from the foot position that would provide symmetry. This resulted in the following equation:

$$p_x^d = \frac{\dot{p}_x T_{\text{stance}}}{2} + K(\dot{p}_x - \dot{p}_x^d) \tag{3.7}$$

where p_x^d is the desired position of the contact point, $\dot{p}_x$ is the current linear velocity of the center of mass, $\dot{p}_x^d$ is the desired linear velocity of the center of mass, and T_{stance} is the stance time.

One thing to note of importance is both the popular approaches find the contact position as a function of linear velocity of the CoM. However, more recently, works have started investigating incorporating angular momentum into the planning phase [PA16, GGP19, GGP21b, GGP21a, GG22]. For a humanoid, this could be a better indicator for the robot's state and where to place its feet, as a humanoid tends to have leg mass, which its movements can be captured through the angular momentum. In this regard, ARTEMIS is not only capable of planning footsteps using both the Raibert heuristics with Capture Point gains [AH20, HAY20], but also by footstep planning based on the desired one-step ahead angular momentum [GG22]. Both approaches are extendable to be used for running as well.

3.2.3.1 Angular Momentum-based Planning

In Section 1.2.3, we mentioned that an assumption with the most commonly used linear inverted pendulum model is the constant height and zero angular momentum assumption. The consequences of dropping $\mathbf{L}_G$ (angular momentum about the center of mass) when the 2D inverted pendulum's coordinates are the position and velocity of the center of mass have been studied in comparison to the less known coordinate representation of the inverted pendulum, where the states are the position of the center of mass and the angular momentum about the contact point [GG22]. This model has been coined as ALIP (where A is angular momentum), and we present a full derivation of the ALIP model in both the lateral and longitudinal directions and use it for footstep planning. We demonstrate that not only is this applicable with hybrid zero dynamics based approaches, but it is equally viable when used with the whole body control framework and actuated ankles. We also adapt this approach for planning footsteps with flight phases for running applications.

Let us first derive the ALIP model. To begin with, we know that we can transform the angular momentum about the CoM $\mathbf{L}_G$ to a different fixed point in the world as follows:

$$\mathbf{L}_{\mathtt{W}} = \mathbf{L}_G + m\mathbf{p}_{\mathtt{WR}} \times \dot{\mathbf{p}}_{\mathtt{WR}} \tag{3.8}$$

where we define $\mathbf{L}_{\mathtt{W}}$ as the angular momentum around the origin of the global coordinate system. m is the robot's mass, and $\mathbf{p}_{\mathtt{WR}}$ and $\dot{\mathbf{p}}_{\mathtt{WR}}$ are the position and velocity of the robot's CoM with respect to the global coordinate system. If we were to separately derive the footstep position along the $\mathbf{e}_x^{\mathtt{G}}$ and $\mathbf{e}_y^{\mathtt{G}}$, Equation (3.8) can be decomposed into the following two parts:

$$L_{\mathtt{W},y} = L_{\mathtt{G},y} + m \begin{bmatrix} p_{\mathtt{WR},x} \\ p_{\mathtt{WR},z} \end{bmatrix} \wedge \begin{bmatrix} \dot{p}_{\mathtt{WR},x} \\ \dot{p}_{\mathtt{WR},z} \end{bmatrix} \tag{3.9}$$

$$L_{\mathtt{W},x} = L_{\mathtt{G},x} + m \begin{bmatrix} p_{\mathtt{WR},y} \\ p_{\mathtt{WR},z} \end{bmatrix} \wedge \begin{bmatrix} \dot{p}_{\mathtt{WR},y} \\ \dot{p}_{\mathtt{WR},z} \end{bmatrix} \tag{3.10}$$

where $\wedge$ operator is the cross product in 2D. Equation (3.9) and Equation (3.10) can be used

along with:

$$\dot{L}_{\mathtt{W},x} = -mgp_{\mathtt{WR},y} + \tau_{\mathtt{A},x} \tag{3.11}$$

$$\dot{L}_{\mathtt{W},y} = mgp_{\mathtt{WR},x} + \tau_{\mathtt{A},y} \tag{3.12}$$

where g is gravity and $\tau_{\mathtt{A},x}, \tau_{\mathtt{A},y}$ are the torques at the ankles about their x and y axis, to represent the velocity of the center of mass as:

$$\dot{p}_{\mathtt{WR},x} = \frac{\dot{p}_{\mathtt{WR},z}}{p_{\mathtt{WR},z}} p_{\mathtt{WR},x} + \frac{L_{\mathtt{W},y} - L_{\mathtt{G},y}}{mp_{\mathtt{WR},z}} \tag{3.13}$$

$$\dot{p}_{\mathtt{WR},y} = \frac{\dot{p}_{\mathtt{WR},z}}{p_{\mathtt{WR},z}} p_{\mathtt{WR},y} - \frac{L_{\mathtt{W},x} - L_{\mathtt{G},x}}{mp_{\mathtt{WR},z}}. \tag{3.14}$$

Assuming a constant center of mass height ($p_{\mathtt{WR},z} = h, \dot{p}_{\mathtt{WR},z} = 0, \ddot{p}_{\mathtt{WR},z} = 0$), Equation (3.13) and Equation (3.14) become:

$$\dot{p}_{\mathtt{WR},x} = \frac{L_{\mathtt{W},y} - L_{\mathtt{G},y}}{mp_{\mathtt{WR},z}} \tag{3.15}$$

$$\dot{p}_{\mathtt{WR},y} = -\frac{L_{\mathtt{W},x} - L_{\mathtt{G},x}}{mp_{\mathtt{WR},z}}. \tag{3.16}$$

Dropping $L_{\mathtt{G},x}$ and $L_{\mathtt{G},y}$ as in [PA16] results in:

$$\dot{p}_{\mathtt{WR},x} = \frac{L_{\mathtt{W},y}}{mp_{\mathtt{WR},z}} \tag{3.17}$$

$$\dot{p}_{\mathtt{WR},y} = -\frac{L_{\mathtt{W},x}}{mp_{\mathtt{WR},z}}. \tag{3.18}$$

This finally brings us to the ALIP model [GG22]. From Equation (3.11), Equation (3.12), Equation (3.17), Equation (3.18), we can compactly represent the ALIP model along both $\mathbf{e}_x^{\mathtt{W}}$ and $\mathbf{e}_y^{\mathtt{W}}$:

$$\begin{bmatrix} \dot{p}_{\mathtt{WR},x} \\ \dot{p}_{\mathtt{WR},y} \\ \dot{L}_{\mathtt{W},x} \\ \dot{L}_{\mathtt{W},y} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \frac{1}{mh} \\ 0 & 0 & -\frac{1}{mh} & 0 \\ 0 & -mg & 0 & 0 \\ mg & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{\mathtt{WR},x} \\ p_{\mathtt{WR},y} \\ L_{\mathtt{W},x} \\ L_{\mathtt{W},y} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \tau_{\mathtt{A},x} \\ \tau_{\mathtt{A},y} \end{bmatrix}. \tag{3.19}$$

Using the ALIP model, a footstep planning strategy can be designed to indirectly track a desired velocity by controlling the angular momentum about the contact point $\mathbf{L}_{\mathtt{W}}$. For now, we assume the following:

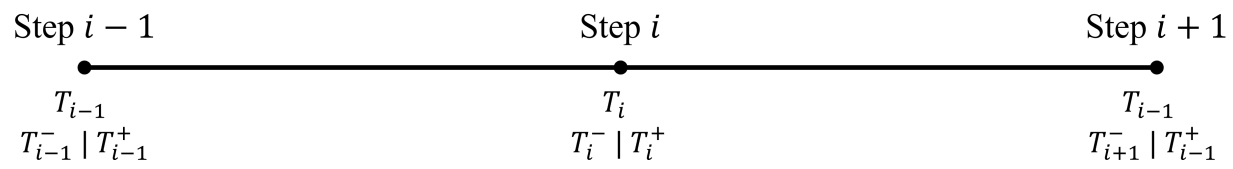

Figure 3.9: A timeline showing the contact events (steps) and the times at which they happen.

- A constant swing time of T_{swing}.
- A constant CoM height of $p_{\texttt{WR},z} = h$. The subsequent whole-body controller will control the CoM height at a constant value so that the footstep planner can at least make this assumption.
- A zero ankle torque. Along the sagittal plane (i.e. the pitch axis of the foot) ARTEMIS does have ankle actuation, but along the lateral plane (i.e. the roll axis of the foot), it does not have actuation.
- An instantaneous switch from one stance leg to the other. Bipedal walking is characterized by a single stance (i.e. one leg is supporting the body) and a double stance phase (i.e. both legs are supporting the body), but for now we assume that there are only single stances.

Following previous approaches [GG21,GG22] we plan for the current swing foot's footstep position based on the desired angular momentum at the end of the following step. For example, if our left foot is in stance and the right foot is in swing, we are solving for the right foot's footstep position at touchdown based on a desired angular momentum when the left foot touches down afterwards. This is a result of assuming no ankle actuation, which results in being unable to control $\mathbf{L}_{\texttt{W}}$ during the current step, but only through step transitions.

To derive the current desired footstep position, let us first define key events and their times during the current step and the ensuing step. From Figure 3.9, we can define the

following.

- We abbreviate T_{swing} as T for simplicity.
- Step $i-1$: The current stance foot's step.
- T_{i-1}: Time at which the current stance foot touched down.
- T_{i-1}^{-}: The instance before the touchdown at T_{i-1} and the end time of Step $i-2$.
- T_{i-1}^{+}: The instance after the touchdown at T_{i-1} and the start time of Step $i-1$.
- Step i: The step that the current swing foot will take.
- T_i: Time at which the current swing foot will touchdown.
- T_i^{-}: The instance before the touchdown at T_i and the end time of Step i.
- T_i^{+}: The instance after the touchdown at T_i and the start time of Step $i+1$.
- Step $i+1$: The step that the current stance foot will next take.
- T_{i+1}: Time at which the current stance foot will touchdown after taking the next step.
- T_{i+1}^{-}: The instance before the touchdown at T_{i+1} and the end time of Step $i+1$.
- T_{i+1}^{+}: The instance after the touchdown at T_{i+1} and the start time of Step $i+2$.

Based on the above, we can start to derive the controller that plans for foosteps based on a desired angular momentum in the coming step. Equation (3.19) has a closed form solution for time t_f from time t_0 as follows:

$$\begin{bmatrix} p_{\mathtt{WR},x}(t_f) \\ p_{\mathtt{WR},y}(t_f) \\ L_{\mathtt{W},x}(t_f) \\ L_{\mathtt{W},y}(t_f) \end{bmatrix} = \mathbf{A}(\Delta t) \begin{bmatrix} p_{\mathtt{WR},x}(t_0) \\ p_{\mathtt{WR},y}(t_0) \\ L_{\mathtt{W},x}(t_0) \\ L_{\mathtt{W},y}(t_0) \end{bmatrix} \tag{3.20}$$

where $\Delta t = t_f - t_0$, $\omega = \sqrt{\frac{g}{h}}$, and $\mathbf{A}(\Delta t)$ is:

$$\mathbf{A}(\Delta t) = \begin{bmatrix} \cosh(\omega\Delta t) & 0 & 0 & \frac{\sinh(\omega\Delta t)}{mh\omega} \\ 0 & \cosh(\omega\Delta t) & -\frac{\sinh(\omega\Delta t)}{mh\omega} & 0 \\ 0 & -mh\omega\sinh(\omega\Delta t) & \cosh(\omega\Delta t) & 0 \\ mh\omega\sinh(\omega\Delta t) & 0 & 0 & \cosh(\omega\Delta t) \end{bmatrix}. \tag{3.21}$$

Using Equation (3.20) we can observe how angular momentum could evolve up to the end of Step $i+1$. Then, we can set the desired angular momentum at the end of Step $i+1$ to be our desired angular momentum and calculate where we should place our feet at Step i to achieve it.

1. t to T_i^-:

   The angular momentum evolves according to Equation (3.20) row 3 and 4 where $\Delta t = T_i^- - t$ is continuously updated.

   $$L_{\mathtt{W},x}(T_i^-) = -mh\omega\sinh(\omega\Delta t)p_{\mathtt{WR},x}(t) + \cosh(\omega\Delta t)L_{\mathtt{W},x}(t) \tag{3.22}$$

   $$L_{\mathtt{W},y}(T_i^-) = mh\omega\sinh(\omega\Delta t)p_{\mathtt{WR},y}(t) + \cosh(\omega\Delta t)L_{\mathtt{W},y}(t). \tag{3.23}$$

2. T_i^- to T_i^+:

   When taking a step, assuming a constant height of h and a flat ground, momentum is constant under impact. This is because angular momentum transfers between two contact points as follows:

   $$\mathbf{L}_B = \mathbf{L}_A + \mathbf{p}_{BA} \wedge \dot{\mathbf{p}}_{\mathtt{Rcm}}. \tag{3.24}$$

   If the center of mass height is constant $\dot{p}_{\mathtt{Rcm},z} = 0$, then $\mathbf{p}_{BA} \wedge \dot{\mathbf{p}}_{\mathtt{Rcm}} = 0$ and $\mathbf{L}_B = \mathbf{L}_A$. Therefore, from T_i^- to T_i^+, the angular momentum is the same assuming the assumptions are valid, and the origin of the world frame is updated to the new contact

point.

$$L_{\mathtt{W},x}(T_i^+) = L_{\mathtt{W},x}(T_i^-) \tag{3.25}$$

$$L_{\mathtt{W},y}(T_i^+) = L_{\mathtt{W},y}(T_i^-) \tag{3.26}$$

$$p_{\mathtt{WRcm},x}(T_i^+) = p_{\text{SwingFoot}\mathtt{Rcm},x}(T_i^-) \tag{3.27}$$

$$p_{\mathtt{WRcm},y}(T_i^+) = p_{\text{SwingFoot}\mathtt{Rcm},y}(T_i^-) \tag{3.28}$$

where $\mathbf{p}_{\text{SwingFoot}\mathtt{Rcm}}$ is the position from the swing foot to the center of mass the moment before impact.

3. T_i^+ to T_{i+1}^-:

   After the current swing foot, we can predict the angular momentum to evolve as according to Equation (3.20) until the next step. Since swing time is fixed at T_{swing}, this results in:

$$L_{\mathtt{W},x}(T_{i+1}^-) = -mh\omega\sinh(\omega T_{\text{swing}})p_{\mathtt{WRcm},x}(T_i^+) + \cosh(\omega T_{\text{swing}})L_{\mathtt{W},x}(T_i^+) \tag{3.29}$$

$$L_{\mathtt{W},y}(T_{i+1}^-) = mh\omega\sinh(\omega T_{\text{swing}})p_{\mathtt{WRcm},y}(T_i^+) + \cosh(\omega T_{\text{swing}})L_{\mathtt{W},y}(T_i^+). \tag{3.30}$$

Recall from Equation (3.27) and Equation (3.28) that $p_{\text{SwingFoot}\mathtt{Rcm},x}(T_i^-)$ and $p_{\text{SwingFoot}\mathtt{Rcm},y}(T_i^-)$ is the current swing foot's desired position at Step i. Using Equations (3.27) to (3.30), the desired position of the current swing can be represented as:

$$p_{\text{SwingFoot}\mathtt{Rcm},x}(T_i^-) = \frac{L_{\mathtt{W},x}(T_{i+1}^-) - \cosh(\omega T_{\text{swing}})L_{\mathtt{W},x}(T_i^+)}{-mh\omega\sinh(\omega T_{\text{swing}})} \tag{3.31}$$

$$p_{\text{SwingFoot}\mathtt{Rcm},y}(T_i^-) = \frac{L_{\mathtt{W},y}(T_{i+1}^-) - \cosh(\omega T_{\text{swing}})L_{\mathtt{W},y}(T_i^+)}{mh\omega\sinh(\omega T_{\text{swing}})}. \tag{3.32}$$

Everything is known in Equations (3.31) and (3.32), since we know $L_{\mathtt{W},x}(T_i^+)$ and $L_{\mathtt{W},y}(T_i^+)$ from Equations (3.22), (3.23), (3.25) and (3.26), and $L_{\mathtt{W},x}(T_{i+1}^-)$, $L_{\mathtt{W},y}(T_{i+1}^-)$ can be our desired angular momentum at the end of the next step. Following [GG22], the desired angular momentum about the lateral axis can trivially be $L_y^d = mh\dot{p}_{\mathtt{WRcm},x}$ while about the sagittal axis, we can assume an offset to the pendulum dynamics:

$$L_x^d = \mp 0.5mhW_{\text{step}}\frac{\omega\sinh(\omega T_{\text{swing}})}{1+\cosh(\omega T_{\text{swing}})}$$

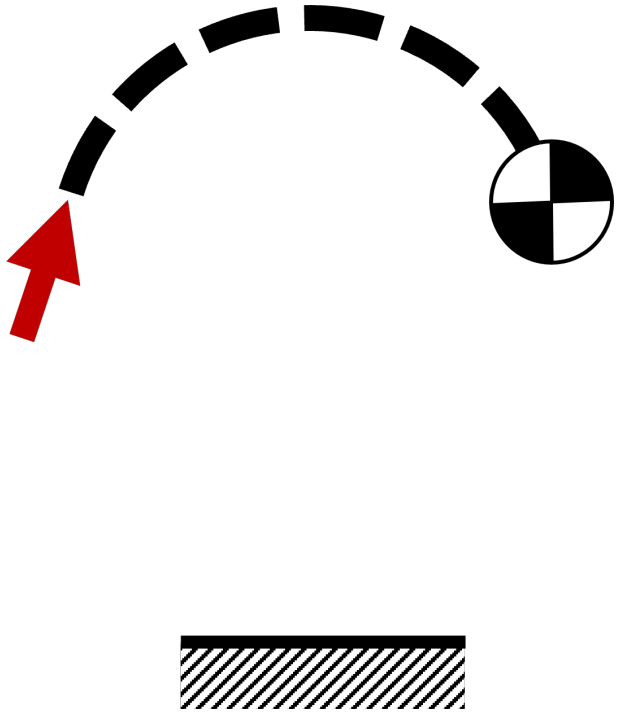

Figure 3.10: Desired ballistic trajectory of the center of mass when in flight phase.

where $\mp$ is negative if the current stance is left leg and positive if it is right leg, and W_{step} is the desired width between the two steps.

3.2.4 Center of Mass Trajectory Planner

Given the foot swing times and the lift-off percentage, we wish to actively generate a center of mass position and velocity trajectory such that when the robot is in a ballistic state, the center of mass should smoothly move up and down as seen in Figure 3.10. Whereas if the robot is in contact with the ground, the center of mass trajectory should be symmetric and convex as seen in Figure 3.11. The center of mass velocities should correspond to the derivatives of the position, except that prior to lift off, there is a vertical velocity that must be reached to satisfy a desired flight phase. Additionally when in flight phase, immediately the velocity slope should be negative until touchdown.

To start with the ballistic trajectory, the desired velocity at the next timestep should be

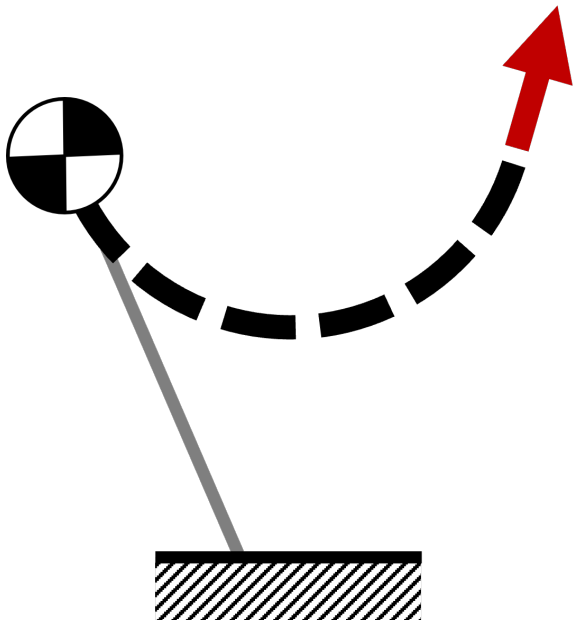

Figure 3.11: Desired convex trajectory of the center of mass when in a foot is in contact.

the current velocity subtracted by the product of gravity and the timestep:

$$v_{\texttt{WRcm},t+1} = v_{\texttt{WRcm},t} - g \cdot \mathrm{dt} \tag{3.33}$$

The desired position is a straightforward integration of the velocity:

$$p_{\texttt{WRcm},t+1} = p_{\texttt{WRcm},t} + v_{\texttt{WRcm},t} \cdot \mathrm{dt} \tag{3.34}$$

For the center of mass trajectory when the robot is in contact with the ground, in an ideal situation, we desire a symmetric, convex arc to reach a desired lift off trajectory. To begin with, using the nominal swing time $t_{\text{swing,nom}}$ and lift off percentage s_{LO}, the nominal air time $t_{\text{air,nom}}$, ground time $t_{\text{ground,nom}}$, and lift off velocity $v_{\texttt{WRcm},\text{z,nom}}$ can be found.

$$t_{\text{air,nom}} = (1 - s_{\text{LO}}) \tag{3.35}$$

$$t_{\text{ground,nom}} = s_{\text{LO}} t_{\text{swing,nom}} - t_{\text{air,nom}} \tag{3.36}$$

$$v_{\texttt{WRcm},\text{z,nom}} = g \cdot t_{\text{air,nom}}/2. \tag{3.37}$$

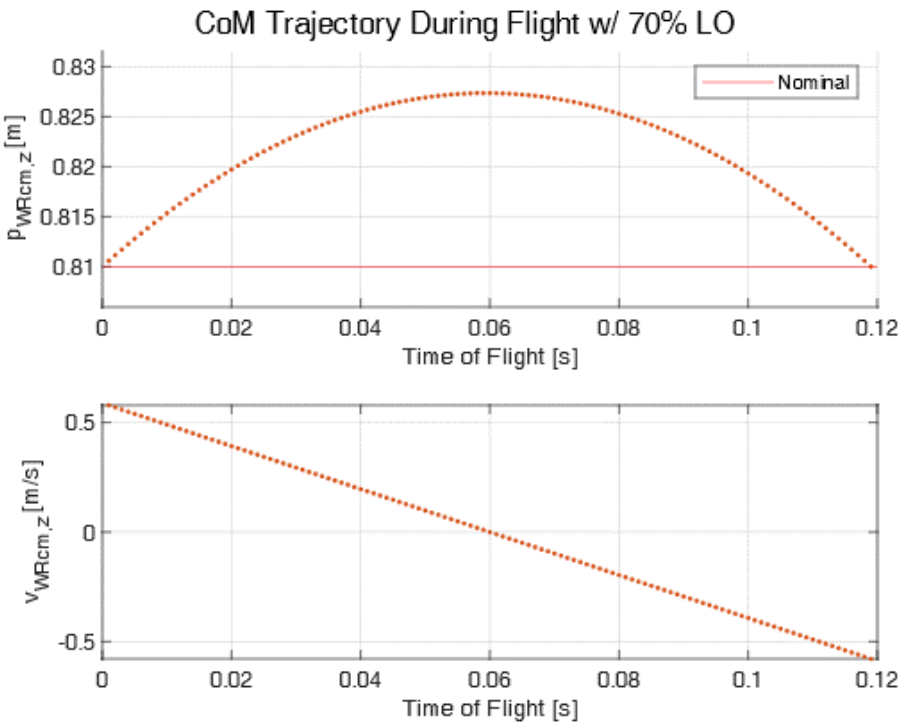

Figure 3.12: Desired center of mass trajectory when in flight with a swing time of 0.4 s and lift-off percentage of 70%.

Afterwards, depending on the swing phase s_{swing} of the foot that is not in contact, the time until the next lift off t_{nextLO} is computed along with the corresponding velocity and position of the center of mass required at the time by:

$$s_{\text{nextLO}} = s_{\text{LO}} - s_{\text{swing}} \tag{3.38}$$

$$t_{\text{nextLO}} = t_{\text{swing,nom}} s_{\text{nextLO}} \tag{3.39}$$

$$v_{\texttt{WRcm}} = 2\frac{v_{\texttt{WRcm,z,nom}}}{t_{\text{ground,nom}}}\left(\frac{t_{\text{ground,nom}}}{2} - t_{\text{nextLO}}\right) \tag{3.40}$$

$$p_{\texttt{WRcm}} = p_{\texttt{WRcm,nom}} + \frac{v_{\texttt{WRcm,z,nom}}}{t_{\text{ground,nom}}} t_{\text{nextLO}}(t_{\text{nextLO}} - t_{\text{ground,nom}}). \tag{3.41}$$

Using the earlier example with a swing time of 0.4 s and a lift-off percentage of 70%, through Equation (3.33) and Equation (3.34), a ballistic trajectory as seen in Figure 3.12 can be generated. A center of mass trajectory using Equation (3.38) when in stance phase can be seen in Figure 3.13.

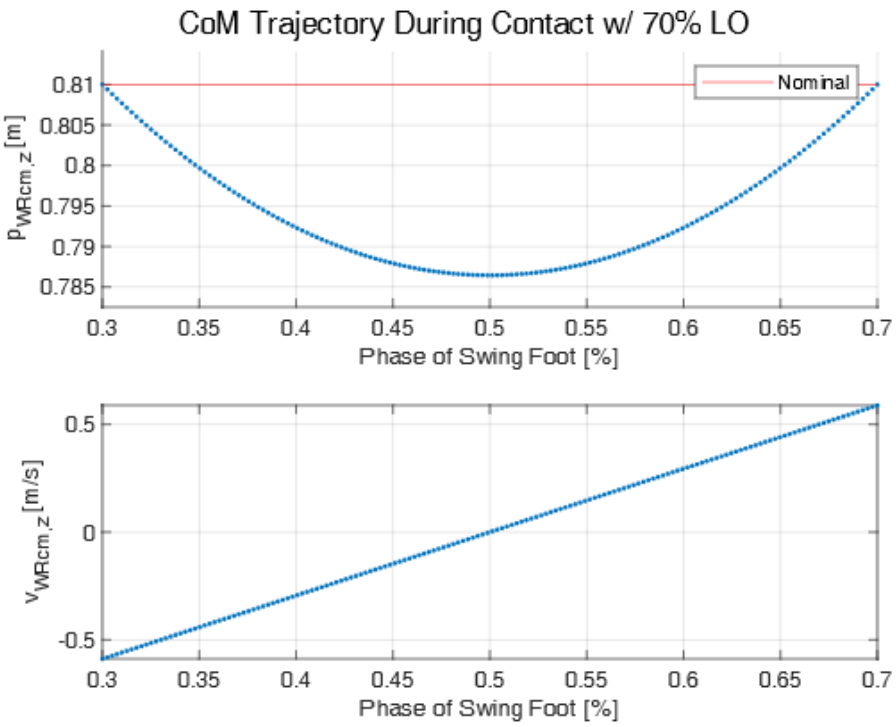

Figure 3.13: Desired center of mass trajectory when in stance with a swing time of 0.4 s and lift-off percentage of 70%.

A desired center of mass trajectory over the course of 4 steps using the same swing time and lift-off percentage can be seen in Figure 3.14. If we adjust the lift-off percentage to 90%, the desired center of mass trajectory would automatically be modified, as seen in Figure 3.15. If the lift-off percentage becomes 100%, the center of mass trajectory collapses to the case of a constant center of mass height. This can be seen in Figure 3.16. These trajectories will be become reference trajectories to be tracked later by the whole-body controller in Section 3.2.6.

3.2.5 Foot Trajectory Planner

Next, given a desired footstep position, we must plan a potential trajectory from the current foot position to the desired footstep position for the end-effector to follow. The simplest approach could be a linearly discretized trajectory that moves the foot from one position in

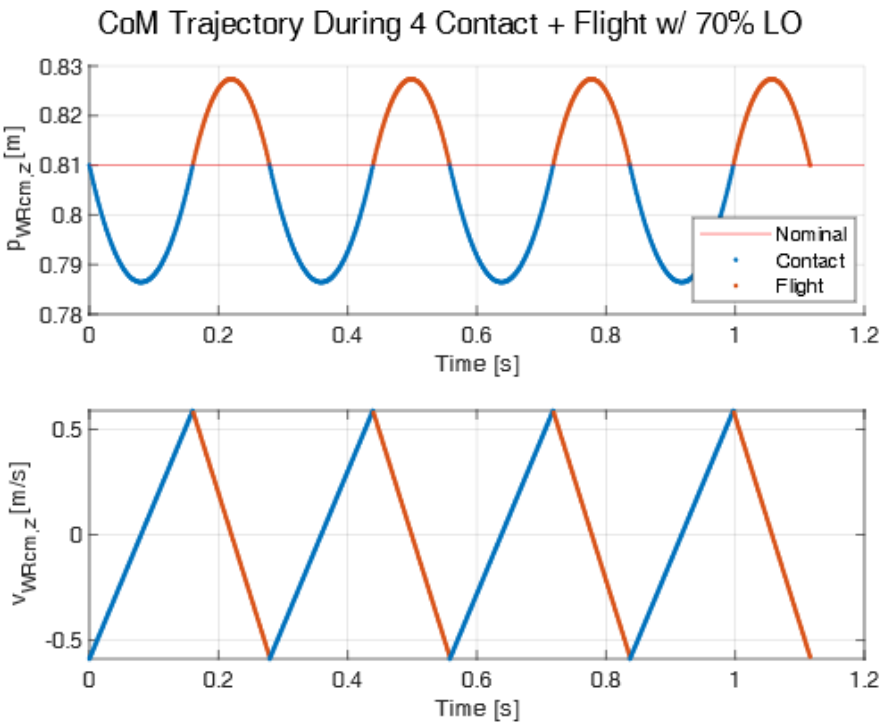

Figure 3.14: Desired center of mass trajectory over the course of 4 steps with a swing time of 0.4 s and lift-off percentage of 70%.

the X and Y plane to another position, while moving the foot up and down in a sinusoidal fashion in the Z direction. While simple to implement, this approach does have its shortcomings in that the timestep (i.e. discretization interval) needs to be known in advance and it is also non-trivial to get the trajectory's exact derivative (i.e. velocity). If we wanted to know the exact position at any given time throughout the trajectory, this is not feasible with such a discretized trajectory. Also, while we could assume the velocity to be the change in position over the timestep, the resulting velocity will also not be continuous.

Another approach could be to use a parametric trajectory. If a trajectory is defined using parametric equations, interpolation is no longer needed and we can query the trajectory's position at any given time. We can also obtain the derivative of the trajectory which also could be continuous. Some examples of these trajectories include the cycloid trajectory, the minimum jerk trajectory, a polynomial, a Bezier curve, or any spline [FH85].

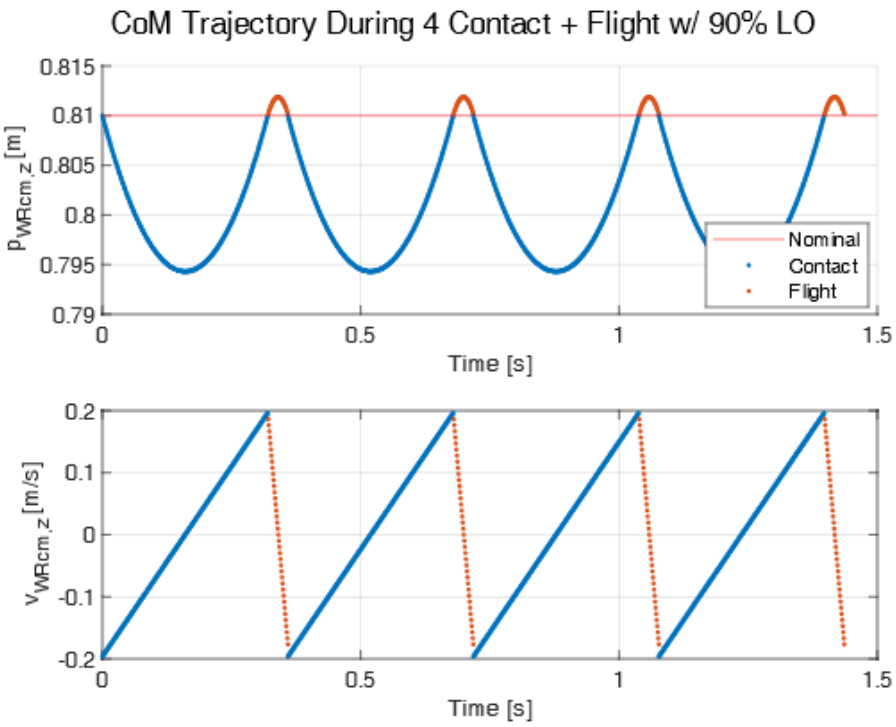

Figure 3.15: Desired center of mass trajectory over the course of 4 steps with a swing time of 0.4 s and lift-off percentage of 90%.

Different to the center of mass trajectory, a footstep trajectory has its own additional properties that we would like to have in the final desired trajectory.

- Desired footstep height: While there are dance moves where the foot slides on the ground, usually when we take a step, it is required to clear the foot off from the ground. There also may be obstacles or steps that that we want to step over and onto.
- Zero initial velocity and acceleration: A foot in contact with the ground has zero velocity and acceleration. Therefore, the position trajectory as the foot starts to move towards the final position should have a zero initial velocity and acceleration.
- Zero final velocity and acceleration: When a foot makes contact with the ground at the end of a step, it also should reach zero velocity and acceleration. This is also a desired behavior because sudden stops or expecting the foot to collide with the ground

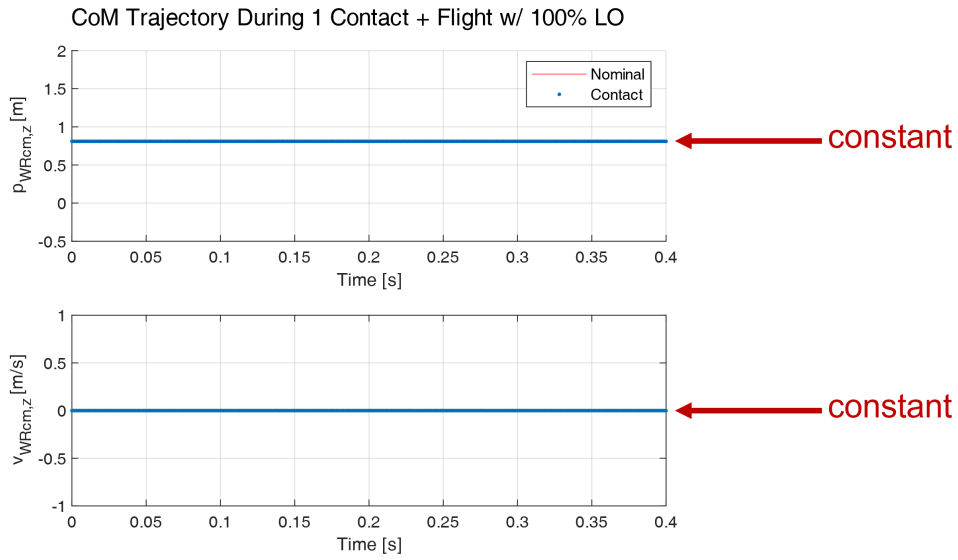

Figure 3.16: Desired center of mass trajectory with a desired lift-off percentage of 100%.

and come to a stop because of the collision with the environment is undesirable as it will introduce undesirable impacts to the system.

Analytical approaches such as the minimum jerk do have properties such as a smooth initial and final velocity and acceleration, but if we were to generate a foot trajectory using it, we would have to stitch two such trajectories to create a single trajectory that moves up to a desired footstep position and then comes down. However at the apex, the foot would momentarily come to a stop and then come back down.

Therefore, parametric representations of trajectories that can, at minimum, satisfy the aforementioned properties, are increasingly used [LLL17, PWK17, KHM20, AH20]. Often times this is achieved by embedding them into an optimization framework to explicitly constrain them through constraints or implicitly as a cost.

Because ARTEMIS walks dynamically and its desired footstep position is constantly changing, we formulate an optimization to solve the trajectory generation problem. Given a desired footstep position from the previous section, a desired swing time, and a desired step height, the optimization solves for a trajectory that satisfies these properties as well as the boundary conditions (i.e. zero initial and final velocity and acceleration). Two different

optimizations are run to generate the complete trajectory in 3D space. They are both continuous in acceleration as they will be used in the subsequent inverse dynamics control, which benefits from continuous acceleration trajectories [SHV06].

The first optimization solves for the trajectory in the X and Y directions. In the case of the trajectory along the XY plane, the only objective is to reach the final footstep position. Therefore, it is sufficient for us to formulate an optimization problem that solves for the coefficients of a quintic spline. A quintic spline is continuous in velocity and acceleration, and has enough degrees of freedom to constrain its starting and ending position, velocity and acceleration [ACH18]. The cost function simply tries to minimize the acceleration of the trajectory. The optimization is as follows:

$$\min_{\mathbf{c}} \quad \mathbf{c}^\top \mathbf{Q}_\mathrm{c} \mathbf{c} \tag{3.42a}$$

$$\text{s.t.} \quad \mathbf{A}_\mathrm{eq}\mathbf{c} = \mathbf{b}_\mathrm{eq}. \tag{3.42b}$$

We use the coefficients of the quintic spline as the decision variables:

$$\mathbf{c}^\top = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{bmatrix}$$

such that the resulting trajectory can be reconstructed as:

$$p(t) = c_1 t^5 + c_2 t^4 + c_3 t^3 + c_4 t^2 + c_5 t + c_6.$$

The cost function's Hessian is

$$\mathbf{Q}_c = \begin{bmatrix} 400T^6 & 240T^5 & 120T^4 & 40T^3 & 0 & 0 \\ 240T^5 & 144T^4 & 72T^3 & 24T^2 & 0 & 0 \\ 120T^4 & 72T^3 & 36T^2 & 12T & 0 & 0 \\ 40T^3 & 24T^2 & 12T & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

with T being the pre-defined swing time, and:

$$\mathbf{A}_{\text{eq}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix}$$

$$\mathbf{b}_{\text{eq}}^{\top} = \begin{bmatrix} p(0) & p(T) & \dot{p}(0) & \dot{p}(T) & \ddot{p}(0) & \ddot{p}(T) \end{bmatrix}$$

is the equality constraint matrix with row 1 and 2 constraining the initial and final positions, row 3 and 4 constraining the initial and final velocities, and row 5 and 6 constraining the initial and final accelerations. Note that Equation (3.42) is for a single dimension (i.e. either X or Y), but identical cost function and the equality matrix can be block diagonally constructed with the decision variables stacked to solve a single optimization problem that solves for both X and Y directions.

The second optimization solves for the trajectory along the Z direction. Unlike the trajectory along the XY plane, the trajectory along the Z direction must satisfy a step height objective as well, where the foot should be lifted off the ground by a pre-defined step height halfway through its trajectory. Stitching two quintic splines can be a possible solution as unlike the minimum jerk trajectory, we could constrain the final position, velocity, and acceleration of the first quintic spline that reaches the apex of the swing trajectory to be equal to the start position, velocity, and acceleration of the second quintic spline. However, in the case the swing time is modulated (which is a potential stepping strategy in bipedal locomotion), this could result in undesirable jumps in the trajectory [KHM20]. Therefore, a single continuous trajectory is more favorable and in our case, we use a ninth-order polynomial. In a similar fashion to Equation (3.42), we solve for the coefficients of a

ninth-order polynomial with boundary constraints.

$$\min_{\mathbf{c}} \quad J(\mathbf{c}, h_{\text{step}}) \tag{3.43a}$$

$$\text{s.t.} \quad \mathbf{A}_{\text{eq}}\mathbf{c} = \mathbf{b}_{\text{eq}} \tag{3.43b}$$

where $\mathbf{c}$ is the vector of coefficients of the ninth order trajectory:

$$\mathbf{c}^\top = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} \end{bmatrix}$$

such that the resulting trajectory can be reconstructed as:

$$p(t) = c_1 t^9 + c_2 t^8 + c_3 t^7 + c_4 t^6 + c_5 t^5 + c_6 t^4 + c_7 t^3 + c_8 t^2 + c_9 t + c_{10}.$$

The cost function $J(\mathbf{c}, h^d_{\text{step}})$ minimizes both the acceleration of the entire trajectory as well as the distance between the foot midway through the swing phase and the desired step height h^d_{step}:

$$J(\mathbf{c}, h_{\text{step}}) = ||\ddot{p}|| + ||p(T/2) - h^d_{\text{step}}||. \tag{3.44}$$

This results in a cost function that can be put in the standard QP form where the quadratic cost is:

$$\mathbf{H} = \begin{bmatrix} \frac{T^{18}}{262144} & \frac{T^{17}}{131072} & \frac{T^{16}}{65536} & \frac{T^{15}}{32768} & \frac{T^{14}}{16384} & \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} \\ \frac{T^{17}}{131072} & \frac{T^{16}}{65536} & \frac{T^{15}}{32768} & \frac{T^{14}}{16384} & \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} \\ \frac{T^{16}}{65536} & \frac{T^{15}}{32768} & \frac{T^{14}}{16384} & \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} \\ \frac{T^{15}}{32768} & \frac{T^{14}}{16384} & \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} \\ \frac{T^{14}}{16384} & \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} \\ \frac{T^{13}}{8192} & \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} & \frac{T^4}{16} \\ \frac{T^{12}}{4096} & \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} & \frac{T^4}{16} & \frac{T^3}{8} \\ \frac{T^{11}}{2048} & \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} & \frac{T^4}{16} & \frac{T^3}{8} & \frac{T^2}{4} \\ \frac{T^{10}}{1024} & \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} & \frac{T^4}{16} & \frac{T^3}{8} & \frac{T^2}{4} & \frac{T}{2} \\ \frac{T^9}{512} & \frac{T^8}{256} & \frac{T^7}{128} & \frac{T^6}{64} & \frac{T^5}{32} & \frac{T^4}{16} & \frac{T^3}{8} & \frac{T^2}{4} & \frac{T}{2} & 1 \end{bmatrix}$$

and the linear cost is:

$$\mathbf{f} = -h^d_{\text{step}} \begin{bmatrix} \frac{T^9}{256} & \frac{T^8}{128} & \frac{T^7}{64} & \frac{T^6}{32} & \frac{T^5}{16} & \frac{T^4}{8} & \frac{T^3}{4} & \frac{T^2}{2} & T & 2 \end{bmatrix}.$$

The equality constraint matrix and vector are similar to that for the quintic spline except it now considers the coefficients corresponding to the higher order terms:

$$\mathbf{A}_{\text{eq}} = \begin{bmatrix} T^9 & T^8 & T^7 & T^6 & T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 9T^8 & 8T^7 & 7T^6 & 6T^5 & 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 72T^7 & 56T^6 & 42T^5 & 30T^4 & 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}.$$

Now that we have the 3D trajectory for the position of the swing foot, we need to design the rotational trajectory of the foot. Given the desired footstep orientation from the footstep planner, we design a separate trajectory for the pitch and yaw position of the foot. To keep things simple, the desired yaw trajectory of the foot is simply a linear interpolation from the current yaw to the desired yaw position computed based on the desired yaw rate sent from the joystick commands. For the pitch trajectory, the trajectory is a constant value that will keep the foot in parallel with the ground.

3.2.6 Whole-Body Control

Given the desired swing trajectories that we would like the robot to follow, we would like to use the full-order model to find the torques to send to the robot. This is because up until this point, the planning has been done using reduced-order models. Some of the assumptions made to use the reduced-order model may be inappropriate (e.g. the massless leg assumptions used in pendulum models). Fortunately, provided desired tasks or operational space objectives (e.g. end-effector position, link orientation, center of mass position, momentum) defined using its desired value and the Jacobian, the instantaneous joint accelerations $\ddot{q}$, torques $\boldsymbol{\tau}$, and reaction forces $\mathbf{F}$ that are dynamically consistent can be found under the whole-body control framework. There are primarily two ways of solving this problem. Given the multiple different tasks/objectives, one approach (weighted) tries to find the solution while finding a balance between all of them. The other approach (prioritized) constructs a

hierarchy or a priority between the different tasks and solves for the solution while adhering to the order.

To solve the problem using the weighted approach, a QP of the following form needs to be solved:

$$\min_{\ddot{\mathbf{q}},\,\mathbf{f}} \quad \sum_{i=1}^{N_t} ||\mathbf{J}_i\ddot{\mathbf{q}} + \dot{\mathbf{J}}_i\mathbf{q}_i - \mathbf{a}_i^d||_{\mathbf{W}_i} + \sum_{j=1}^{N_j} ||\mathbf{f}||_{\mathbf{W}_j} + ||\ddot{\mathbf{q}}||_{\mathbf{W}_{\ddot{q}}} \tag{3.45a}$$

$$\text{s.t.} \quad \mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\mathbf{g} = \sum_{j=1}^{N_c} \mathbf{J}_{c,j}^\top \mathbf{f}_j, \tag{3.45b}$$

$$\mathbf{f}_j \in \mathcal{C}_j \quad j = 1, \cdots N_c \tag{3.45c}$$

where N_t is the number of tasks, N_c is the number of contacts, and $\mathcal{C}_j$ is the friction cone constraint for contact j. To keep things linear, the friction cone constraint can be approximated as a friction pyramid constraint. Additionally, while this formulation only constrains the floating base acceleration, the entire dynamics can also be constrained while constraining the torque limits by replacing the constraints in Equation (3.45) with:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\mathbf{g} = \boldsymbol{\tau} + \mathbf{J_c}^\top \mathbf{f}$$

$$\underline{\boldsymbol{\tau}} \leq \boldsymbol{\tau} \leq \bar{\boldsymbol{\tau}}$$

$$\mathbf{f} \in \mathcal{C}_i$$

The cost function consists of all information from a task, which is defined by its desired value and its Jacobians. $\mathbf{J}_i$ and $\dot{\mathbf{J}}_i$ are the task Jacobian and its derivative corresponding to task i, and $\mathbf{a}_i^d$ is the desired task space acceleration defined as:

$$\mathbf{a}^d = \mathbf{K}_p\Delta\mathbf{p} + \mathbf{K}_d\Delta\mathbf{v} + \mathbf{a}^{\text{ref}} \tag{3.46}$$

where $\Delta\mathbf{p} \in \mathbb{R}^3$ and $\Delta\mathbf{v} \in \mathbb{R}^3$ are the position and velocity errors, $\mathbf{K}_p \in \mathbb{R}^{3\times3}$ and $\mathbf{K}_d \in \mathbb{R}^{3\times3}$ are proportional and derivative gains, and $\mathbf{a}^{\text{ref}} \in \mathbb{R}^3$ is the reference acceleration. This feedback (PD) and feedforward (reference $\mathbf{a}^{\text{ref}}$) can be used either for a translational term such as the center of mass position or a rotational term such as the body orientation. $\mathbf{W}_i \in \mathbb{R}^{3\times3}$ is the weight for task i, which can be used to introduce implicit hierarchy between

the different tasks. $\mathbf{f}$ and $\ddot{\mathbf{q}}$ are regularization costs to ensure the the optimization's cost function is positive definite, while $\mathbf{W}_j$ and $\mathbf{W}_{\ddot{q}}$ are their respective weights which are kept small.

The advantage of this formulation is the simplicity in that only a single optimization problem needs to be solved. However, in reality, finding the right balance in weights between the different tasks can be a challenge. Furthermore, significant differences in the magnitudes of the different tasks to enforce an implicit hierarchy between the different tasks can cause their own numerical issues.

Instead, a strict hierarchy could be enforced to the different tasks to guarantee a task with a higher priority will be guaranteed before a task of lower priority is allowed. For example, a task with priority 1 is more important than a task with priority 2 (note that a priority of lower numerical value is of higher importance), so 2 will only be satisfied while 1 is satisfied. This can be achieved through a series of optimization problems.

If the series or cascade of optimization problems are unconstrained least squares problems or an equality constrained least squares problems, they can be solved efficiently. Given a simple least squares problem of the following form:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \, ||\mathbf{A}\mathbf{x} - \mathbf{b}||$$

it can be solved by a simple pseudoinverse, resulting in $\mathbf{x}^* = \mathbf{A}^\dagger b$. If the problem includes an equality constraint $\mathbf{C}\mathbf{x} = \mathbf{d}$, the solution can be found using the nullspace $\mathcal{N}(\mathbf{C}) = \mathbf{N}_\mathbf{C} = \mathbf{I} - \mathbf{C}^\top\mathbf{C}$, which results in $\mathbf{x}^* = \mathbf{C}^\dagger\mathbf{d} - \mathbf{N}_\mathbf{C}(\mathbf{A}\mathbf{N}_\mathbf{C})^\top(\mathbf{b} - \mathbf{A}\mathbf{C}^\dagger\mathbf{d})$.

However, if inequality constraints are required (e.g. to enforce torque limits or friction constraints), we need to revert back to an optimization problem to include them. To also consider priorities/hierarchicies, a series of optimizations must be solved in descending priorities while retaining the optimal solution of the previous optimizations.

In the case with ARTEMIS, to ensure a solution is always available online, the weighted approach was taken to reduce the number of quadratic programs that need to be solved at 500 Hz. This meant significant investment had to be made into tuning the feedback

gains for the desired accelerations as well as the weights in the cost function. Details of the implementation are further described in Section 4.2.

CHAPTER 4

Implementation and Results

Ideally the implementation of the approaches are identical both in simulation and on hardware. However, because of the many practical differences such as the imperfect model of the physical platform, the noise in the state information, or the computational burdens of running different algorithms simultaneously to name just a few, the implementation to realize the controllers on hardware can be non-trivial. In that regard, this chapter details the steps taken to implement the approach described in Chapter 3 on hardware. Additionally, results to assess the performance of the approach are also presented.

4.1 Robot State Estimation

As noted in Section 3.2, the planning of the task space trajectories are with respect to the world frame. However, prior to planning, the pose and the velocities of the floating base (i.e. the base frame of the robot) are required. This requires an estimation of the floating base pose and velocities. Different approaches exist in literature, ranging from using complementary filters to estimate these values [DPL21, She22], to variants of the Kalman Filter [CED08, PHH09, BHH13, WLC13, RBR14], and more recently, deep learning-based approaches as well [JMK22]. For ARTEMIS, we implemented a variant of the Extended Kalman Filter known as the Invariant Extended Kalman Filter (InEKF) [BMS09] to estimate the global pose and velocities of the robot [HGE20]. The motivation behind this choice among others was the strong convergence properties that are present in a nonlinear observer.

The estimator takes in angular velocities and linear accelerations from the IMU, joint angles from the legs (i.e. leg kinematics), and contact information. Information from the

Measurement	Noise
Gyroscope	0.002
Accelerometer	0.004
Contact	0.001
Kinematics	0.001

Table 4.1: Noise parameters for the robot's global pose and velocities state estimator.

IMU is used to propagate the dynamics forward, and then kinematics measurement is used for correction. While both the left and the right invariant EKF exists, the estimator was tuned using the right invariant EKF. Additionally, it is often the case that the gyroscope biases and accelerometer biases are also estimated along with the global pose and the velocities of the robot when an IMU is used for state estimation. However, because this conflicts with the standard invariant EKF theory, resulting in an imprecise invariant EKF, the bias estimations were not done using the invariant EKF. Instead, because the IMU has its own EKF, which computes a filter-compensated angular velocity and linear acceleration using its own bias estimation, these values were used to run the invariant EKF with no additional bias estimation [mic]. Afterwards, tuning of the hyperparameters of the invariant EKF is required to get the filter to quickly converge. These parameters are shown in Table 4.1.

Additionally, as noted in [RBR14, HGE20], the complete pose of the robot is not fully observable at all times. The absolute position of the robot as well as its rotation about the gravity vector (yaw) is unobservable. This can be seen in Figure 4.1, which shows a bird's eye view of the robot's SE(2) pose as it walks around the 3rd floor of UCLA's Engineering IV building. As expected, the robot started at $X = 0$ m, $Y = 0$ m with its heading angle aligned with the positive X axis (i.e. 0° yaw). At the end of the walk, ARTEMIS ended up at $X = 13.45$ m, $Y = 4.10$ m with a heading angle of 20.13°. In reality however, the robot started and ended at the same place, but significant drift in its pose had occurred over the course of 554 seconds of walking. Note however, this information is not required for the robot to achieve dynamically stable locomotion. Rather, the roll and pitch of the floating

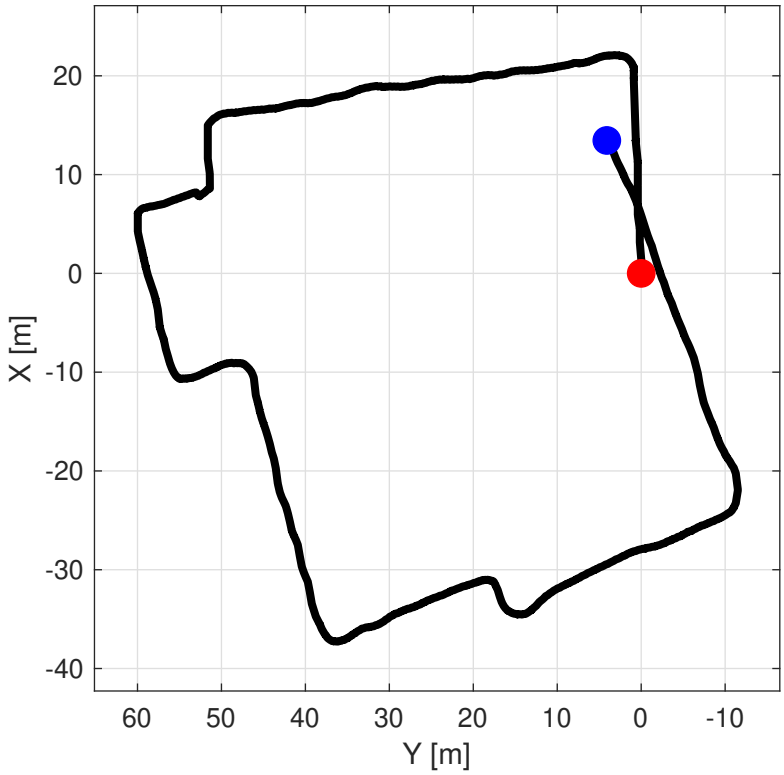

Figure 4.1: Absolute position and yaw output of the state estimator after walking around the 3rd floor of UCLA's Engineering IV building seen in birds eye view.

base and its velocities are important for stability as the footstep planner decides where to place the foot based on these values and the whole body controller will correct the body orientation from the estimates.

Ideally, the accuracy of the estimate would be verified by comparing to the outputs of a motion capture system, but in the case with ARTEMIS, the estimator was tuned empirically. This was achievable by initially keeping the robot standing in position control but with the state estimator enabled. By pushing and rotating the robot manually without breaking contact, the general trend of the estimator's velocity and orientation outputs could be observed. Afterwards, the balance controller was tuned in tandem with the state estimator. Once the balance controller could withstand external disturbances, the locomotion controller and the estimator was simultaneously tuned. The same iterative approach was taken during walking

and running (and jumping) tests, except different outputs from the state estimator were observed during these two different behaviors.

For walking, a ground truth that we can approximate by observation is the distance the robot travels over a period of time. ARTEMIS initially taking a step and approximately measuring the base frame's displacement over time was compared with the estimator's output to initially tune the estimator. If walking was measuring the displacement along the X and Y directions, for running, the initial step was to assess the estimator's behavior when in ballistic motion (i.e. along the Z direction). An offline jumping trajectory was generated and tracked using the techniques from Section 3.2.6. From first principles, symmetry in the estimator's trajectory during ballistic motion was expected. However, that was not the case until the IMU currently on the robot was integrated into the system, as the robot always *seemingly* landed earlier (i.e. the estimator thought the robot had jumped onto a higher platform).

Note that while one may think that the estimator could be tuned offline, this is only possible assuming ground truth data is available. Another thought could be that the estimator could be tuned in simulation assuming a good simulated representation of the IMU's noise parameters from the real world. While this seemed initially promising, because of multiple additional uncertainties in reality (e.g. sensor noise in the joint encoders, slight slips in the contact), re-tuning was required in reality.

During the integration and tuning process of the state estimator, the IMU on ARTEMIS was replaced twice, initially from Vectornav VN-100 to Microstrain 3DM-GX5-25 AHRS, and then to the current IMU, Microstrain 3DM-CV7-AHRS. The initial replacement was to improve the frequency of the IMU's output, as the VN-100 required additional steps to obtain a raw 1,000 Hz reading of the angular velocity and linear acceleration readings as opposed to the GX5-25. However, when using the GX5-25, the state estimator had difficulty estimating a symmetric ballistic trajectory along the Z direction. In the air, the estimation should have primarily been driven by the IMU dynamics, which should have created the symmetry we wanted to see. Instead, the estimation thought the robot was landing at a

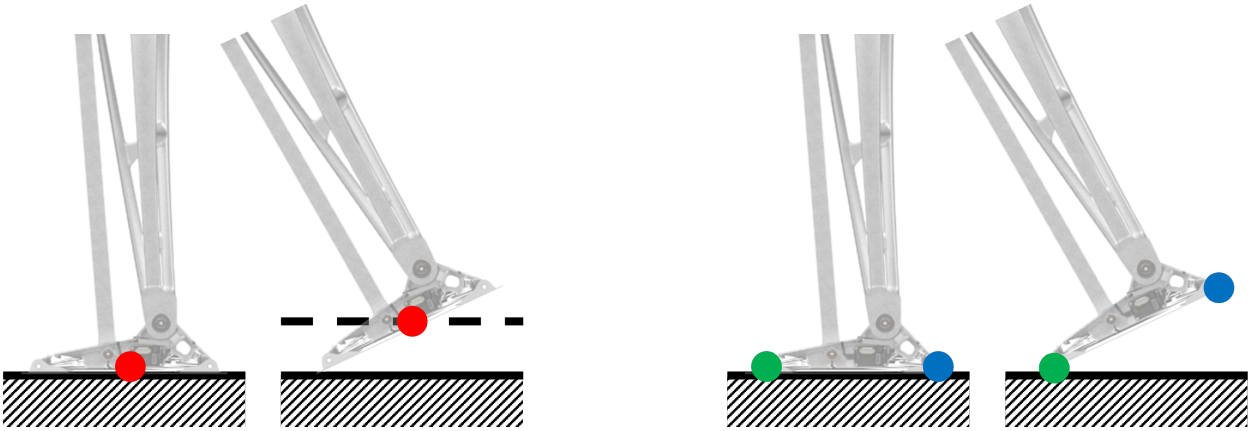

Figure 4.2: Left: If the contact frame for the state estimator is set to only be at the middle of the foot, anytime the foot is not parallel with the ground and comes in contact with the ground, the estimator will think the robot took a step up from the ground to the dotted line. Right: If the contact frame for the state estimator is set to be at the edges of the foot (toe and heel), the offset between the contact point and the contact frame will be minimized, leading to less drift along the Z direction.

higher Z location than where it jumped off of, suggesting a possible delay in the estimator.

This was validated when the GX5-25 was replaced with the CV7. The same raw angular velocity and linear acceleration values were streamed at 1,000 Hz from both the GX5-25 and the CV7. Surprisingly, the GX5-25 had a 10 ms delay compared to the CV7, resulting in the asymmetric ballistic behavior. With the relatively reduced delay when the IMU was replaced with the CV7, the estimator's output showed a near symmetric ballistic trajectory.

Another small, yet important detail during the implementation of the state estimator was defining the contact positions. As can be seen in the frame definitions in Figure 2.5, we defined three frames on the foot. Although the toe and the heel frames are used to put a high cost on their positions during contact, these frames are also used during state estimation. At the simplest, it is possible to only use the sole frame (or any single frame), but for a humanoid that has a line foot (or any non-point foot), any contact when the foot is not parallel with

the ground (which is actually any time that the ground is not completely flat) will result in small offsets between the contact frame and the actual point of contact. Even if the ground was completely flat, the foot's pitch control will have small errors. This behavior can be seen in Figure 4.2. Although the offset may seem small, because the estimator is updated at 500 Hz, the differences can accumulate quickly. While it is difficult to deterministically compare the performance between a single point contact and a two point contact by nature, empirically, we have seen significantly less "drift" in position when the two point contact approach was used.

4.2 Whole Body Control

Tuning the whole body controller can be a significant challenge because of the sheer number of gains and weights that need to simultaneously be tuned. Additionally, optimization solvers need to be deployed with the appropriate settings to achieve a robust, consistent solution. In hopes that future researchers can start from some reference rather than from scratch, the weights and gains used in the whole body controller are shown in Table 4.2 and Table 4.3.

To solve the quadratic problem presented in Equation (3.45) using the weights and gains from above, an optimization solver is required to find the solution. It must also find the solution fast enough as the controller is required to be run in a 500 Hz loop. With Equation (3.45), the optimization's structure does not change, hence a custom solver tailored for this problem could be the ideal approach to obtaining the solution as quickly as possible. Instead however, for simplicity, an off-the-shelf solver was used and its parameters were tuned to satisfy our timing constraints. On ARTEMIS, off-the-shelf solvers OSQP [SBG20] and ProxQP [BET22] were used, with the former being primarily used during walking and the latter for running. For OSQP, the settings used are shown in Table 4.4.

Custom settings are motivated by the need to satisfy solve time constraints and smoothness of the solutions. For example, because the structure of the problem is fixed during locomotion (a single support leg is always assumed during walking), it is beneficial to always

Tasks	Weights		
	Balancing	**Walking**	**Running**
Center of Mass Position	[10, 10, 100]	[1, 1, 100]	[1, 1, 100]
Body Orientation	[80, 80, 80]	[20, 20, 40]	[20, 20, 40]
Angular Momentum	[10, 10, 0.1]	[10, 10, 10]	[10, 10, 10]
Stance Leg Position	[1000, 1000, 1000]	[1000, 1000, 1000]	[1000, 1000, 1000]
Swing Leg Position	[10, 10, 10]	[40, 40, 40]	[40, 40, 40]
Swing Leg Orientation	[1, 1, 1]	[1, 1, 1]	[1, 1, 10]
Arm Posture	[1, 1, 1, 1]	[10, 10, 10, 10]	[10, 10, 10, 10]
Head Posture	[1, 1]	[10, 10]	[10, 10]
Force Regularization	0.001	0.001	0.005
$\ddot{\mathbf{q}}$ Regularization	0.0001	0.0001	0.0001

Table 4.2: Weights of the whole-body controller for balancing, walking, and running behaviors.

utilize the solution from the previous solution as a starting point to solving the problem. Therefore, warm starting is enabled for walking such that the parameters and the solutions can continuously be used in the next optimization, which effectively reduces the number of iterations required until the tolerance for the solution is reached again. Quite often, the number of iterations required are a magnitude smaller than solving from scratch. Therefore, requiring a termination check at every iteration of the solver is beneficial since the solver will then return the solution immediately, as opposed to the default value of 25 requiring additional iterations for marginal improvement. This did allow being able to demand a tighter error tolerance before terminating the solver by two orders of magnitude.

For running, the number of contacts can differ (as there can also be no contacts during flight phases). Different number of contacts means the relative importance of the tasks in the cost function can change. This can translate to the QP's cost function drastically changing from the previous time the QP was solved. Therefore, while warm starting could

Tasks	P/D Gains		
	Balancing	**Walking**	**Running**
Center of Mass Position	[50, 50, 50] [10, 10, 15]	[0, 0, 100] [2.5, 2.5, 5]	[0, 0, 300] [2.5, 2.5, 60]
Body Orientation	[500, 500, 200] [25, 25, 20]	[250, 250, 100] [40, 40, 10]	[500, 500, 200] [40, 40, 40]
Angular Momentum	[-, -, -] [10, 10, 1]	[-, -, -] [10, 10, 10]	[-, -, -] [10, 10, 10]
Stance Leg Position	[1000, 1000, 1000] [100, 100, 100]	[50, 50, 50] [5, 5, 5]	[50, 50, 50] [5, 5, 5]
Swing Leg Position	[100, 100, 100] [10, 10, 10]	[1500, 1500, 1000] [5, 5, 20]	[250, 250, 1500] [2.5, 2.5, 20]
Swing Leg Orientation	[0, 50, 300] [0, 10, 50]	[0, 25, 150] [0, 5, 25]	[0, 25, 150] [0, 5, 25]
Arm Posture	[100, 100, 100, 100] [20, 20, 20, 20]	[100, 100, 100, 100] [20, 20, 20, 20]	[100, 100, 100, 100] [20, 20, 20, 20]
Head Posture	[100, 100] [20, 20]	[100, 100] [20, 20]	[100, 100] [20, 20]

Table 4.3: PD gains for the whole-body controller for balancing, walking, and running behaviors.

Setting	Value
Warm Start	True
Adaptive Rho Interval	50
Scaling	100
Scaled Termination	True
Time Limit	1e-3
Relative Tolerance	1e-5
Absolute Tolerance	1e-5
Max Iteration	1000
Check Termination	1

Table 4.4: OSQP settings for Whole-Body Control.

be beneficial in most cases, it may instead be a pitfall in other cases as the solution may infact, not be similar. For example, when the robot is in flight and both feet are off the ground, the weights and the gains for the Stance Leg Position task in Table 4.2 are reduced to 0.001 resulting in two Swing Leg Position tasks being dominant in the cost function. When one of the feet come in contact with the ground, that leg's Swing Leg Position task and Swing Leg Orientation task will drop to 0.001 and the Stance Leg Position task's weights and gains are restored. In this case, if we naively deployed a warm start strategy, which not only uses the solution from the previous solution, but also the solver's parameters that were being updated throughout the iterations, it was frequently noticeable that the solver could not making meaningful steps towards finding a solution and instead would get stuck in excessively long iterations until it timed out. To address this issue, a "cold start" of the solver had to be done. This was achieved by manually reformulating the optimization model from scratch, setting the previous solution as the initial solution, and then using the starting parameters of the solver as opposed to the solver's parameters at the termination of the last optimization. Through this simple "reset", the transition from flight to contact and vice-versa could just as easily be solved without getting stuck in long iterations (albeit

slower on average than when warm starting because additional iterations in the solver were required).

As an additional implementation reference, although there is an explicit time limit setting of $1e-3$ to ensure the optimization does not delay the control loop, the entire optimization was multi-threaded to fully utilize the on-board computer, which currently is an off-the-shelf mini PC with an Intel Core i7-12700T CPU with 12 cores and 32 GB of RAM. This might seem overly powerful and more than sufficient to run the entire stack in a single thread, but the majority of the development and testing was done on a computer with an Intel i7-7700HQ CPU with frequency scaling disabled and the priorities of the controller set at the highest.

4.3 Locomotion Controller

Overall, while the locomotion framework may look simple as explained in Section 3.2 and shown in Figure 3.4, many times the implementation details make a significant difference the success of the control framework, yet are hidden in these high-level block diagrams. The pseudocode of the locomotion controller can be seen in Algorithm 4, but a more in-depth explanation of each of the steps are explained below.

1. Given the current generalized coordinates $(\mathbf{q}, \dot{\mathbf{q}})$ which includes the floating base information, the kinematics and dynamics information of the robot is updated. This can be done manually by implementing existing algorithms [Fea14], but there are also existing kinematics and dynamics libraries [Smi, Fel17, LP17, LXH18, CSB19] that are capable of computing these quantities as well. Most recent solvers implement the same algorithms, but the noticeable difference between the different libraries or manual implementations is the reference frame and order the linear and angular velocities are with respect to. This minor difference, however, can result in significant increase in efficiency [Fea14]. Therefore, for ARTEMIS, a customized version of pinocchio [CSB19] with a wrapper around it is used for its computational efficiency, ability to run in a

Pseudocode 4 `update_locomotion_controller()`

Require: `robot_model`, `desired_robot_state`, `current_robot_state`

1: `update_end_effector_position_velocity()` ▷ Update what the current foot's position and velocity are.

2: `update_input_commands()` ▷ Update the desired commands based on the user joystick commands.

3: `generate_contacts_for_locomotion()` ▷ Generates the desired contact state for the next control tick.

4: `update_gait_timings()` ▷ Updates the stance and swing leg bookkeeping module.

5: `update_swing_foot_position_velocity_acceleration()` ▷ Solves for the desired final footstep position and desired footstep position, velocity, and acceleration at the next control tick.

6: `update_com_position_velocity()` ▷ Generates the desired CoM position and velocity along the Z direction.

7: `solve_wbc()` ▷ Solves the whole body control QP problem from Section 3.2.6 and Section 4.2.

real-time environment (no dynamic memory allocation), and support for potentially useful features down the road (e.g. analytical derivatives, auto-differentiation). One of the key additions that are included is the inclusion of reflected inertia in the dynamics, which its impact will be further discussed in Section 4.3.1.2.

2. The locomotion controller also consumes a command whose type consists of the desired linear velocity in the X and Y direction, yaw rate, center of mass height, and body orientation. This can come either from an operator sending commands through a joystick or a high-level path-planner. The desired values are then passed as desired references for the whole body controller. Note that as seen in Figure 3.1, the commands come in at 10 Hz, which is still sufficiently fast to introduce undesirable noise or jerky commands. Such commands can make the robot aggressively change its desired footstep positions multiple times through a single swing leg trajectory, which can create

additional undesirable disturbances. To overcome this, low-pass filters on the desired yaw rate and orientation are applied at 10 Hz. For the linear velocities, a slew rate limiter is implemented to constrain a maximum acceleration and deceleration of the robot. A slew rate was particularly important in ramping up/down the velocity of the robot because in practice, the robot could more stably track a gradually increasing desired linear velocity as opposed to a step input. A low-pass filter was inappropriate mostly because a smooth acceleration would result in an undesirable slow deceleration, and a quick deceleration would result in an undesirable high acceleration.

3. Given

   (a) Locomotion mode

   (b) Lift-off percentage

   (c) Locomotion parameters (desired velocities and gait timings)

   (d) Current foot contact statues

   a `desired_contact_state` $\in \mathbb{R}^2$ for the next control tick/step is generated. Note that although the "terminology" used is *desired* contact state, this is the contact state that is sent to the whole body controller when determining which end-effectors should be in contact. So the output of the optimization will generate ground reaction forces for the end-effectors whose *desired* contact state is `True`. For example, in the case the robot is balancing, the `desired_contact_state` is equal to `[True, True]`, which informs the whole body controller that both feet should always be in contact. The logic becomes more complex when contact states for walking and running need to be generated.

4. Depending on which foot (if any) is on the ground, we update the amount of time that has elapsed since a foot came in contact with the ground or since a foot started swinging in the air. This bookkeeping is useful throughout the locomotion stack and is done separately for each foot. After this step, the complete information about a foot's stance or swing state is available at all times through a struct that provides for each foot:

(a) Step Phase: Percentage that a foot is through a complete step which is defined as a collection of a stance followed by a swing.

(b) Stance Phase: Percentage of the total stance time that the leg has gone through. Only primarily used to enforce a minimum stance time.

(c) Swing Phase: Percentage of the total swing time that the leg has gone through.

(d) Remaining Stance Time: Time remaining until the stance phase is completed.

(e) Remaining Swing Time: Time remaining until the swing phase is completed.

(f) Imminent Lift-Off: Boolean indicating that in an ideal situation, a lift-off is next.

(g) Imminent Touchdown: Boolean indicating that in an ideal situation, a touchdown is next.

5. Now that a desired contact status and the feet timings have been updated, next is to calculate the desired footstep position of the swing foot based on Section 3.2.3.1. A desired footstep position is computed by a dedicated module that continuously updates the position, allowing the robot to react as quickly as possible to external disturbances. Simultaneously, a swing trajectory based on Equation (3.42) for the X and Y footstep positions and Equation (3.43) for the Z direction are continuously solved using the last desired footstep position. These three optimizations can then return the swing foot's desired position, velocity, and acceleration for the next control cycle, which is sent to the whole-body controller to be tracked.

6. A trajectory also needs to be generated for the center of mass. Depending on the lift-off percentage, a desired center of mass position and velocity trajectory only along the Z direction is generated based on Section 3.2.4.

7. Given the desired end-effector positions, velocities, and accelerations, the center of mass position and velocity along the Z direction, and the body's orientation, the whole-body control optimization problem is formulated and solved for. This returns desired ground reaction forces which can be used to compute the desired joint torques as in Section 3.2.6 and Section 4.2.

4.3.1 Walking

ARTEMIS is able to achieve robust walking behaviors when the locomotion stack described in Section 3.2 is implemented as above in Section 4.3. This section presents results corresponding to dynamic walking, which includes:

1. The linear velocity that the robot's center of mass can achieve in an unmodeled world.
2. The tracking performance of the swing leg trajectory, which is critical for dynamic stability.
3. The angular momentum of the robot during walking.
4. The robustness of the walking controller.
5. The passivity of locomotion.
6. The cost of transport and Froude number for our locomotion.

4.3.1.1 Walking Velocity

Using the presented approach, ARTEMIS is capable of ramping up its center of mass linear velocity from rest up to 2.1 m/s, making it the fastest walking humanoid as far as we are aware at this time. Considering that an average human walking speed is anywhere from 0.95 m/s to 1.33 m/s [ACR20] for the age group greater than 65 and less than 30 respectively, ARTEMIS is capable of walking up to 221% faster than an average elderly person. A video of the walking over a different range of linear velocities can be seen in `https://youtu.be/1Hy3T12YUzo`.

Using the current approach, there were two ways to achieve fast walking. ARTEMIS had to either step out further or step faster such that its legs can carry the robot forward as quickly as possible. However, this simple modification did not immediately result in a stable, fast walk. In reality, the slew rate limiter, which limits the acceleration of the commanded linear velocity was critical in ramping up the robot's velocity. When the robot was at rest and

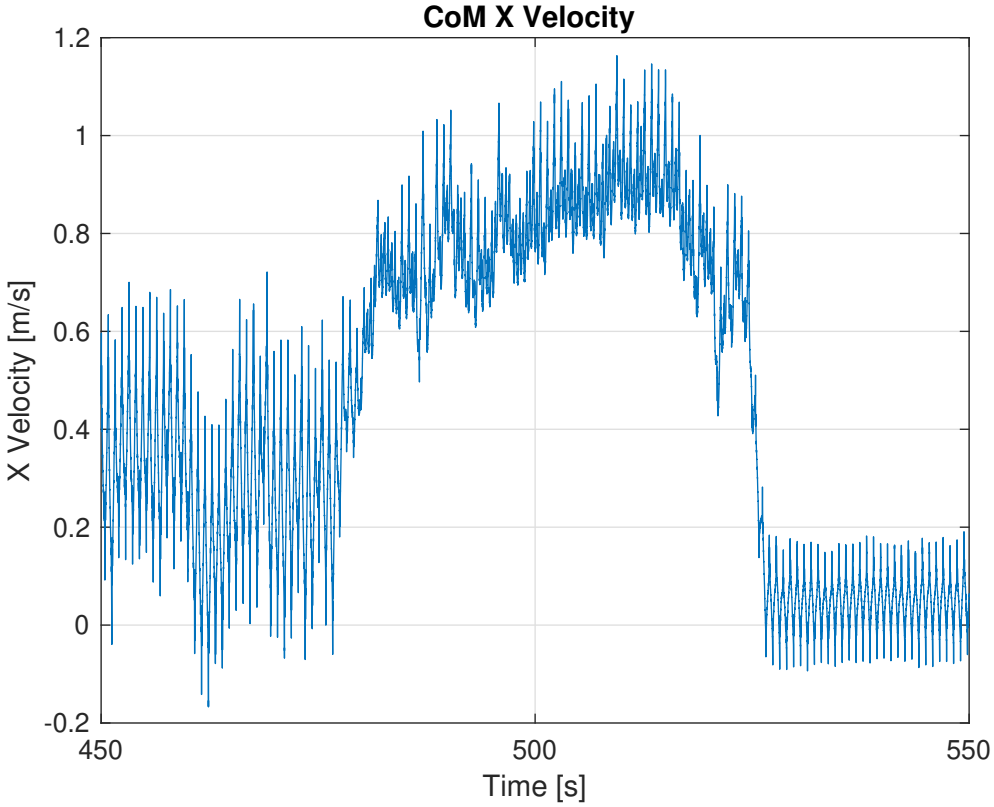

Figure 4.3: Last 100 seconds of walking around the third floor of UCLA Engineering IV building.

an instantaneous forward velocity was commanded, ARTEMIS often hit joint limits in the knee as the legs tried to reach extremely far back to tip its center of mass forward, resulting in triggering the robot's safety mechanisms explained in Section 3.1.4. Additionally, if the operator was unable to input smooth velocity commands, the robot also had a tendency to get unstable. This was because the swing foot continuously modifies its final footstep position based on the desired velocity, which introduces undesirable oscillatory behavior at the foot while it is swinging, because its destination is constantly changing.

With an acceleration rate limited command, the change in the commanded velocities are smooth such that the robot is capable of walking stably and also quickly. Figure 4.3 shows the center of mass velocity of the robot along the forward X direction during the last 100 seconds of the walk around the third floor of UCLA's Engineering IV building (also shown

Joint Name	Reflected Inertia [$\mathbf{kgm^2}$]
Hip Yaw / Roll	0.0490
Knee / Hip Pitch	0.0882
Ankle Pitch	0.0195
Arms	0.0084
Neck	0.0012

Table 4.5: Reflected inertia values used in the dynamics. The values are identical for the left and the right leg and arms.

in Figure 4.1). This is a straight path that the robot ramps up from 0 m/s to roughly 0.9 m/s, sustaining the top speed during this sequence for roughly 20 seconds. The swing times were set at 0.45 seconds, which is slightly faster than the average human walking frequency of 2 Hz.

4.3.1.2 Swing Trajectory Tracking

To achieve dynamic walking, it is imperative that ARTEMIS is able to track the tasks specified in the WBC. One of the most important tasks besides holding the contact foot position, is tracking the swing foot position. If ARTEMIS can track the swing foot position, it will be able to move its legs to its desired position to catch itself from falling. This includes not only moving the foot to some desired X and Y position as computed in Section 3.2.3.1, but also includes being able to clear the foot off the ground along the Z direction.

The desired trajectories that are generated through the optimization explained in Equation (3.42) and Equation (3.43) are found using OSQP [SBG20]. Unlike in WBC, OSQP is used with default settings except that at the start of a new swing phase, the optimization problem is re-created and warm start is applied through the duration of the remaining swing trajectory.

Figure 4.4 shows the current and desired feet positions for the left (LF) and right (RF) foot. Overall, the tracking performance is extremely good, even during this interval from

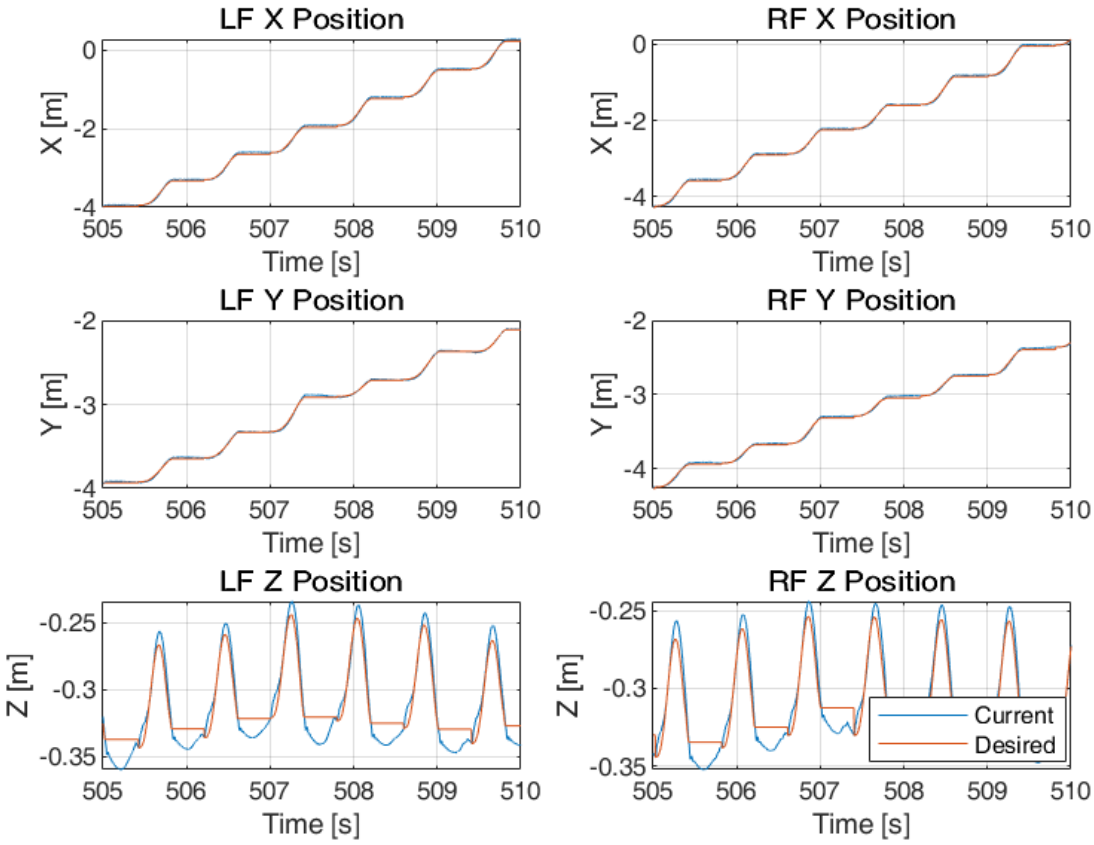

Figure 4.4: 5 seconds of the left and right foot's current and desired positions.

505 seconds to 510 seconds, which is when, according to Figure 4.3, ARTEMIS is traveling at near 1 m/s in the forward direction. This tracking performance was achievable because of a collective effort from feedforward torques, joint PD feedback, as well as considering the reflected inertia of the actuators as previously discussed in Section 4.3. Especially including the reflected inertias across the mass matrix for each joint allowed primarily using the feedforward torques as the input to the actuators, with less than 5% of the final torque contribution coming from the joint PD feedback. The reflected inertia values used in the dynamics can be seen in Table 4.5.

Furthermore, a keen reader will notice that the Z foot position is in the negatives and may

wonder why that is the case when the robot stayed at the third floor. This is the absolute position drift from the state estimator as discussed in Section 4.1. Although this behavior exists in X and Y, it is less noticeable because the robot is actively moving along those axes. Additionally, the desired foot position is a constant when it is not in a swing phase, which is the most noticeable along the Z axis. This is because when a foot comes in contact with the ground, the no-slip condition is assumed, hence the desired foot position when in contact is set to be a constant. However, the current position does dip into the ground which again, is an artifact of the absolute position of the robot being an unobservable state.

4.3.1.3 Angular Momentum

During locomotion, another task that is of interest but an explicit trajectory was not designed for it is the angular momentum of the robot. Angular momentum task is a part of the optimization and it is listed in Table 4.2. Recall that when people walk, angular momentum oscillates around zero. Motivated by this observation, the desired angular momentum for the robot is also $[0, 0, 0]$. Then, the expected behavior is that the arms will make adjustments to regulate the robot's momentum to be zero, while still trying to reach its desired positions.

Figure 4.5 shows the angular momentum of the robot for 5 seconds while it is walking out in the hallway. If we observe the angular momentum of the robot as seen in Figure 4.5, we can notice that the momentum around all the axes are oscillating about zero. When the signals are averaged over a 5 second window, we can see that they indeed are close to zero.

4.3.1.4 Robustness

To make the robot be able to walk around outdoors on uneven terrain without any support, the walking performance must be extremely robust. Good experiments to conduct to assess the robustness of the locomotion framework could be making the robot stay balanced while walking across terrains with debris on the ground or aggressively pushing the robot from side to side. If the robot can recover from these uneven and harsh treatments, it has a good chance of being able to stay balanced and walk around outdoors as well.

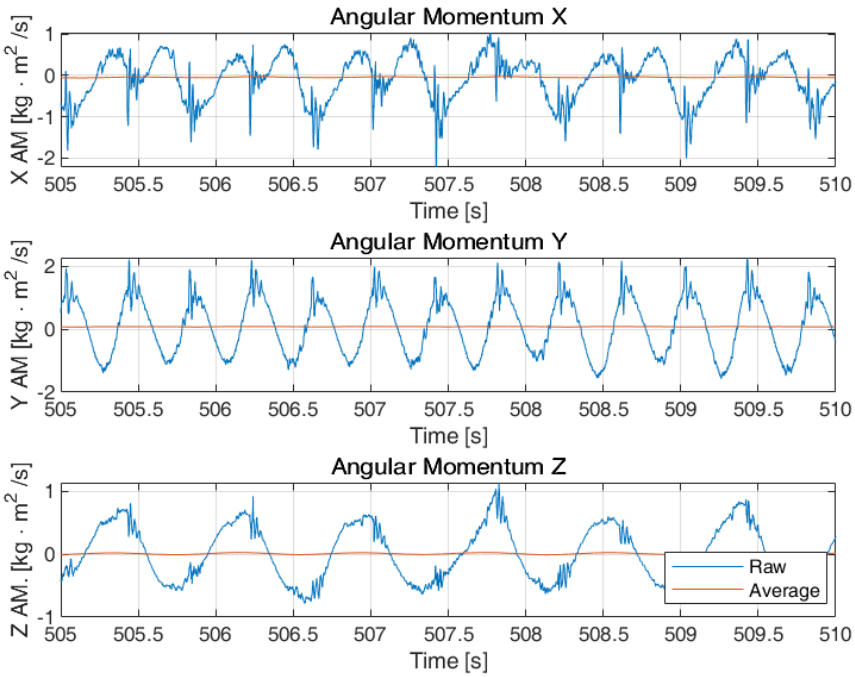

Figure 4.5: Angular momentum of the robot at its center of mass with respect to the inertial frame. Blue is the raw signal computed from the floating base state estimation and dynamics, while the red is an average over 5 seconds.

To evaluate the robustness of the locomotion controller, ARTEMIS was put under the aforementioned two tests. ARTEMIS was pushed and pulled around from the front, the back, and from the sides, as seen in Figure 4.6 and in Figure 4.7. Figure 4.8 shows ARTEMIS, without using any perception data, able to walk on terrain with unknown, randomly positioned debris simply by detecting contact, taking the right footstep, and trying to maintain the center of mass height and body orientation. While difficult to quantify or consistently apply a force in reality, significant pushes were applied to the robot, as can be seen in `https://youtu.be/ZqUGCLM29Vc`.

While a formal analysis on the nonlinear dynamical model is not done in this dissertation, we can still observe important phase plots of the robot to see if the robot is robust to these external disturbances. When the robot is not disturbed and is dynamically stable, we should

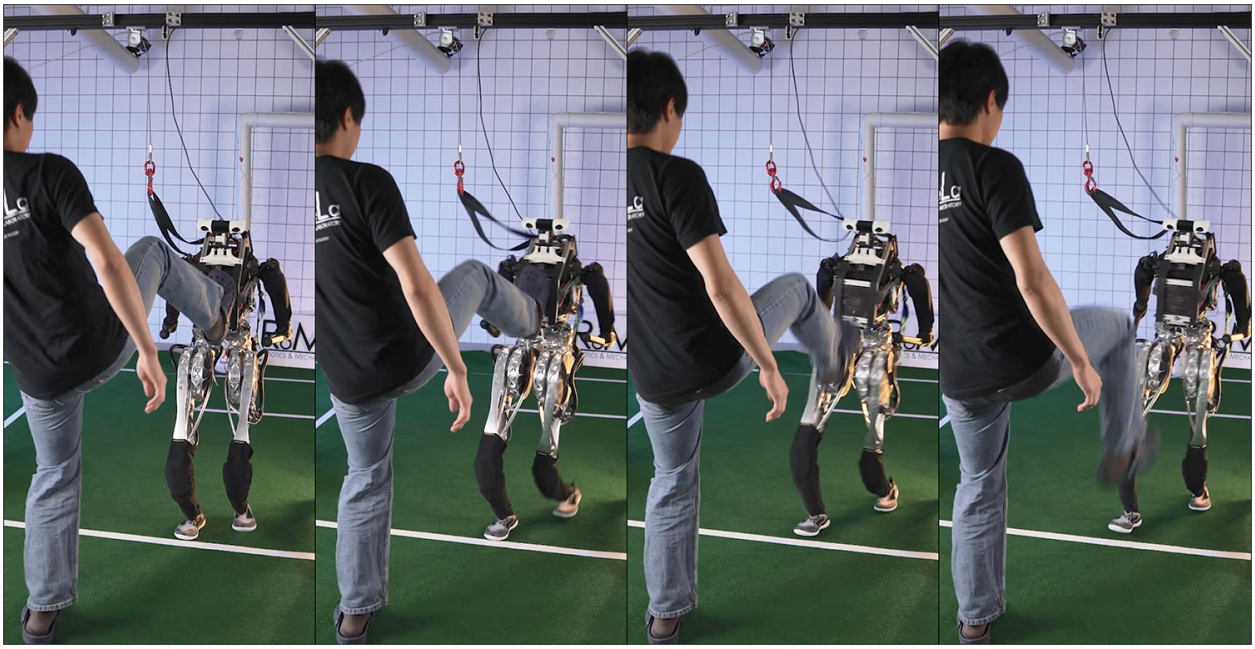

Figure 4.6: ARTEMIS is pushed from the front, causing the robot to adaptively change its footstep position and take a step back to maintain stability.

be able to observe a limit cycle. When a disturbance is applied, the state could diverge from the limit cycle. However, if the robot is robust and able to recover, the state should return back to its original limit cycle.

Figure 4.9 shows an instance where the robot is pushed backwards twice at times 113 seconds and 115 seconds. We can see that the linear velocity along the X becomes negative (i.e. the robot is moving backwards) and the velocity along the Z also slightly becomes positive. Especially the second push nears 1 m/s, which is a significant push considering the robot's inertias.

For dynamic stability and balancing, it is useful to observe the phase plots of the center of mass height and the roll and pitch of the body. If these are stable, the robot should not have fallen over. The phase plots for these three quantities are shown in Figure 4.10, Figure 4.11, and Figure 4.12. The colors on these four plots signify a temporal synchronization across the different plots. The red interval in the curves on the phase plots correspond to the red

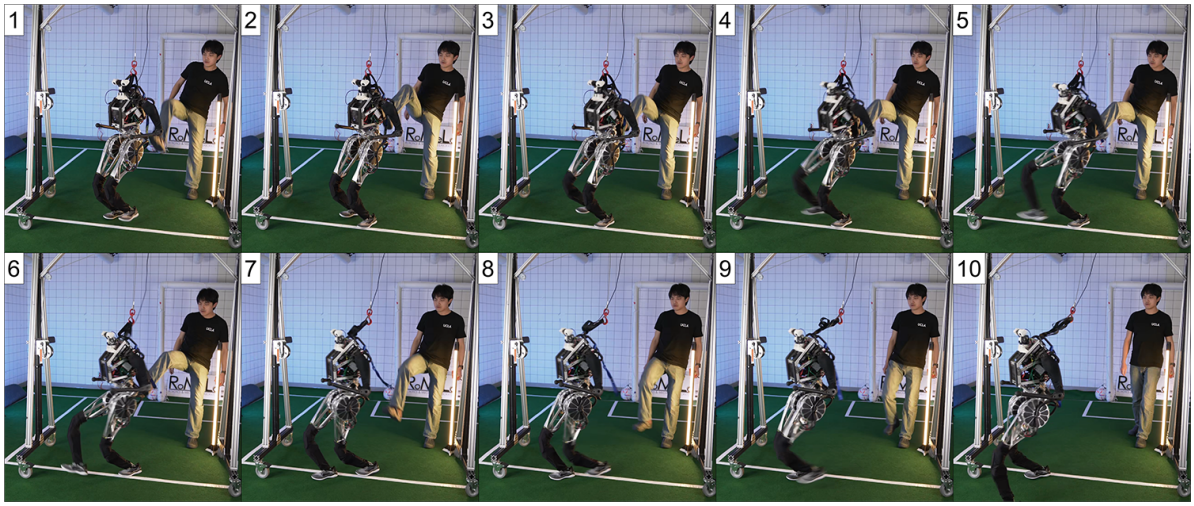

Figure 4.7: ARTEMIS is pushed from the back, causing the robot to adaptive change its footstep position and take a step forward to maintain stability.

section in Figure 4.9, which happened at roughly 111 seconds into the experiment.

When disturbances were applied at 113 s (yellow to light green curve) and 115 s (green to cyan curve), we can observe in the phase plots that the curves diverge from the original limit cycle. However, after 116 s (starting from the blue curve onwards), the curves return back to the original limit cycle. This confirms the robustness of the locomotion controller on ARTEMIS.

4.3.1.5 Passivity

Recalling our assumption on the footstep planning model assuming a passive ankle, it is of interest to see how much the ankle is actually contributing to the locomotion. Even beyond the model we assumed for footstep planning, ankle contribution within the entire system is a metric we can use to observe how "natural" or "human-like" our locomotion is [MK13]. As humans, we pivot about our stance foot. This allows us to take advantage of gravity and inertia during walking [TP19]. In that sense, observing the passivity of a locomotion can expose the efficiency of the locomotion as well as its dynamic stability, because you cannot stay balanced without taking steps when you have passive ankles.

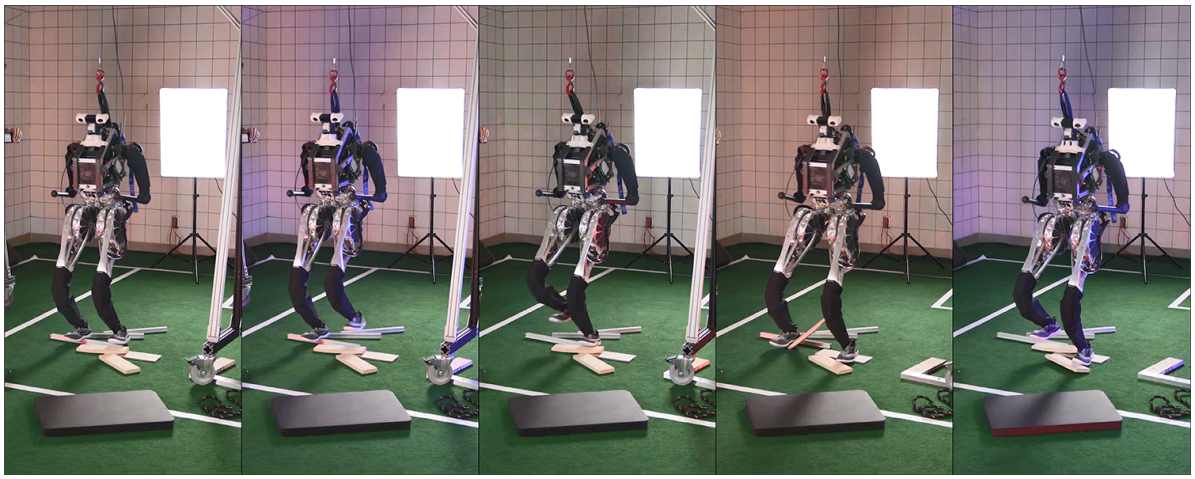

Figure 4.8: ARTEMIS walking on unknown, randomly placed debris.

A measure known as the Passive Gait Measure (PGM) allows us to quantify how passive a walk is [MK13]. PGM is defined as:

$$PGM = 1 - \frac{RMS(\boldsymbol{\tau}_{\text{ankle}})}{RMS(\boldsymbol{\tau}_{\text{all}})}. \tag{4.1}$$

In essence, it is the ratio between the root mean square of the ankle torques compared to the root mean square of the torques at all the joints. If PGM is 1, it means that $RMS(\boldsymbol{\tau}_{\text{ankle}}) = 0$ and there are no ankle torque contributions in the locomotion, hence a passive ankle. If PGM is 0, it means that the ankle torques are the only torques in the current motion. The closer the PGM is to 1, the closer our locomotion is to the model and the more "efficient" it is in terms of utilizing the gravity and inertia (or momentum) for walking.

Figure 4.13 is a histogram of the passivity gait measure at each timestep during the walk on the 3rd floor of Engineering IV. We can see that the majority of the bar is towards the right (closer to 1), making the average PGM of the entire run 0.88. Speaking of "efficiency," there clearly is momentum in the middle of steps (although it averages out to 0 as discussed in Section 4.3.1.3) and with such little contribution from the ankles, gravity is playing a role as well in propelling the robot forward. Speaking of dynamic locomotion, this value confirms minimal use of an actuated ankle during locomotion. As an interesting comparison, humans [MK13] have a PGM of roughly 0.6 during single support.

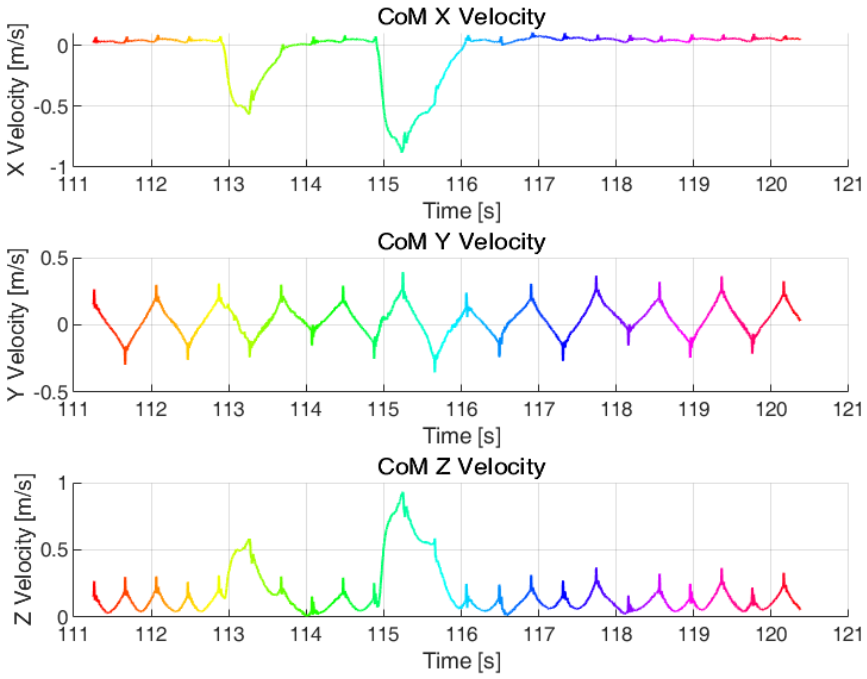

Figure 4.9: The center of mass linear velocity signals when the robot was kicked twice from the front.

4.3.1.6 Cost of Transport

Although dynamic locomotion was the objective of this work, it is difficult to evaluate the performance of the robot compared to existing platforms because humanoids come in all shapes and sizes. A huge robot such as THOR has significantly longer legs than, for example DARwIn-OP, making it easier for a taller robot to walk faster because of its greater stride length. Therefore, it is difficult to compare performance across different platforms, yet there is a continued effort to find common benchmarks that could be applied across them [CGS18, TP19].

However, we can still compare the platforms on the more traditional performance indicators, with Cost of Transport (CoT) being one of the more widely used. It is defined as the ratio between the total power consumption and the product of the weight and velocity.

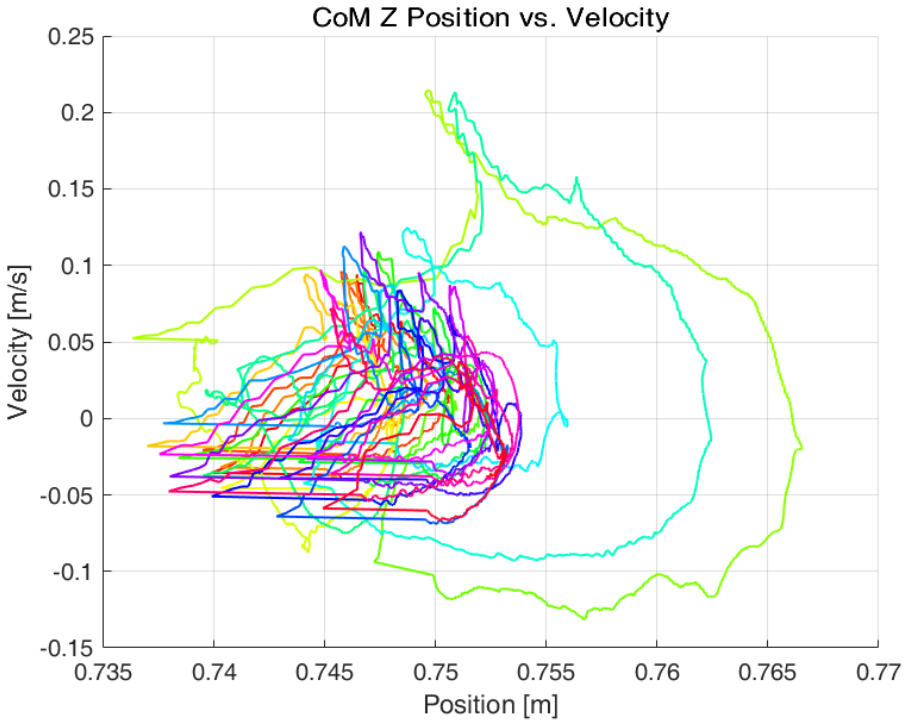

Figure 4.10: The phase plot of the Z position and velocity of the robot.

Effectively, it measures the efficiency of legged locomotion [BCR12]

$$COT = \frac{P}{mgv} \tag{4.2}$$

where $P = \sum \boldsymbol{\tau} \times \boldsymbol{\omega}$ is the mechanical power, m is the mass of the system, $g = 9.81$ and v is the linear velocity of the system.

There exist works that document the CoT of animals [Tuc75] and recently, this has been extended to include robots as well [SWC14]. As evident from Equation (4.2), the lower this value, the more efficient the system is. To compute the total power consumption P, we currently only sum the mechanical power of all the joints.

Figure 4.14 shows a plot of ARTEMIS's CoT for five seconds during the walk around the third floor of Engineering IV. The value oscillates from 0.1 to even 1.2 momentarily. However, when averaged over a window of 5 seconds, the CoT oscillates between 0.3 to 0.4. When only comparing the mechanical power, this is in a similar range as biological counterparts with similar mass as ARTEMIS [Tuc75]. Compared to platforms such as ASIMO (CoT 1.6)

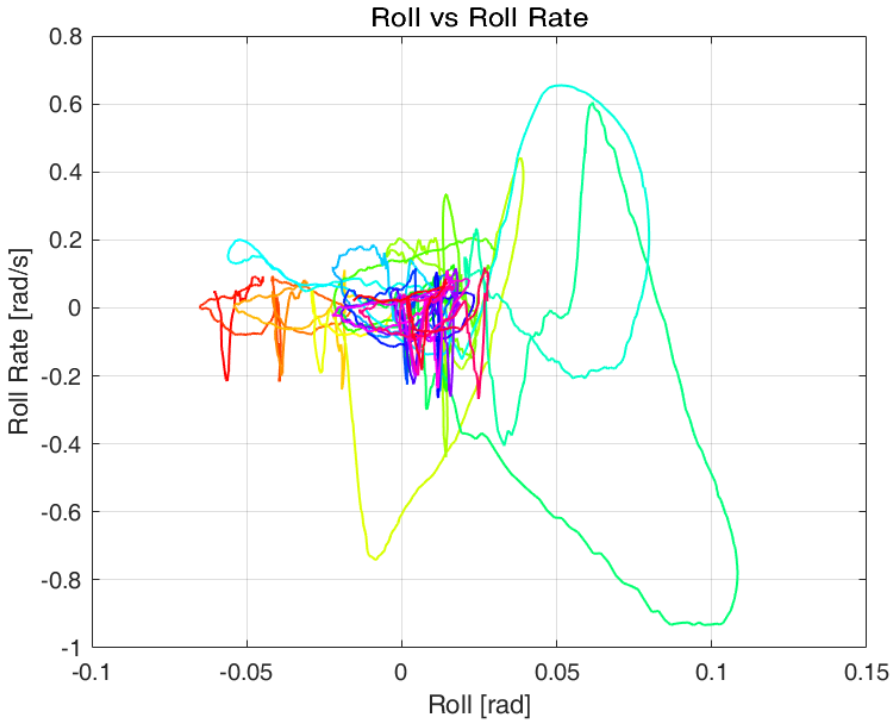

Figure 4.11: The phase plot of the roll and the roll rate of the robot.

[SWA02,SPP11], which is one of the few running humanoids, ARTEMIS' CoT is significantly lower. Infact, it is on par or slightly higher than platforms that were designed with energy efficiency in mind [Wes03, GHM09]. For reference purposes, the torques applied to the robot during those 5 seconds are shown in Figure 4.15 and Figure 4.16.

4.3.1.7 Froude Number

Another interesting metric to inspect for legged platforms is the Froude number. Defined as

$$Fr = \frac{v^2}{gh} \tag{4.3}$$

where v is the velocity of the system, g is gravity, and h is the hip height with respect to the contact point, the Froude number is a dimensionless quantity that allows comparisons between different species (and now robots) while considering their different sizes. Observing Equation (4.3) closely shows that it is also infact a ratio between centripetal forces ($\frac{mv^2}{h}$) to gravitational forces (mg).

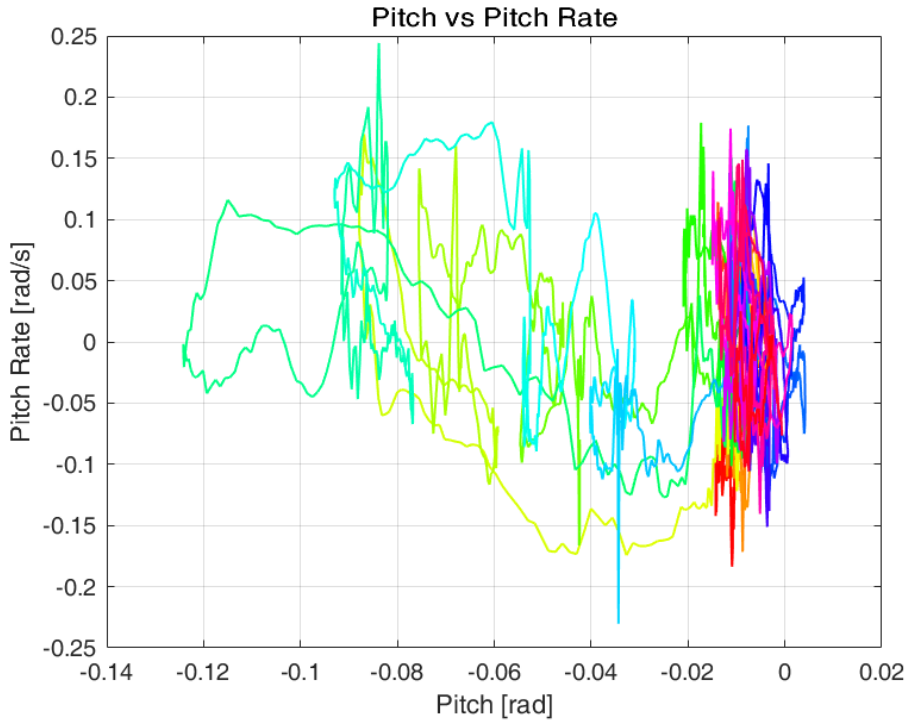

Figure 4.12: The phase plot of the pitch and the pitch rate of the robot.

For record keeping purposes, when ARTEMIS was walking on the third floor of Engineering IV, the Froude number was the highest towards the end when the velocity measurement was nearly 1 m/s. This resulted in an average Froude number of 0.135 as can be see in Figure 4.17. At maximum walking speeds ($> 2m/s$) during speed walk tests, the Froude number averaged at 0.61.

What stands out about this value is that humans have shown to transition from walking to running at a Froude number of roughly 0.5 [Hre95] as there is a tendency to not want to walk at a frequency higher than 2 to 3 Hz [Ush05]. Under these circumstances, the maximum walking speed at such frequencies becomes 2.3 to 2.6 m/s [Ush05]. Assuming the Froude number is a valid metric to compare across different legged systems, the data from humans suggest a possible explanation behind why ARTEMIS is currently only able to walk up to 2.1 m/s.

Currently ARTEMIS can only reliably track swing trajectories when the swing time is

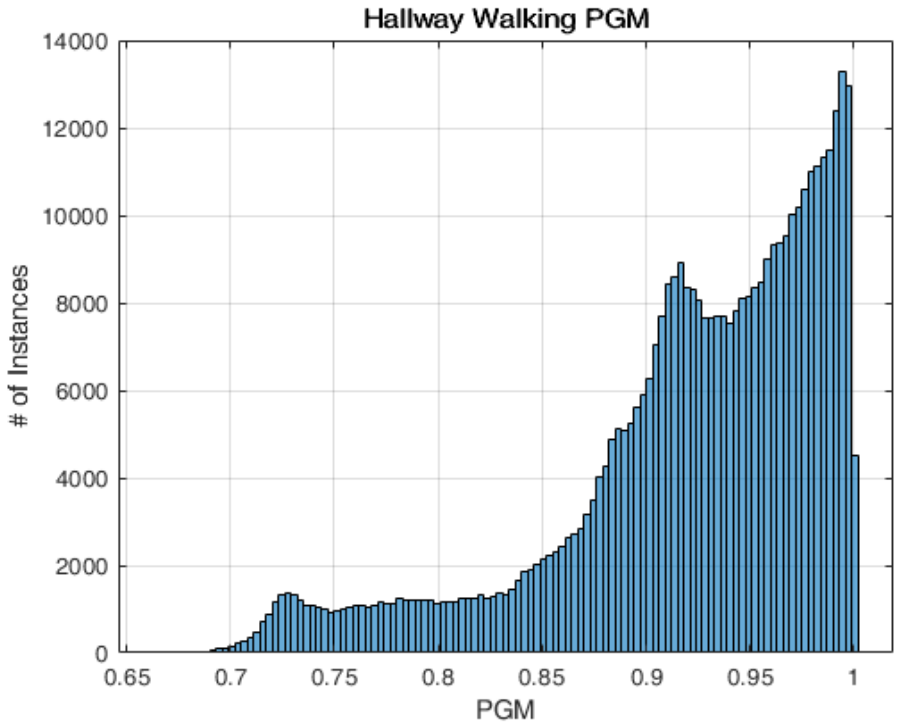

Figure 4.13: A histogram of the passivity values over the course of the Engineering IV walk. The average passivity across the run is 0.884519.

greater than 0.3 m/s. Less than that, the swing duration is too short for the foot to be track well. Given these constraints due to tracking performance, the fastest it can walk at is roughly 3.3 Hz. Recalling human data of being able to walk up to 2.6 m/s at 3 Hz, and that humans, in reality, prefer to transition from walking to running as our Froude number gets close to 0.5, ARTEMIS might be nearing its walking speed limit with the current tracking capabilities. Instead, to achieve faster locomotion, the next step could be to transition from walking to running when it reaches roughly a Froude number of 0.5.

4.3.2 Running

Through modulating the lift-off percentage, ARTEMIS was also able to achieve a gradual modification of locomotion behavior from walking to running. Videos of this behavior as well as when the swing time is modified can be seen in `https://youtu.be/VRHXjum-wO4`

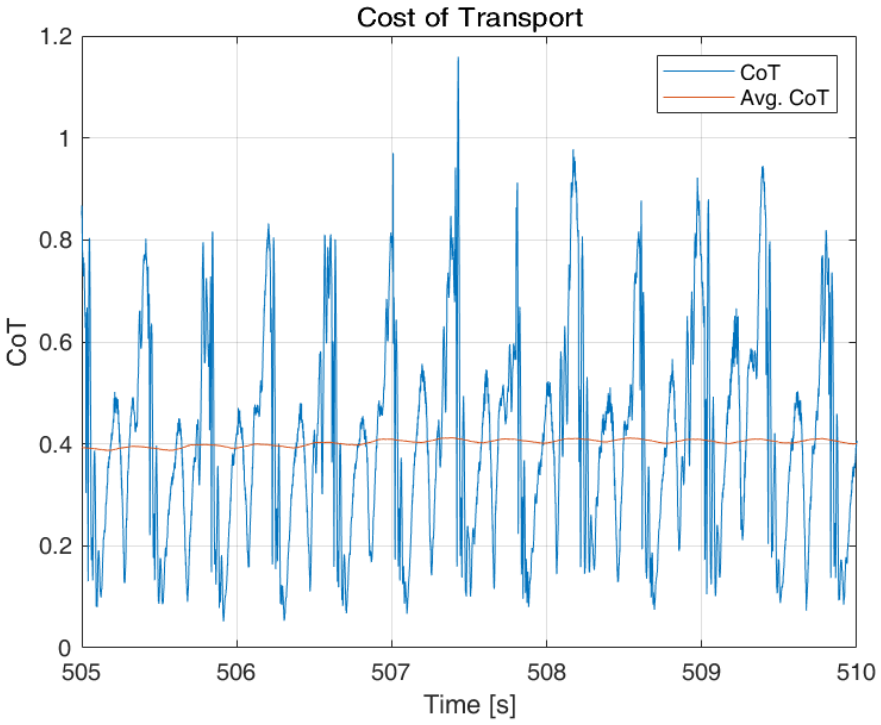

Figure 4.14: Cost of Transport during the run from 505 seconds to 510 seconds.

and `https://youtu.be/0kCqJM4XBuA`. While the analysis from walking can also be applied to running, this section instead specifically highlights data that can only be observed and are used in defining running. These include:

1. The center of mass position and velocity tracking along the Z direction. The center of mass should have a smooth ballistic trajectory in the air followed by a smooth convex shape when a leg is in contact.

2. The contact states of both feet. There should be intervals where both feet are off the ground and not in contact with anything.

4.3.2.1 Center of Mass Position and Velocity

As previously mentioned, depending on the lift off percentage, the center of mass position and velocity along the Z direction are automatically generated to follow a convex shape when

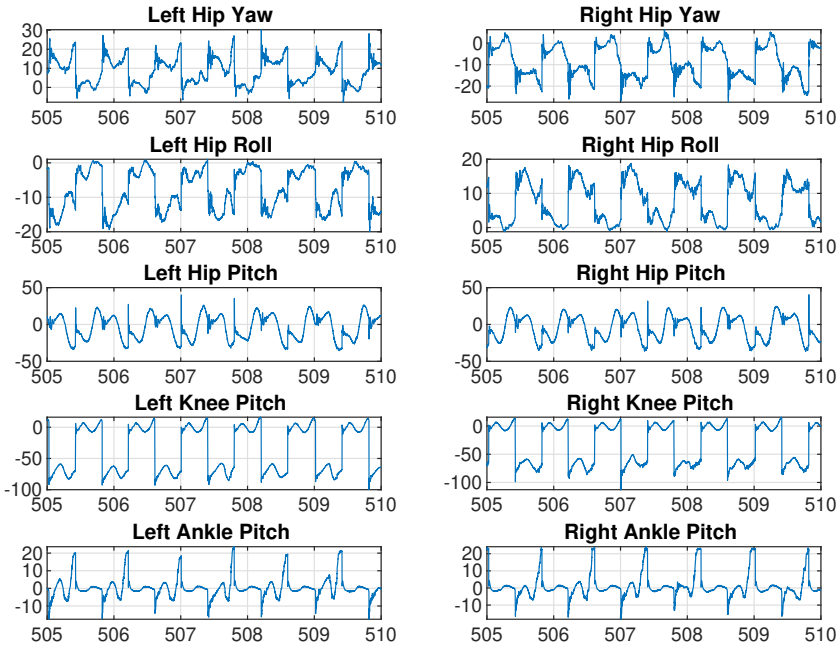

Figure 4.15: Applied torques for the leg joints during walking. Notice that during walking, significant torque is required at the hip pitch and knee pitch. Y axis for all the graphs are torques [Nm] and X axis for all the graphs are time [s].

a foot is in contact, and a ballistic trajectory when the robot is in a flight phase. Velocities corresponding to those position trajectories are also generated for the whole body controller to track.

Figure 4.18 shows experimental data where the center of mass position and velocity were generated for a step with a swing time of 0.4 s and a lift-off percentage of 70%. Hence, the desired duration of the flight phase is 0.12 s. While the flight phase could be difficult to easily identify by only looking at the position plot, we can quickly identify the flight phases in the velocity graph from first principles. When the robot is falling, we should immediately see a negative slope in the velocity graph. Inspecting its duration, we can see that in most situations, the flight phase is just short of 0.12 s and a flat velocity line is generated. This is indicative of an early touchdown (i.e. the robot did not have as long of a flight phase),

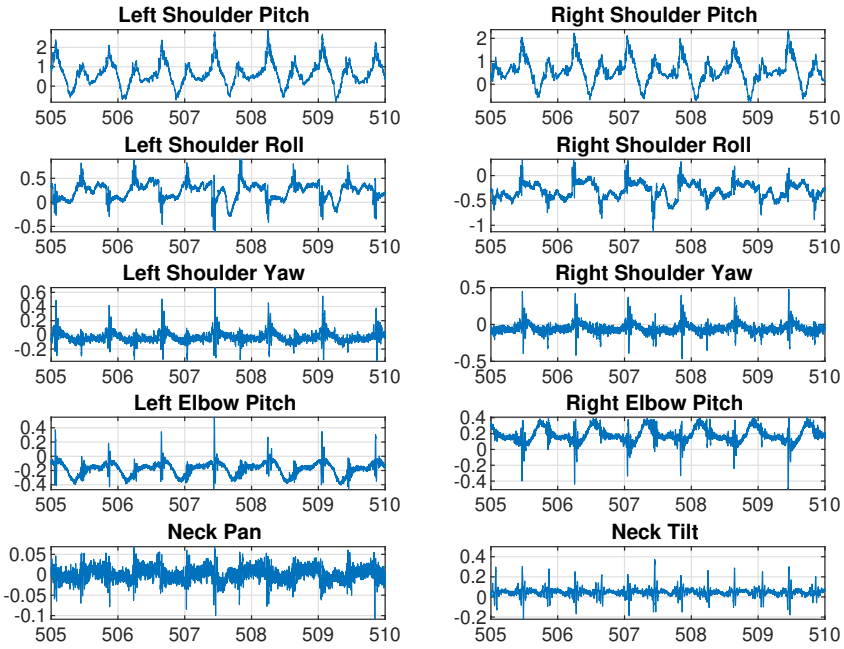

Figure 4.16: Applied torques for the arm joints and the neck joint during walking. It is noticeable that nearly no torque is applied at the arms and the neck right now during walking. Y axis for all the graphs are torques [Nm] and X axis for all the graphs are time [s].

which can also be seen in the flat artifacts on the position plot as well. Because the current approach assumes a symmetric, convex position trajectory when in stance and constant velocities outside the symmetric region, such artifacts are visible.

It is interesting however that despite the robot lifting off from the ground at a vertical velocity greater than what is planned to achieve a desired flight phase, the robot still comes in early touchdown. This could be due to various reasons, but one noticeable issue was the ankle pitch tracking performance as the feet were swinging up and down while the robot was also in flight phase. We assume the pitch of the foot to be parallel with the ground it lifted off of, but if the tracking performance is poor, the feet came in contact with the ground earlier than planned. Another possible issue could again be the state estimator, as earlier

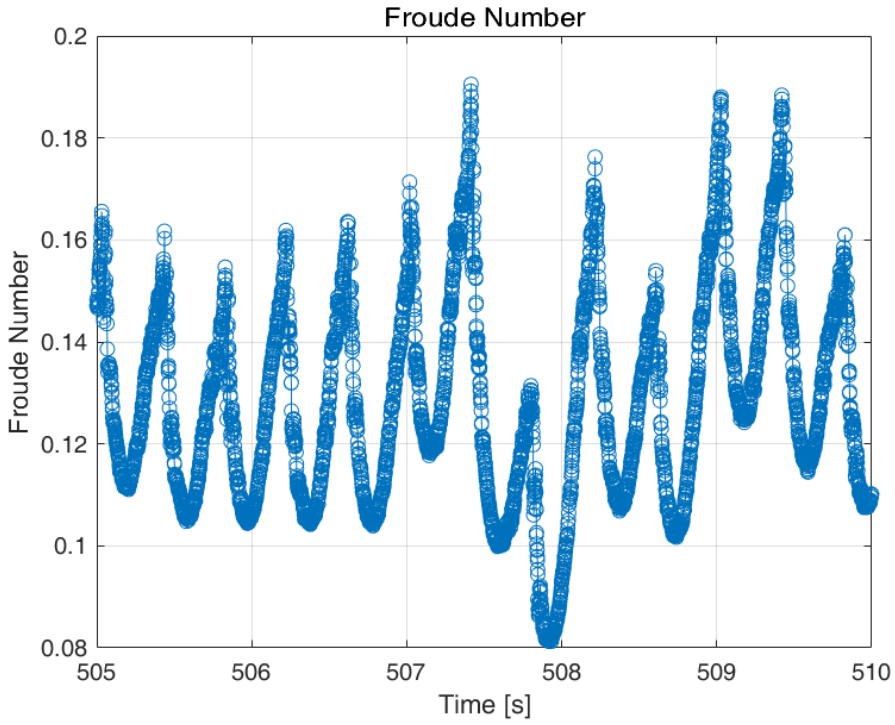

Figure 4.17: Froude number during the run from 505 seconds to 510 seconds.

explained in Section 4.1.

4.3.2.2 Contact States for Running

The second part of running is having intervals in time where all the feet are off the ground and not in contact with anything else. This can be seen in Figure 4.19. When a foot is in contact with the ground, the signal is 1 while when it is not in contact, it is 0. These are raw values from the contact module that determine whether we are in contact or not purely based on the sensor signals or from the estimation.

It is clear that there are times when both the left foot and the right foot's contact status is set to 0, which corresponds with the times when the velocity graph in Figure 4.18 has a negative slope. This confirms that the robot does indeed have a flight phase and is running.

A noticeable anomaly in Figure 4.19 however is the momentary loss in contact after the initial early touchdown. This is a consequence of the foot coming in contact with the ground

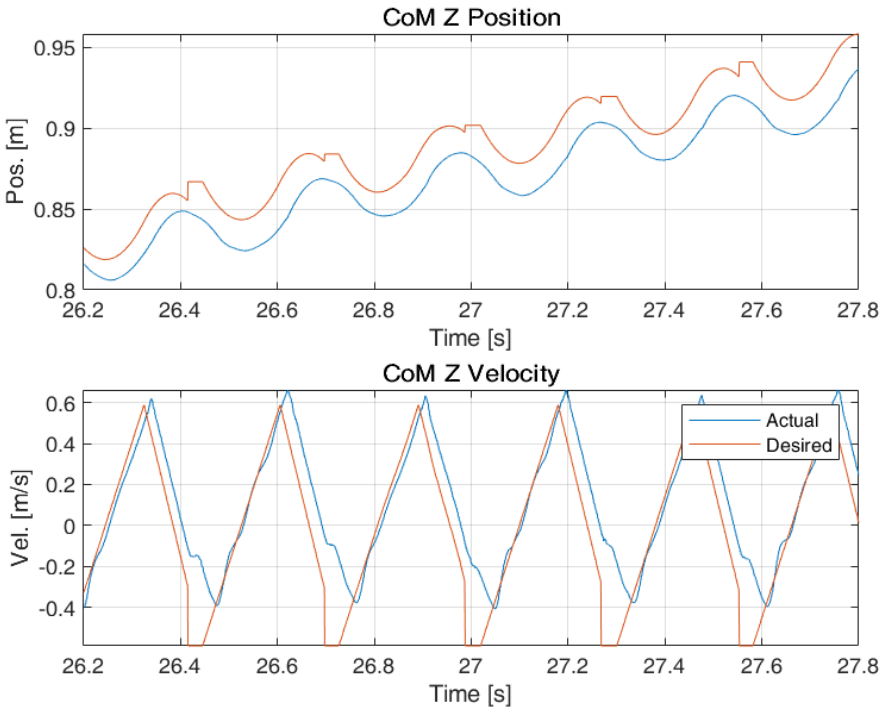

Figure 4.18: Center of mass position and velocity along the Z direction during running.

from a flight phase and then momentarily bouncing off it just enough to make the contact module determine that it lost contact before it determines it is in contact once again. This was left as is because this module is to serve as simply an interface to either the raw sensor values or the contact estimator. To deal with this, a debouncer exists to remove "contact noise" prior to sending the contact status to the locomotion framework.

This resulted in the *first* humanoid that can run and was developed completely in academia to the best of my knowledge. The first running in place can be see in Figure 4.20 which shows a sequence of single support on the left leg (frame 1), flight phase (frame 2), single support on the right leg, flight phase (frame 5), and back to single support on the left leg. Notice how when in contact (frame 3-4 and frame 6-7), the body (i.e. the center of mass) drops down. This is exactly the trajectory that was desired, which the robot follows until pushing off to get back into flight. A video of ARTEMIS running on a gantry can also be seen in `https://youtu.be/NImT1uGdHCU`.

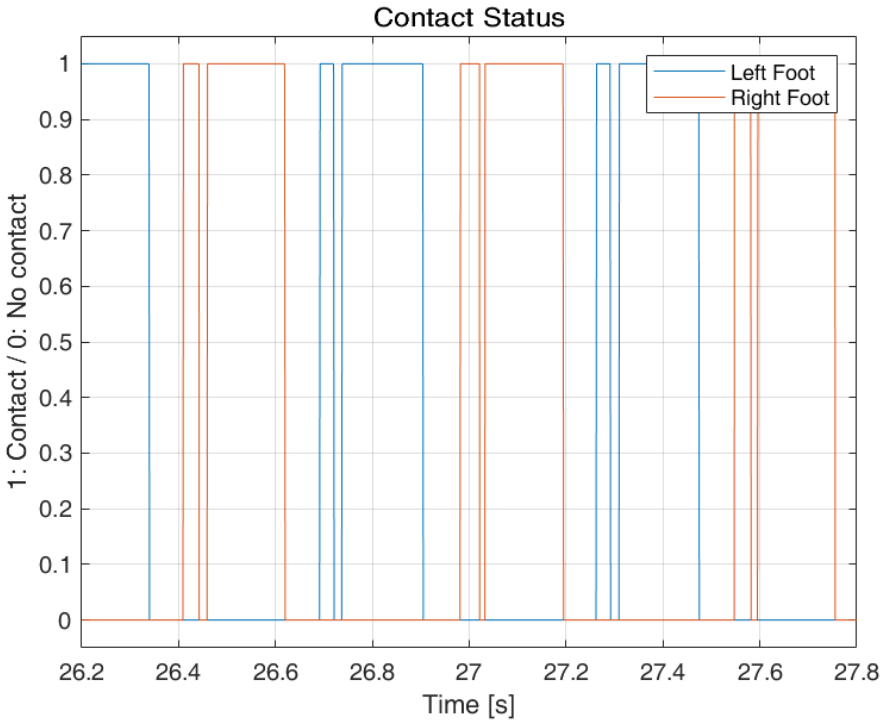

Figure 4.19: Contact status of the left and the right foot. When the foot is in contact, the signal is 1 and when the foot is not in contact, the signal is 0. Each foot come in early contact with the ground, momentarily losing contact as the foot bounces. However, this is filtered out by the debouncer.

Lastly, as an interesting comparison against walking, we can compare the torques applied at the robot. One clear difference between walking and running is that during running, the robot is colliding with the ground at a higher velocity than when walking. To dampen this behavior, we designed a symmetric convex center of mass trajectory. During this sequence however, the knee pitch actuators exhibit close to 150 Nm of torque during stance as seen in Figure 4.21 (compared to ranging from 50 to 100 Nm in Figure 4.15) to successfully decelerate the body and change its velocity direction. Additionally during running, the body would also noticeably rotate about its body's Z axis, as little can be done to correct its yaw orientation while in flight. Therefore, the majority of the correction had to be done while a foot was in stance and this can be seen in the hip yaw actuators' torques, which are significantly higher than those in Figure 4.15.

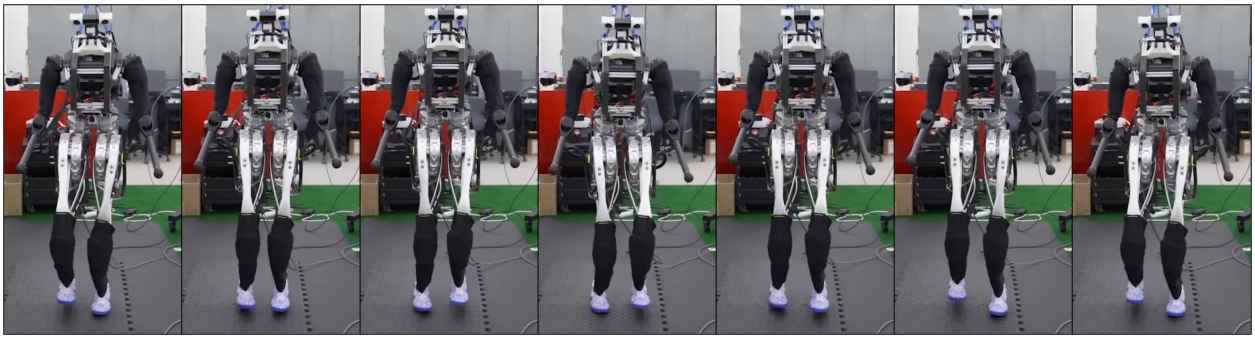

Figure 4.20: Snapshots of ARTEMIS running in place. Frames 2 and 5 correspond to the robot being in flight phase. Frames 3 and 4 show how the body drops when in contact. The same can be see in Frames 6 and 7. Note that ARTEMIS is also capable of wearing ordinary shoes where here, ARTEMIS is wearing the Nike Zoom Mercurial Superfly 9 Elite CR7 FG soccer shoes.

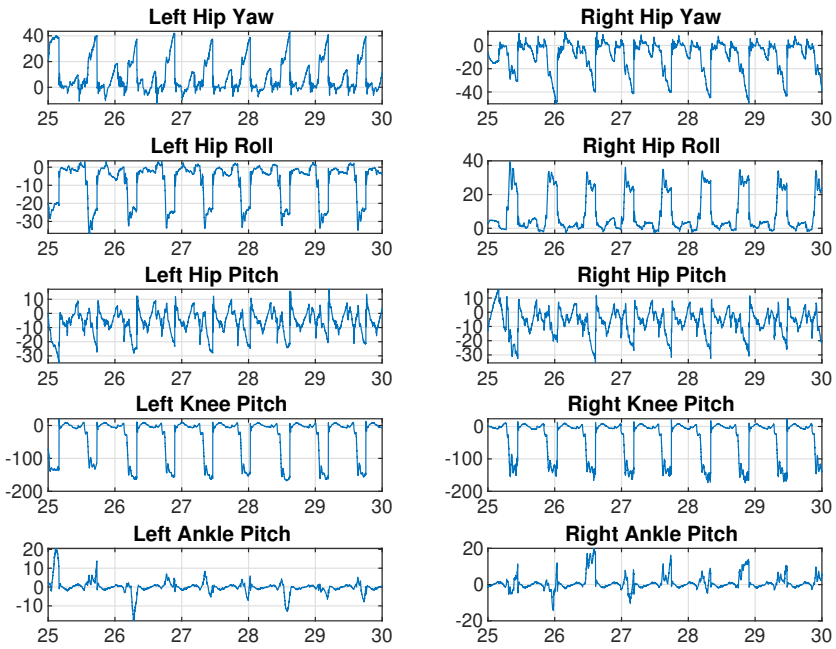

Figure 4.21: Torques applied at the legs while running. Y axis for all the graphs are torques [Nm] and X axis for all the graphs are time [s].

CHAPTER 5

What's Next

5.1 For ARTEMIS

5.1.1 Highly Dynamic Behaviors

The initial motivation behind ARTEMIS was a platform that can do hyper dynamic motions. This dissertation focused on the first "step" of this effort, which is the ability to robustly walk and run, which in the case with ARTEMIS, is a controlled repetition of falling and catching itself. Now, to fulfill the desire to achieve hyper dynamic motions, the task of actually generating these behaviors still remain.

My initial efforts were to solve a nonlinear trajectory optimization problem to include the complete dynamics of the model. These efforts, unsurprisingly, had varying degrees of success because of multiple factors:

1. The problem is nonlinear, which often makes finding a feasible solution a challenge to begin with.

2. Finding a solution can also take anywhere from tens of seconds for simpler problems to tens of minutes for more complex behaviors, making the iteration time consuming.

3. It requires expert knowledge of the platform's limitations and understanding of the desired motion to make it work on the platform.

Using the above approach, solutions for simple jumping and turning 180° sequences or front/backflips are currently achievable. However, since the ARTEMIS project was originally

planned to realize hyper dynamic motions on hardware, the goal is to still achieve non-conventional jumps and twists in the air on hardware.

In that regard, two future areas of investigation to achieve highly dynamic behaviors include:

1. Generalizing the motion generation pipeline such that it does not require expert knowledge and experience with the platform, which also opens up possibilities for generalizing to different platforms.

2. Developing robust controllers that can track potentially complex and rapidly changing trajectories.

5.1.2 Path Planning

ARTEMIS is capable of robustly walking and running to places given an operator's commands. This work has shown that even without perception data and in presence of significant obstacles on the ground, ARTEMIS is capable of staying balanced and catching itself from falling when it steps on an unexpected debris. This is because of the reactive nature of ARTEMIS' locomotion stack.

The next step forward then, is to remove the operator from the loop. Currently, ARTEMIS can robustly move from point A to point B, but which path to take still remains an unfinished task. As simple as it may seem where a naive thought might be to just command SE(2) commands assuming ARTEMIS is a wheeled mobile platform, in that case, it would be implementing a solved path planning problem to the robot.

However, to stay better aligned with the goal of making ARTEMIS a hyper dynamic platform, path planners that also take into account of ARTEMIS' hardware capabilities could be a challenging, yet intriguing problem to tackle. Rather than running down a narrow alley and turning 90° in place at a right-angled path while maintaining the body's roll and pitch angles, why not roll about the body and walk along the walls to re-direct the momentum as opposed to lose it by stopping and turning? For platforms that are not capable of such

hyper dynamic motions, it would be difficult to imagine planning such a path in the first place. With ARTEMIS, its locomotion performance on contact surfaces whose normal vector is similar to the gravity vector suggests that robust locomotion could also be possible when the terrain's normal vector is significantly different to gravity. This is not only limited to ARTEMIS, but also opens up an exciting next step in the field of path planning for hybrid systems.

5.1.3 Perception-based Locomotion

The current work is limited in that ARTEMIS is unable to walk and run across terrains that require *precise* footstep placements. This includes stairs, or in a more general sense, terrains consisting of stepping stones, defined as the *only* regions that the robot can place its feet. To accomplish this task, the first requirement is perception data to inform the robot of safe regions that the feet can be positioned, which can be obtained from either the stereo camera on the head or the two located at its body as seen in Figure 5.1 and Figure 5.2.

After the perception data becomes available, the next challenge is to find the right balance between placing the feet in safe regions (which are non-negotiable to prevent the robot from falling), and still making the robot follow a path. This could be a challenge because due to the lack of a degree of freedom in ARTEMIS' leg, it is limited on where it can place its swing foot while still staying balanced on a single foot and not going unstable with the next touchdown.

To overcome this limitation, a longer horizon footstep planning can be helpful, where similar to the one-step ahead footstep planning, an N-step ahead planning can be done in tandem with the available perception data. This would allow the robot to prepare *in advance* before coming to a situation where the foot must land in a safe region.

An additional area that perception data could be beneficial is in state estimation. Currently, the position states of the robot's floating base are unobservable, causing significant drift in the world frame. If perception data is fused with the state estimation in the context of localization, all the states become observable. However when localization is not needed,

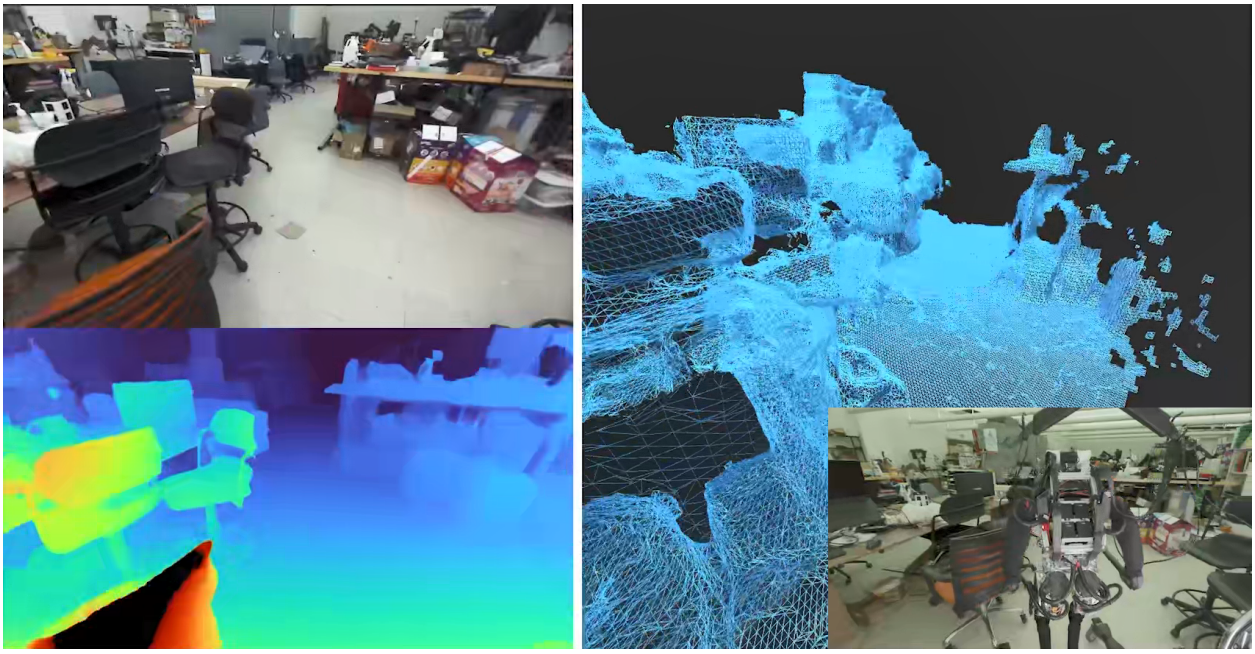

Figure 5.1: Stereo vision from the ZED 2 camera on ARTEMIS' head can be used for perception-based planning.

we could instead use perception data only along select axes to prevent drift from happening along those axes. An example of this fusion that could be done immediately is the floating-base Z position when just walking on flat ground. When the robot is walking along the hallway, its Z position in the world frame continues to move up and down due to a combination of drift, as well as early touchdowns / late touchdowns, which makes the estimator think it is stepping up / stepping down, as seen in Figure 5.3. By using the distance from the body to the contact surface —which should minimally be changing when walking mostly on flat ground —as another measurement information in the estimation, a greater subset of the floating base's pose and velocities can be better estimated.

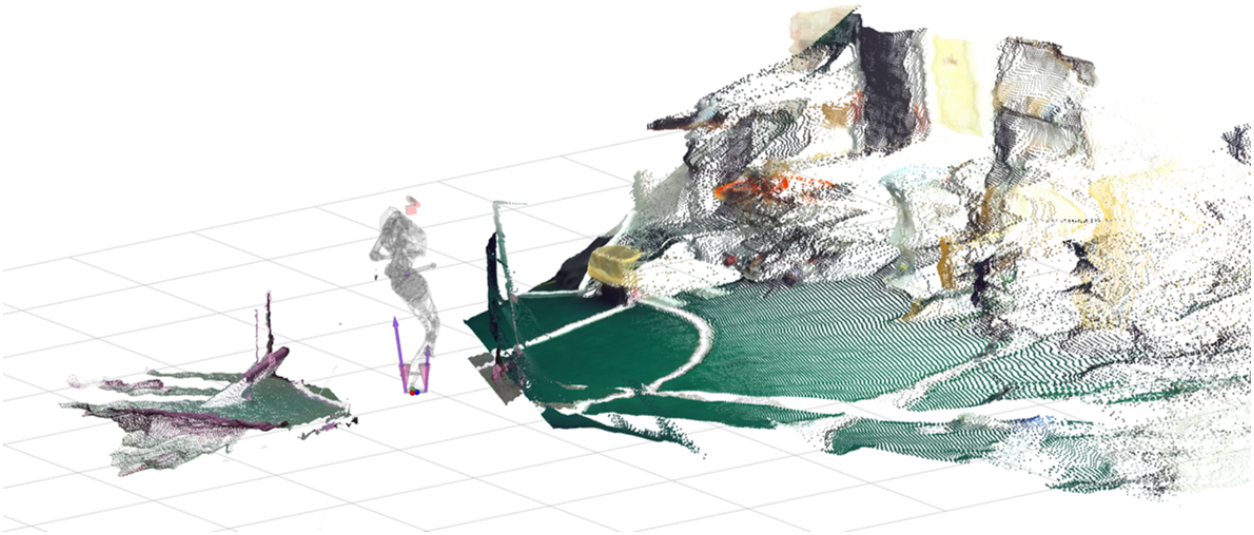

Figure 5.2: Stereo vision from the ZED 2 on the head and the two Intel RealSense D435i stitched together to generate a point cloud of the surrounding environment.

5.2 Beyond ARTEMIS

The learnings obtained from the development of ARTEMIS, a humanoid that can walk, run, and also achieve hyper dynamic motions in the future, go beyond just humanoids. Below are some potential areas of work that are less directly related to ARTEMIS, but are areas of potential future collaboration and work that researchers from other fields could not partake in.

5.2.1 Modeling Contacts

Personally, one of the biggest lessons learned throughout this work was experiencing first-hand the difficulty with realizing ideas on a physical hardware, making hardware implementation a significant part of this work as well. During this process tho, it was also evident that testing in a simulated environment is difficult to gauge the success of a locomotion controller, mostly because many times it took significant work to get it also working on

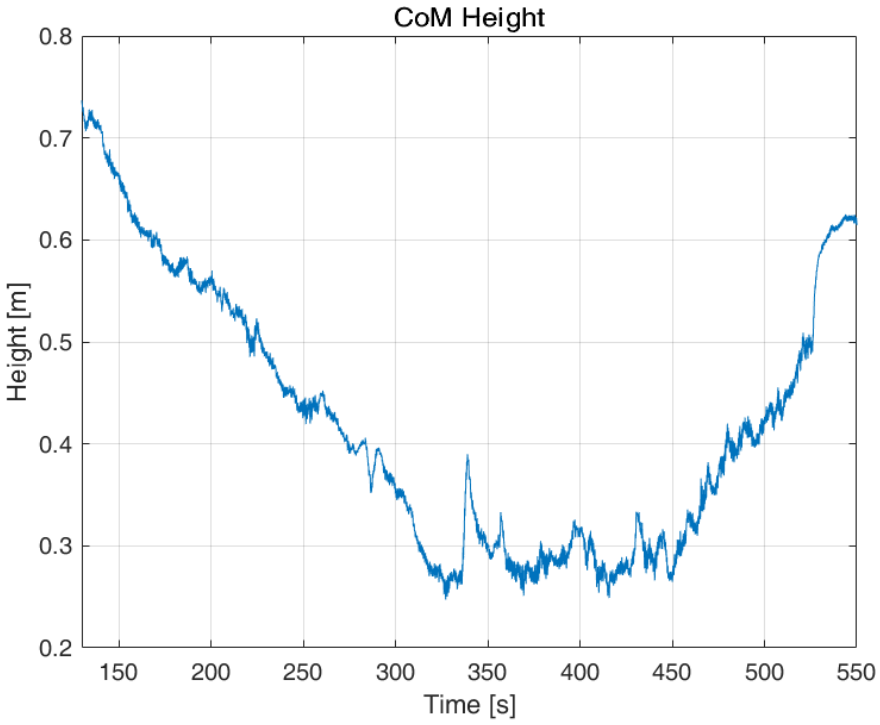

Figure 5.3: Center of mass height during the walk on the third floor of Engineering IV. Although the robot's CoM position relative to the ground was consistently at roughly 0.73 m, because the absolute position of the robot's base frame drifts, we can observe that the center of mass Z position in the world frame also drifts downwards during the middle of the run and climbs back up.

hardware. However, interestingly, the difficulty of stabilizing the system between different simulators also existed.

Simulators will be wrong, but they are supposed to be helpful. Yet, many times, that was not the case based on the amount of effort required to transfer a working controller from simulation to reality. Hence, it may be worthwhile to evaluate why this may be the case, because the gap between simulation and reality is often discussed in the learning communities, but is considerably less highlighted in robotics.

More specifically, one of the most notable differences between the two worlds is how contacts behave in the real-world, compared to how they are modeled in simulation. Even

across the different simulators, the way they model contact differs [KSJ08, Tod14, HLH18]. For applications that experience repeated contacts, such as with legged robots or humanoids, the inevitable inaccuracies in the modeling could be exacerbated in the resulting behavior of the robots. This was observed when additional tuning was required just when switching between different simulators for ARTEMIS.

In fact, this issue could be a recurring issue in other fields of robotics where collision checking is important, such as the field of dexterous manipulation and grasping. Although repeated engaging and disengaing of contact points will be less than for a walking robot, the sheer number of simultaneous contacts occurring to grasp onto an object will be more. An inaccurate simulation could inadvertently signal contact or enforce contact constraints when in reality, that might not be the case, resulting in the object being dropped.

In that regard, based on my experiences throughout this work, a continued effort into the development of simulation environments and contact modeling, where particularly contact between one or more bodies with high acceleration are considered, could be beneficial to the robotics community in the future. It may be true that a single solution does not exist as the contact problem is a trade-off between computational efficiency and solution accuracy. In that case, even a burgeoning benchmark or community that investigates the applicability of the different approaches to different applications could significantly help guide future researchers in choosing a more realistic environment to simulate and iterate their control approaches before deploying on hardware.

5.2.2 Learning and Control Intersection

On a similar note, it is difficult to perfectly model everything and testing on hardware all the time is a luxury most researchers cannot afford. However, successful state-of-the-art approaches in robotics still heavily rely on model-based control which, as its nomenclature suggests, requires some form of a model at the end of the day. Through ARTEMIS, what I learned about model-based control was how sensitive the control actually is to the model. Changes in the center of mass position of a single rigid body just by centimeters in the rigid

body chain prior to calculating inverse dynamics had drastically different behaviors in the robot even just for balancing tasks.

While there exists works that consider robustness [KSM20] or stochasticity [LDT16], another approach could be adopting learning-based approaches within the pipeline [ACT22]. While recognizing that the model will never be perfect, one area of future work that would be beneficial in any robotics field that relies on a model would be a continual learning of the model. We may think that learning of a model can be more effective in systems whose model could indeed be rapidly changing [CAN21]. However, while it may not be as rapid, even systems such as ARTEMIS, which is supposed to be a rigid body system, did face gradual changes on the hardware over time. Compared to when it was first assembled, the joints in the system gradually became "smoother" (resulting in a change in the joint friction) and the backlash on the actuators also worsened. These changes overtime are captured in the logs that were continuously recorded and could have been used for better modeling of the robot.

Additionally, nonlinearities in the models often prohibit their usage in online planning and control because of the computational burden associated with them. Although efforts to speed up nonlinear optimization and control for usage in an online fashion exist, one approach could be to adapt the linear models which are constructed using data-driven or learned methods to continue to enjoy the plethora of existing methodologies in linear control theory. This could be a more incremental step towards end-to-end learning-based control approaches as the intermediate steps can be more modular and interpretable when based on measuring the amount of complexity between the input and the output of the system.

5.2.3 Human-Robot Interaction (HRI)

With the stability in balancing and locomotion that ARTEMIS provides, along with its adult-sized anthropomorphic body, ARTEMIS could also be an intriguing platform to utilize to reach out into topics beyond hyper dynamic motions, with one of them being human-robot interaction. The first component of that could be the reception of such dynamically moving

humanoids in close proximity to us. As robots become more ubiquitous in our society, how it will interact with humans such that its presence is amicable rather alarming is an inevitable task to be solved. Robots will come in all shapes and sizes and understanding what can work in the vicinity of humans (beyond just functionality) will be critical in their adoption.

Based on my experiences through 8 years of live robot demonstrations of nearly 20 different robotic platforms, the designs and interaction methods that people were attracted to were mostly universal. The robots that were adored 8 years ago are still adored the most today (e.g. DARwIn-OP, LARA), while platforms that at least I expected to have less attention (e.g. BALLU) received just as much interest from the general public through close-up interactions (including physical contacts). With ARTEMIS, when it was first demonstrated to the general public through the ANA Avatar XPRIZE competition in 2022 and more recently across the campus in the UCLA, it received mixed responses from people.

What is particularly interesting, especially between DARwIn-OP, LARA, and BALLU, are that clearly looks are not everything. As DARwIn-OP and LARA takes an anthropomorphic shape with a cartoon-ish face, its looks could be a dominant factor in their welcomed reception. However, BALLU does not have noticeable features. Possibly, the words that people generally associate with "balloons", such as *soft* or *fun*, could be the ingredients for its popularity among people. With ARTEMIS, personally I feel that it commands attention (regardless of whether it is a positive or a negative reception) when its foot swing times are relatively faster than a human (under 0.5 seconds), whereas when it is closer to that of a human, it is easy to forget that it is there (which again could be good or bad depending on the situation). In that regard, evaluating the correlation between the speed that robots move and people's perceptions on different feelings such as safety or intrusiveness could be an interesting future research direction. Potentially interested researchers could easily use a turnkey platform like ARTEMIS to conduct tests and gather data to get further insights on how these robotic platforms could seamlessly be a part of our lives. What type of costs that consider interaction or emotion, which are completely different to locomotion trajectory costs, when generating motions to seamlessly blend in with people could be another inter-

esting research direction. Additionally, as the current platform lacks components purely for aesthetics, this too could be an area where researchers in HRI that are more focused on industrial design could delve into. At what point do some of the aesthetics start to get close to the "uncanny valley," whether that uncomfortable-ness purely stems from aesthetics, and how these appearances could generalize to beyond humanoids while not looking like your everyday home electronics (e.g. looking like Baymax as opposed to a walking computer) are some of the numerous questions that a platform like ARTEMIS could be utilized to help get closer to a better understanding on interactions between a robot and a human.

5.2.4 Synthesizing Motions

One of my learnings from ARTEMIS is that as the work was on humanoids, a significant amount of time was spent on understanding what humans do, whether it is how we walk, run, swing our arms, or even just simply clap. It is incorrect to assume that what we do should all transfer over to a robot. However, what could be claimed is that we can try to better understand what humans do and that certainly can inspire the design of controllers, gait patterns, arm motions, and even regulating angular momentum to zero. These experiences make me think that better understanding the motions we generate can be beneficial even beyond ARTEMIS.

Better understanding behaviors or intents from motions could be beneficial in multiple fields. It could be used to understand what is coming next for trajectory optimization problems to utilize them in the field of model predictive control. Equally they could be used in an HRI context to properly respond to how people are behaving. To generate stylistic trajectories that give a sense of elegance in the motions, which could make the difference between a mime or a dance, being able to decompose the core properties that govern the motion rather than just simply space and time, could be useful.

For example, the scale at which motions happen and at what speed, can imply what the ensuing motion will be. Additionally, the differences in the spatial and temporal aspects of a motion can also signal emotions and intents. The legs crouched down and then pushing

off the ground with the arms also reaching upwards can be a sequence of motions to take before a big jump to reach high into the air. Slight differences in this trajectory's spatiotemporal properties could make the motion signify someone jumping because of joy or someone being surprised. Identifying the features that distinguish a jump of joy as opposed to a surprised motion could instill more life into humanodis and other robots used for HRI and animatronics.

We could experience the upside of such research in our daily lives as well. If image data could be solely used to understand these motions, surveillance cameras in our homes could take preemptive action by detecting suspicious activity and alarming the right people before an incident happens rather than simply logging or reacting to a situation after it has occurred. In situations where the information can only be primarily passed through motions, such research could be useful as well. For example, rather than people manually conducting traffic control when there are incidents on our streets, manipulator arms could do it with the right spatiotemporally modified trajectories to properly inform passing drivers.

Therefore, such research would be exciting to pursue in the future from both a continued motion generation research for HRI or humanoid platforms like ARTEMIS, as well as for the deployment of such technologies into our everyday lives.

CHAPTER 6

Conclusion

With many other labs in academia, research institutions, and now even companies developing their own humanoids, my instinct is that these platforms could be in a similar situation that quadrupeds were in late-2017, where hardware and software for humanoids could be converging to a similar style. This could lead to a proliferation of this technology not too distant in the future, just as we are witnessing a growing number of quadrupeds being productionized by multiple companies. With humanoids, their existence in our everyday lives could be an inevitable future because they could provide unparalleled flexibility in terms of the work they do. While I do not think the timeline will be anything close to what we are seeing with quadrupeds, I do believe that this could be the dawn of a new era for humanoids. Given such a situation, my hope is that this dissertation contributes to the beginning of a proliferation of these new platforms.

To achieve such goals, we also need more passionate and interested engineers and researchers to join in on this journey. In hopes of making that process easier, this dissertation tried to be informative to beginners by starting out in Chapter 1 with a brief history of the field of legged robots and humanoids. This discussion includes relatively recent history as well, as it also covers the direction that modern platforms are headed towards in terms of both design and control. Afterwards, we directly dive into our platform in Chapter 2 to explain its features that embody these new design ideas, that will also enable modern control approaches to be applied. Then in Chapter 3, one of the earlier motivations behind this work is further explained in detail, where the dynamic locomotion stack built to be run on a modern platform is presented. Based on my experience, often times, implementation details and unrevealed know-how's are required for an algorithm to successfully perform. In

Chapter 4, the necessary details for future researchers to start from as well as the results based on our experiments are shared. Lastly, because I believe this is just the beginning for the field of modern humanoid robots, some future works directly pertinent to ARTEMIS as well as some that are less so are discussed in Chapter 5 in hopes of continued progress with this platform, humanoids in general, as well as collaboration with other fields such biomechanics, sociology, or even media and art!

To be honest, I find it a bit sad naming this section as a "Conclusion," because I do strongly believe this is just the beginning and a word like "Commencement" might be more appropriate! ARTEMIS, at the time of this dissertation, is frequently taking walks around the UCLA campus without a support gantry as seen in Figure 6.1, is becoming a surprise guest in many schoolwide events, and has definitely now stepped out of its comfort zone inside the lab. These moments, along with other humanoids' performances in general, make me believe even more that this is truly a beginning of an exciting new chapter for humanoids!

To those that have allowed me to start this work from where it could start from, as well as those that will now start from where I personally conclude, thank you for helping push this technology forward! I took my "step" with this work in this field and to now let you have a go at pushing this forward, I will conclude my dissertation. Thank you. :)

ARTEMIS Official Launch Video: `https://youtu.be/gTkupawAG6w`

Figure 6.1: ARTEMIS taking a stroll just outside of Royce Hall.

APPENDIX A

Ignored Collision Combinations

The following collision combinations are ignored because either they never happen or because they are adjacent links. By removing the majority of the collision combinations, we are able to check for self-collisions at 1,000 Hz.

Link 1	**Link 2**	**Reason**
Body	Left/Right Hip Roll	Adjacent
Body	Left/Right Hip Pitch	Never
Body	Left/Right Femur	Never
Body	Left/Right Tibia	Never
Body	Left/Right Foot	Never

Link 1	**Link 2**	**Reason**
Left Hip Roll	Left Hip Pitch	Adjacent
Right Hip Roll	Right Hip Pitch	Adjacent
Left/Right Hip Roll	Left/Right Femur	Never
Left/Right Hip Roll	Left/Right Foot	Never

Link 1	Link 2	Reason
Left Hip Pitch	Left Femur	Adjacent
Right Hip Pitch	Right Femur	Adjacent
Left Hip Pitch	Right Femur	Never
Right Hip Pitch	Left Femur	Never
Left Hip Pitch	Left/Right Foot	Never
Right Hip Pitch	Left/Right Foot	Never

Link 1	Link 2	Reason
Left Femur	Left Tibia	Adjacent
Right Femur	Right Tibia	Adjacent
Left Femur	Right Femur	Never
Right Femur	Left Femur	Never
Left Femur	Left/Right Foot	Never
Right Femur	Left/Right Foot	Never

Link 1	Link 2	Reason
Left Tibia	Left Foot	Adjacent
Right Tibia	Right Foot	Adjacent

APPENDIX B

Inertial Parameters

The inertial parameters follow the Universal Robot Description Format (URDF) convention. The right leg and the right arm are not shown for brevity.

Link	**Mass [kg]**	**CoM [m]**	**ixx**	**ixy**	**ixz**	**iyy**	**iyz**	**izz**
Body	11.8028	$x = -0.0679$ $y = 0.0018$ $z = 0.1450$	0.3679	0.0	-0.0147	0.3861	0.0	0.2171

Link	**Mass [kg]**	**CoM [m]**	**ixx**	**ixy**	**ixz**	**iyy**	**iyz**	**izz**
Neck Bottom	0.3	$x = 0.0$ $y = 0.0014$ $z = -0.0012$	0.0001	0.0	0.0	0.0002	0.0	0.0001
Neck Top	0.1818	$x = -0.0010$ $y = -0.0557$ $z = 0.0$	0.0004	0.0	0.0	0.0004	0.0	0.0005

Link	**Mass [kg]**	**CoM [m]**	**ixx**	**ixy**	**ixz**	**iyy**	**iyz**	**izz**
Left Hip	1.4896	$x = 0.0153$ $y = 0.0$ $z = -0.0467$	0.0025	0.0	-0.0002	0.0021	0.0	0.0014
Left Adductor	2.3689	$x = -0.0039$ $y = -0.0029$ $z = -0.1278$	0.0108	0.0	-0.0005	0.0178	0.0	0.0080
Left Femur	6.0440	$x = 0.0598$ $y = -0.0029$ $z = 0.0260$	0.0183	-0.0019	-0.0017	0.0905	-0.0003	0.1011
Left Tibia	1.1257	$x = 0.1200$ $y = -0.0211$ $z = -0.0011$	0.0007	0.0	-0.0001	0.0122	0.0	0.0125
Left Foot	0.1342	$x = 0.0269$ $y = -0.0244$ $z = 0.0002$	0.0002	0.0	0.0	0.0	0.0	0.0002

Link	**Mass [kg]**	**CoM [m]**	**ixx**	**ixy**	**ixz**	**iyy**	**iyz**	**izz**
Left Clavicle	0.3323	$x = 0.0065$ $y = -0.0010$ $z = -0.0014$	0.0002	0.0	0.0	0.0001	0.0	0.0001
Left Scapula	0.3481	$x = 0.0596$ $y = 0.0002$ $z = 0.0002$	0.0002	0.0	0.0	0.0002	0.0	0.0002
Left Upperarm	0.3967	$x = -0.0054$ $y = 0.0154$ $z = 0.2521$	0.0014	0.0	0.0	0.0013	0.0001	0.0002
Left Forearm	0.1325	$x = 0.1245$ $y = 0.0143$ $z = -0.0007$	0.0	0.0	0.0	0.0016	0.0	0.0016

APPENDIX C

Joint Position and Torque Limits

The hard limits are the limits based on the hardware and the soft limits are those set in the software. Note that only the limits for the left leg and arm are shown as it is symmetric on the right.

Joint Name	**Hard Position Limits [rad]**	**Soft Position Limits [rad]**
Hip Yaw	[-0.5, 2.0]	[-0.4, 0.9]
Hip Roll	[-1.0, 2.0]	[-0.6, 0.6]
Hip Pitch	[-2.6, 1.0]	[-2.0, 0.7]
Knee Pitch	[0.375, 2.82]	[0.28, 2.3]
Ankle Pitch	[-1.25, 0.64]	[-1.3, 0.7]
Shoulder Pitch	-	[-3.2, 3.2]
Shoulder Roll	-	[-1.6, 1.4]
Shoulder Yaw	-	[-3.2, 1.6]
Elbow Pitch	-	[-3.0, 0.1]
Neck Pan	-	[-1.6, 1.6]
Neck Tilt	-	[-0.7, 1.7]

Table C.1: Joint position limits.

Joint Name	**Peak Torque [Nm]**	**Soft Torque Limit [Nm]**
Hip Yaw	90.0	59.004
Hip Roll	90.0	59.004
Hip Pitch	250.0	167.0
Knee Pitch	250.0	167.0
Ankle Pitch	28.0	16.929
Shoulder Pitch	30.0	30.0
Shoulder Roll	30.0	30.0
Shoulder Yaw	30.0	30.0
Elbow Pitch	30.0	30.0
Neck Pan	11.0	11.0
Neck Tilt	11.0	11.0

Table C.2: Joint torque limits.

REFERENCES

[ACH18] Min Sung Ahn, Hosik Chae, and Dennis W Hong. "Stable, Autonomous, Unknown Terrain Locomotion for Quadrupeds Based on Visual Feedback and Mixed-Integer Convex Optimization." In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3791–3798. IEEE, 2018.

[ACN19] Min Sung Ahn, Hosik Chae, Donghun Noh, Hyunwoo Nam, and Dennis Hong. "Analysis and noise modeling of the intel realsense d435 for mobile robots." In *2019 16th International Conference on Ubiquitous Robots (UR)*, pp. 707–711. IEEE, 2019.

[ACR20] Fernando Alves, Sara Cruz, Anabela Ribeiro, Ana Bastos Silva, João Martins, and Inês Cunha. "Walkability index for elderly health: a proposal." *Sustainability*, **12**(18):7360, 2020.

[ACT22] Min Sung Ahn, Hosik Chae, Colin Togashi, Dennis Hong, Joohyung Kim, and Sungjoon Choi. "Learning-based Motion Stabilizer Leveraging Offline Temporal Optimization." In *2022 19th International Conference on Ubiquitous Robots (UR)*, pp. 129–136. IEEE, 2022.

[AH20] Min Sung Ahn and Dennis Hong. "Dynamic, robust locomotion for a non-anthropomorphic biped." In *2020 17th International Conference on Ubiquitous Robots (UR)*, pp. 185–191. IEEE, 2020.

[Ahna] Min Sung Ahn. "CBEAR." `http://www.github.com/aminsung/CBEAR2`. Accessed: 2012-10-30.

[Ahnb] Min Sung Ahn. "CV7 Wrapper." `http://www.github.com/aminsung/cv7_wrapper`. Accessed: 2012-10-30.

[Ale76] RMcn Alexander. "Mechanics of bipedal locomotion." *Perspectives in experimental biology*, **1**:493–504, 1976.

[BCR12] Pranav A Bhounsule, Jason Cortell, and Andy Ruina. "Design and control of Ranger: an energy-efficient, dynamic walking robot." In *Adaptive Mobile Robotics*, pp. 441–448. World Scientific, 2012.

[BET22] Antoine Bambade, Sarah El-Kazdadi, Adrien Taylor, and Justin Carpentier. "Prox-qp: Yet another quadratic programming solver for robotics and beyond." In *RSS 2022-Robotics: Science and Systems*, 2022.

[BHH13] 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.

[Bli89] Reinhard Blickhan. "The spring-mass model for running and hopping." *Journal of biomechanics*, **22**(11-12):1217–1227, 1989.

[BMS09] Silvere Bonnable, Philippe Martin, and Erwan Salaün. "Invariant extended Kalman filter: theory and application to a velocity-aided attitude estimation problem." In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 1297–1304. IEEE, 2009.

[BPK18] Gerardo Bledt, Matthew Powell, Benjamin Katz, Jared Di Carlo, Patrick Wensing, and Sangbae Kim. "MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot." In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.

[CAN21] Hosik Chae, Min Sung Ahn, Donghun Noh, Hyunwoo Nam, and Dennis Hong. "BALLU2: A Safe and Affordable Buoyancy Assisted Biped." *Frontiers in Robotics and AI*, p. 290, 2021.

[CB19] Erwin Coumans and Yunfei Bai. "PyBullet, a Python module for physics simulation for games, robotics and machine learning." `http://pybullet.org`, 2016–2019.

[CCC08] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, Guido Ranzuglia, et al. "Meshlab: an open-source mesh processing tool." In *Eurographics Italian chapter conference*, volume 2008, pp. 129–136. Salerno, Italy, 2008.

[CED08] Jose A Cobano, Joaquin Estremera, and P Gonzalez De Santos. "Location of legged robots in outdoor environments." *Robotics and Autonomous Systems*, **56**(9):751–761, 2008.

[CEL19] Stéphane Caron, Adrien Escande, Leonardo Lanari, and Bastien Mallein. "Capturability-based pattern generation for walking with variable height." *IEEE Transactions on Robotics*, **36**(2):517–536, 2019.

[CGS18] R Conti, F Giovacchini, L Saccares, N Vitiello, Jose L Pons, and D Torricelli. "What do people expect from benchmarking of bipedal robots? Preliminary results of the EUROBENCH survey." In *Wearable Robotics: Challenges and Trends: Proceedings of the 4th International Symposium on Wearable Robotics, WeRob2018, October 16-20, 2018, Pisa, Italy*, pp. 132–136. Springer, 2018.

[Cra06] John J Craig. *Introduction to robotics*. Pearson Educacion, 2006.

[CSB19] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. "The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives." In *IEEE International Symposium on System Integrations (SII)*, 2019.

[DFF00] Michael H Dickinson, Claire T Farley, Robert J Full, MAR Koehl, Rodger Kram, and Steven Lehman. "How animals move: an integrative view." *science*, **288**(5463):100–106, 2000.

[DPL21] 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.

[DVT14] Hongkai Dai, Andrés Valenzuela, and Russ Tedrake. "Whole-body motion planning with centroidal dynamics and full kinematics." In *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 295–302. IEEE, 2014.

[DWK18] 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 (IROS)*, pp. 1–9. IEEE, 2018.

[EWO14] Johannes Englsberger, Alexander Werner, Christian Ott, Bernd Henze, Maximo A Roa, Gianluca Garofalo, Robert Burger, Alexander Beyer, Oliver Eiberger, Korbinian Schmid, et al. "Overview of the torque-controlled humanoid robot TORO." In *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 916–923. IEEE, 2014.

[Fea14] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.

[Fel17] Martin L Felis. "RBDL: an efficient rigid-body dynamics library using recursive algorithms." *Autonomous Robots*, **41**(2):495–511, 2017.

[FH85] Tamar Flash and Neville Hogan. "The coordination of arm movements: an experimentally confirmed mathematical model." *Journal of neuroscience*, **5**(7):1688–1703, 1985.

[FJS17] Farbod Farshidian, Edo Jelavic, Asutosh Satapathy, Markus Giftthaler, and Jonas Buchli. "Real-time motion planning of legged robots: A model predictive control approach." In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 577–584. IEEE, 2017.

[FK99] Robert J Full and Daniel E Koditschek. "Templates and anchors: neuromechanical hypotheses of legged locomotion on land." *Journal of experimental biology*, **202**(23):3325–3332, 1999.

[GFR19] Ruben Grandia, Farbod Farshidian, René Ranftl, and Marco Hutter. "Feedback mpc for torque-controlled legged robots." In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4730–4737. IEEE, 2019.

[GG21] Yukai Gong and Jessy Grizzle. "One-step ahead prediction of angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-inspired controller." In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2832–2838. IEEE, 2021.

[GG22] Yukai Gong and Jessy W Grizzle. "Zero dynamics, pendulum models, and angular momentum in feedback control of bipedal locomotion." *Journal of Dynamic Systems, Measurement, and Control*, **144**(12):121006, 2022.

[GGP19] Gabriel Garcia, Robert Griffin, and Jerry Pratt. "Convex optimization of the full centroidal dynamics for planning in multi-contact scenarios." *available online at ResearchGate Preprint*, 2019.

[GGP21a] 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 (Humanoids)*, pp. 276–282. IEEE, 2021.

[GGP21b] Gabriel García, Robert Griffin, and Jerry Pratt. "Time-Varying Model Predictive Control for Highly Dynamic Motions of Quadrupedal Robots." In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7344–7349. IEEE, 2021.

[GHM09] 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*, pp. 2030–2036. IEEE, 2009.

[GJK88] Elmer G Gilbert, Daniel W Johnson, and S Sathiya Keerthi. "A fast procedure for computing the distance between complex objects in three-dimensional space." *IEEE Journal on Robotics and Automation*, **4**(2):193–203, 1988.

[HAH20] Dennis W Hong, Min Sung Ahn, Joshua R Hooks, and Jeffrey C Yu. "Legged Robots, Design of." *Journal: Encyclopedia of Robotics*, pp. 1–11, 2020.

[Han06] Duane C. Hanselman. *Brushless Permanent-Magnet Motor Design*. Magna Physics Publishing, 2006.

[HAY20] Joshua Hooks, Min Sung Ahn, Jeffrey Yu, Xiaoguang Zhang, Taoyuanmin Zhu, Hosik Chae, and Dennis Hong. "Alphred: A multi-modal operations quadruped robot for package delivery applications." *IEEE Robotics and Automation Letters*, **5**(4):5409–5416, 2020.

[HGE20] Ross Hartley, Maani Ghaffari, Ryan M Eustice, and Jessy W Grizzle. "Contact-aided invariant extended Kalman filtering for robot state estimation." *The International Journal of Robotics Research*, **39**(4):402–430, 2020.

[HGJ16] 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.

[HHH98] Kazuo Hirai, Masato Hirose, Yuji Haikawa, and Toru Takenaka. "The development of Honda humanoid robot." In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pp. 1321–1326. IEEE, 1998.

[HLH18] Jemin Hwangbo, Joonho Lee, and Marco Hutter. "Per-contact iteration method for solving contact dynamics." *IEEE Robotics and Automation Letters*, **3**(2):895–902, 2018.

[HLL16] Michael A Hopkins, Alexander Leonessa, Brian Y Lattimer, and Dennis W Hong. "Optimization-based whole-body control of a series elastic humanoid robot." *International Journal of Humanoid Robotics*, **13**(01):1550034, 2016.

[HP08] Hugh Herr and Marko Popovic. "Angular momentum in human walking." *Journal of experimental biology*, **211**(4):467–481, 2008.

[Hre95] Alan Hreljac. "Determinants of the gait transition speed during human locomotion: kinematic factors." *Journal of biomechanics*, **28**(6):669–677, 1995.

[HTH10] K Hashimoto, Y Takezaki, K Hattori, H Kondo, T Takashima, HO Lim, and A Takanishi. "A study of function of the human's foot arch structure using biped humanoid robot." In *Proc. of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2206–2211, 2010.

[IH00] Hirochika Inoue and Hirohisa Hirukawa. "HRP: Humanoid robotics project of MITI." *Journal of the Robotics Society of Japan*, **18**(8):1089–1092, 2000.

[JMK22] Gwanghyeon Ji, Juhyeok Mun, Hyeongjun Kim, and Jemin Hwangbo. "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion." *IEEE Robotics and Automation Letters*, **7**(2):4630–4637, 2022.

[KDK16] Gavin D 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.

[KDK19a] Benjamin Katz, Jared Di Carlo, and Sangbae Kim. "Mini cheetah: A platform for pushing the limits of dynamic quadruped control." In *2019 international conference on robotics and automation (ICRA)*, pp. 6295–6301. IEEE, 2019.

[KDK19b] 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." *arXiv preprint arXiv:1909.06586*, 2019.

[KDR12] 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.

[KH04] 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 (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pp. 2149–2154. IEEE, 2004.

[KHM20] 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.

[KKK01] 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 *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, volume 1, pp. 239–246. IEEE, 2001.

[KKK03] 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 *ICRA*, volume 3, pp. 1620–1626, 2003.

[KKS19] Kenji Kaneko, Hiroshi Kaminaga, Takeshi Sakaguchi, Shuuji Kajita, Mitsuharu Morisawa, Iori Kumagai, and Fumio Kanehiro. "Humanoid robot HRP-5P: An electrically actuated humanoid robot with high-power and wide-range joints." *IEEE Robotics and Automation Letters*, **4**(2):1431–1438, 2019.

[KMS19] Iori Kumagai, Mitsuharu Morisawa, Takeshi Sakaguchi, Shin'ichiro Nakaoka, Kenji Kaneko, Hiroshi Kaminaga, Shuuji Kajita, Mehdi Benallegue, Rafael Cisneros, and Fumio Kanehiro. "Toward industrialization of humanoid robots: Autonomous plasterboard installation to improve safety and efficiency." *IEEE Robotics & Automation Magazine*, **26**(4):20–29, 2019.

[KOK74] 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*, pp. 11–24. Springer, 1974.

[KSJ08] Danny M Kaufman, Shinjiro Sueda, Doug L James, and Dinesh K Pai. "Staggered projections for frictional contact in multibody systems." In *ACM SIGGRAPH Asia 2008 papers*, pp. 1–11. ACM, 2008.

[KSM20] Johannes Köhler, Raffaele Soloperto, Matthias A Müller, and Frank Allgöwer. "A computationally efficient robust model predictive control framework for uncertain nonlinear systems." *IEEE Transactions on Automatic Control*, **66**(2):794–801, 2020.

[KSP04] Oussama Khatib, Luis Sentis, Jaeheung Park, and James Warren. "Whole-body dynamic behavior and control of human-like robots." *International Journal of Humanoid Robotics*, **1**(01):29–43, 2004.

[LBU09] Sebastian Lohmeier, Thomas Buschmann, Heinz Ulbrich, and Friedrich Pfeiffer. "Humanoid Robot LOLA-Research Platform for High-SpeedWalking." In *Motion and Vibration Control*, pp. 221–230. Springer, 2009.

[LDT16] Matthias Lorenzen, Fabrizio Dabbene, Roberto Tempo, and Frank Allgöwer. "Constraint-tightening and stability in stochastic model predictive control." *IEEE Transactions on Automatic Control*, **62**(7):3165–3177, 2016.

[Lee14] Bryce Kenji Tim-Sung Lee. *Design of a humanoid robot for disaster response*. PhD thesis, Virginia Tech, 2014.

[LLL17] Young Hun Lee, Yoon Haeng Lee, Hyunyong Lee, Luong Tin Phan, Hansol Kang, Uikyum Kim, Jeongmin Jeon, and Hyouk Ryeol Choi. "Trajectory design and control of quadruped robot for trotting over obstacles." In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4897–4902. IEEE, 2017.

[LM68] RA Liston and RS Mosher. "A versatile walking truck." In *Transportation Engineering Conference*, 1968.

[LP17] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[LWO15] Yiping Liu, Patrick M Wensing, David E Orin, and Yuan F Zheng. "Dynamic walking in a humanoid robot based on a 3D actuated Dual-SLIP model." In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5710–5717. IEEE, 2015.

[LWS16] Yiping Liu, Patrick M Wensing, James P Schmiedeler, and David E Orin. "Terrain-blind humanoid walking based on a 3-D actuated dual-SLIP model." *IEEE Robotics and Automation Letters*, **1**(2):1073–1080, 2016.

[LXH18] Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C Karen Liu. "Dart: Dynamic animation and robotics toolkit." *The Journal of Open Source Software*, **3**(22):500, 2018.

[MC90] Thomas A McMahon and George C Cheng. "The mechanics of running: how does stiffness couple with speed?" *Journal of biomechanics*, **23**:65–78, 1990.

[McG90] Tad McGeer et al. "Passive dynamic walking." *I. J. Robotic Res.*, **9**(2):62–82, 1990.

[mic] "3DM-CV7-AHRS Manual.".

[MK13] Carlotta Mummolo and Joo H Kim. "Passive and dynamic gait measures for biped mechanism: formulation and simulation analysis." *Robotica*, **31**(4):555–572, 2013.

[MLP16] Khaled Mamou, E Lengyel, and A Peters. "Volumetric hierarchical approximate convex decomposition." In *Game Engine Gems 3*, pp. 141–158. AK Peters, 2016.

[MYY17] Stephen G McGill, Seung-Joon Yi, Hak Yi, Min Sung Ahn, Sanghyun Cho, Kevin Liu, Daniel Sun, Bhoram Lee, Heejin Jeong, Jinwook Huh, et al. "Team THOR's entry in the DARPA Robotics Challenge Finals 2015." *Journal of Field Robotics*, **34**(4):775–801, 2017.

[PA16] Matthew J Powell and Aaron D Ames. "Mechanics-based control of underactuated 3D robotic walking: Dynamic gait generation under torque constraints." In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 555–560. IEEE, 2016.

[PCD06] Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. "Capture point: A step toward humanoid push recovery." In *2006 6th IEEE-RAS international conference on humanoid robots*, pp. 200–207. IEEE, 2006.

[PD07] Jerry E Pratt and Sergey V Drakunov. "Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model." In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 4653–4660. IEEE, 2007.

[PG09] Ioannis Poulakakis and Jessy W Grizzle. "The spring loaded inverted pendulum as the hybrid zero dynamics of an asymmetric hopper." *IEEE Transactions on Automatic Control*, **54**(8):1779–1793, 2009.

[PHH09] Sangbum Park, Youngjoon Han, and Hernsoo Hahn. "Balance control of a biped robot using camera image of reference object." *International Journal of Control, Automation, and Systems*, **7**(1):75, 2009.

[Pra02] Gill Andrews Pratt. "Low impedance walking robots." *Integrative and Comparative Biology*, **42**(1):174–181, 2002.

[PW95] Gill A Pratt and Matthew M Williamson. "Series elastic actuators." In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 1, pp. 399–406. IEEE, 1995.

[PWK17] Hae-Won Park, Patrick M Wensing, and Sangbae Kim. "High-speed bounding with the MIT Cheetah 2: Control design and experiments." *The International Journal of Robotics Research*, **36**(2):167–192, 2017.

[Rai86] Marc H Raibert. *Legged robots that balance*. MIT press, 1986.

[RBC84] Marc H Raibert, H Benjamin Brown Jr, and Michael Chepponis. "Experiments in balance with a 3D one-legged hopping machine." *The International Journal of Robotics Research*, **3**(2):75–92, 1984.

[RBF21] Niraj Rathod, Angelo Bratta, Michele Focchi, Mario Zanon, Octavio Villarreal, Claudio Semini, and Alberto Bemporad. "Model predictive control with environment adaptation for legged locomotion." *IEEE Access*, **9**:145710–145727, 2021.

[RBN08] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, and Rob Playter. "Bigdog, the rough-terrain quadruped robot." *IFAC Proceedings Volumes*, **41**(2):10822–10825, 2008.

[RBR14] Nicholas Rotella, Michael Bloesch, Ludovic Righetti, and Stefan Schaal. "State estimation for a humanoid robot." In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 952–958. IEEE, 2014.

[RHJ15] Siavash Rezazadeh, Christian Hubicki, Mikhail Jones, Andrew Peekema, Johnathan Van Why, Andy Abate, and Jonathan Hurst. "Spring-mass walking with atrias in 3d: Robust gait control spanning zero to 4.3 kph on a heavily underactuated bipedal robot." In *Dynamic Systems and Control Conference*, volume 57243, p. V001T04A003. American Society of Mechanical Engineers, 2015.

[Ryg93] Lewis A. Rygg. "Mechanical horse.", 1893. Patent No. US491927A, Filed Apr. 1892, Issued Feb. 14th, 1893.

[SBG20] 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.

[SGM15] Claudio Semini, Jake Goldsmith, Diego Manfredi, Flaviana Calignano, Elisa Paola Ambrosio, Jukka Pakkanen, and Darwin G Caldwell. "Additive manufacturing for agile legged robots with hydraulic actuation." In *2015 International Conference on Advanced Robotics (ICAR)*, pp. 123–129. IEEE, 2015.

[She22] Junjie Shen. *Locomotion Analysis and Control of a Miniature Bipedal Robot*. PhD thesis, UCLA, 2022.

[SHV06] Mark W Spong, Seth Hutchinson, Mathukumalli Vidyasagar, et al. *Robot modeling and control*, volume 3. Wiley New York, 2006.

[SK06] Luis Sentis and Oussama Khatib. "A whole-body control framework for humanoids operating in human environments." In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2641–2648. IEEE, 2006.

[Smi] R. Smits. "KDL: Kinematics and Dynamics Library." `http://www.orocos.org/kdl`.

[SPP11] Koushil Sreenath, Hae-Won Park, Ioannis Poulakakis, and Jessy W Grizzle. "A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on MABEL." *The International Journal of Robotics Research*, **30**(9):1170–1193, 2011.

[SWA02] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. "The intelligent ASIMO: System overview and integration." In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pp. 2478–2483. IEEE, 2002.

[SWC14] Sangok Seok, Albert Wang, Meng Yee Chuah, Dong Jin Hyun, Jongwoo Lee, David M Otten, Jeffrey H Lang, and Sangbae Kim. "Design principles for energy-efficient legged locomotion and implementation on the MIT cheetah robot." *Ieee/asme transactions on mechatronics*, **20**(3):1117–1129, 2014.

[SWO12] S. Seok, A. Wang, D. Otten, and S. Kim. "Actuator design for high force proprioceptive control in fast legged locomotion." In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1970–1975, Oct 2012.

[SWS21] Philipp Seiwald, Shun-Cheng Wu, Felix Sygulla, Tobias FC Berninger, Nora-Sophie Staufenberg, Moritz F Sattler, Nicolas Neuburger, Daniel Rixen, and Federico Tombari. "LOLA v1. 1–An Upgrade in Hardware and Software Design for Dynamic Multi-Contact Locomotion." In *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pp. 9–16. IEEE, 2021.

[TCN17] Nikos G Tsagarakis, Darwin G Caldwell, F Negrello, W Choi, L Baccelliere, VG Loc, J Noorden, L Muratore, A Margan, A Cardellino, et al. "WALK-MAN: A High-Performance Humanoid Platform for Realistic Environments." *Journal of Field Robotics*, **34**(7):1225–1259, 2017.

[TET12] Emanuel Todorov, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control." In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

[Tod14] Emanuel Todorov. "Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in mujoco." In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6054–6061. IEEE, 2014.

[TP19] D Torricelli and Jose L Pons. "Eurobench: Preparing robots for the real world." In *Wearable Robotics: Challenges and Trends: Proceedings of the 4th International Symposium on Wearable Robotics, WeRob2018, October 16-20, 2018, Pisa, Italy 3*, pp. 375–378. Springer, 2019.

[Tuc75] Vance A Tucker. "The energetic cost of moving about: walking and running are extremely inefficient forms of locomotion. Much greater efficiency is achieved by birds, fish—and bicyclists." *American Scientist*, **63**(4):413–419, 1975.

[Ush05] James Richard Usherwood. "Why not walk faster?" *Biology letters*, **1**(3):338–341, 2005.

[VS72] Miomir Vukobratović and J Stepanenko. "On the stability of anthropomorphic systems." *Mathematical biosciences*, **15**(1-2):1–37, 1972.

[Wes03] Eric Richard Westervelt. *Toward a coherent framework for the control of planar biped locomotion*. University of Michigan, 2003.

[WLC13] Liyang Wang, Zhi Liu, CL Philip Chen, Yun Zhang, Sukhan Lee, and Xin Chen. "A UKF-based predictable SVR learning controller for biped walking." *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **43**(6):1440–1450, 2013.

[WWS17] 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.

[XA20] Xiaobin Xiong and Aaron D Ames. "Sequential motion planning for bipedal somersault via flywheel slip and momentum transmission with task space control." In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3510–3517. IEEE, 2020.

[YGD20] Haitao Yu, Haibo Gao, and Zongquan Deng. "Toward a unified approximate analytical representation for spatially running spring-loaded inverted pendulum model." *IEEE Transactions on Robotics*, **37**(2):691–698, 2020.

[YHG16] Jeffrey Yu, Joshua Hooks, Sepehr Ghassemi, Alexandra Pogue, and Dennis Hong. "Investigation of a non-anthropomorphic bipedal robot with stability, agility, and simplicity." In *Ubiquitous Robots and Ambient Intelligence (URAI), 2016 13th International Conference on*, pp. 11–15. IEEE, 2016.

[YHZ18] 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 (Humanoids)*, pp. 1–9. IEEE, 2018.

[Yu20] 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.

[ZAH21] 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 (ICAR)*, pp. 1137–1143. IEEE, 2021.

[Zha19] Xiaoguang Zhang. *Application of Proprioception Quasi-Direct Drive Actuators on Dynamic Robotic Systems*. University of California, Los Angeles, 2019.

[ZHH19] 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 (AIM)*, pp. 36–43. IEEE, 2019.