UNIVERSITY OF CALIFORNIA

Los Angeles

Control of a Dynamic Torque Controlled Humanoid For Soccer

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

by

Colin Travis Togashi

2024

ABSTRACT OF THE DISSERTATION

Control of a Dynamic Torque Controlled Humanoid For Soccer

by

Colin Travis Togashi
Doctor of Philosophy in Mechanical Engineering
University of California, Los Angeles, 2024
Professor Dennis W. Hong, Chair

Robots have become more ubiquitous in daily life than ever before. Package delivery, factory automation, and transportation are just a few of the areas that have started to see robots take defining roles. Despite this, robotics still has a long way to before realizing the end goal of living and working amongst humans. The real world presents many challenges including dynamic obstacles, uneven contact surfaces, and changing operating conditions that require robots to quickly adapt on the fly in order to be successful. While current state of the art robots have become quite useful as single task platforms, there is still a need for a more generalizeable solution in order to meet the challenging needs of real world tasks. In order to truly adapt to an environment made for a human, robots must now take on the same humanoid form.

To this end, a bipedal locomotion control stack was developed for the torque controlled humanoid Artemis. A model based hierarchical stack using a linear inverted pendulum inspired footstep planner, polynomial swing leg trajectories, and an implicitly weighted hierarchical whole body controller were tested to yield stable walking at a maximum walking speed of 1.2 m/s. In addition, to verify the effectiveness of the platform and its underlying software under real world conditions, an autonmous soccer playing software stack was implemented to compete in Robocup 2024. The stack includes a vision model capable of detecting balls and field landmarks, a localization algorithm to estimate the game state, a behavior tree to make real time soccer decisions, and a path planner to avoid obstacles through gameplay. Furtheremore, a soccer kick was developed that was capable of kicking the soccer over half of the miniturized field in a single kick. These advancements resulted in a first place finish at Robocup 2024.

The dissertation of Colin Travis Togashi is approved.

University of California, Los Angeles

2024

To my family and friends

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would first like to thank my advisor, Professor Dennis Hong. Through his guidance, I have been given a number of opportunities at RoMeLa that most students would only dream of and that have helped shaped me into the person I am today. He's always allowed me to push the limits while providing a safe environment for growth and I am forever grateful for that.

I would also like to thank my committee members: Professor Jacob Rosen, Professor Veronica Santos, and Professor Khalid Jawed. They have the served as excellent educators in my studies and guided me througout my PhD. I am particularly grateful to have served as their TA as it has sparked a joy for teaching I previously didn't know existed.

To all my lab members both past and present, thank you for all of the great memories we've had together. You all truly made my stay at RoMeLa fly by with all the fun we had. It was truly a pleasure to work with you every day.

Thank you to all my friends outside of lab and in particular, Jonathan Hao, Cynthia Tran, Joey Chen, Eunice Park, Derek Hsieh, and Marlene Hong. Despite the fact that I would disappear every year for a couple months, buried in my work, you all still kept in touch and let me think about things not related to robotics. I would not have been able to survive this journey without you.

Finally, I am truly grateful for my mom, Kyung, my dad, Calvin, and my brother and sister, Ryan and Allyson. Without you all, this journey would have been meaningless. Even though my decision to do a PhD was questioned, my family backed me with full support the moment I embarked on my journey. I love you guys!

VITA

2010–2014	B.S. in Mechanical Engineering, Univeristy of California, San Diego
2012–2014	Engeering Aide, Scripps Institute of Oceanography Machine Science Development Center
2013–2014	Undergraduate Researcher, UCSD Flow Control and Coordinated Robotics Lab
2014–2018	Controls Engineer, Cymer Inc.
2016–2018	M.S. in Mechanical Engineering, Univeristy of California, Los Angeles
2016–2021	Teaching Assistant, Mechincal Engineering Department at Univeristy of California, Los Angeles
2020–2024	Graduate Student Researcher, Robotics and Mechanisms Laboratory (RoMeLa) at Univeristy of California, Los Angeles
2018–2024	Doctor of Philosophy Candidate, Univeristy of California, Los Angeles

PUBLICATIONS

G. I. Fernandez, C. Togashi, D. W. Hong, and L. F. Yang, “Deep reinforcement learning with linear quadratic regulator regions,” arXiv preprint arXiv:2002.09820, 2020.

T. Zhu, G. I. Fernandez, C. Togashi, Y. Liu, and D. Hong, “Feasibility study of limms, a multi-agent modular robotic delivery system with various locomotion and manipulation modes,” in 2022 19th International Conference on Ubiquitous Robots (UR). IEEE, 2022, pp. 30–37.

M. S. Ahn, H. Chae, C. Togashi, D. Hong, J. Kim, and S. Choi, “Learning-based motion stabilizer leveraging offline temporal optimization,” in 2022 19th International Conference on Ubiquitous Robots (UR). IEEE, 2022, pp. 129–136.

G. I. Fernandez, R. Hou, C. Togashi, A. Xu and D. W. Hong, “CLAP: Clustering to Localize Across n Possibilities, A Simple, Robust Geometric Approach in the Presence of Symmetries“, Submitted to 2025 IEEE International Conference on Robotics and Automation (ICRA)

G. I. Fernandez, Y. Liu, C. Togashi, K. Gillespie, A. Zhu, Q. Wang, Y. Wang, H. Nam, S. Wang, R. Hou, M. Zhu, A. Navghare, A. Xu, T. Zhu, M. S. Ahn, A. F. Alvarez, J. Quan, E. Hong, and D. W. Hong, “RoboCup 2024 Adult-Sized Humanoid Champions Guide for Hardware, Vision, and Strategy“ Submitted to 2024 RoboCup Symposium

CHAPTER 1

Introduction

1.1 Motivation

Over the past few years, large advances in robotics have pushed platforms out of laboratories and into the the real world. From vacuum cleaners to autonmous cars, package delivery and even healthcare, robots have started to interact with humans more than ever and help them in their daily lives. Despite these developments, however, most robots still tend to be relatively slow and targeted at solving single tasks. Multiple robotic systems are often required to coordinate in sequence to solve a problem that a single human could manage in a fraction of the time. Even where robots have excelled, a human is still often required to bridge the gap between single task systems to complete any given objective. To this end, there is a growing need to find a more generalized and multipurpose platform.

A simple approach for the design of such a generalized robotic system is to copy the form of the entity already performing the task: humanoids. The humanoid design offers several advantages. For one, it has a much smaller spatial footprint compared to wheeled or multi-legged robots. While humanoids may sacrifice some payload capacity in comparison to wheeled platforms, they are capable of navigating human environments, such as stairs, drops, and uneven surfaces, much more effectively. Their arms provide high dexterity, enabling them to manipulate a wide range of objects. Additionally, most dual-arm robotic research follows the humanoid form, which allows for leveraging a diverse and mature set of research. By adopting a human-like design, the robot can even operate within the same environments without requiring major modifications. This reduces the amount of auxilary work when applying the platform to a new location. Finally, humanoid robots allow designers to easily define tasks based on their own personal experiences and intuitions. This may be the most crucial as roboticists can more effectively create trajectories from what they know.

The humanoid form does have a few disadvantages as well. Relative to wheeled platforms, humanoids can only carry relatively smaller payloads and have a lower cost of transport [1]. Humanoids are also much more complex in terms of both hardware and software components. Whereas most wheeled platforms are inherently stable across flat ground, humanoids require complex control even to take a single step safely. The cost of manufactuing the platforms also differs quite greatly which limits the number of applications that humanoids are economically feasible to serve in. Despite these drawbacks, for a truly generalized platform that can go where humans go, it should be of similar form and function.

1.2 Background

Humanoid development has rapidly advanced throughout the last 50 years. Starting with WABOT-1 developed by Waseda University was the world's first humanoid [2]. This marked the first conceptual demonstration of a full sized humanoid that could interact with its surroundings and even communicate with humans. A few years later and the Honda P-Series of robots including the famous Asimo robot really demonstrated fast, stable walking at the full size scale [3]. This really showed that humanoid bipedal locomotion was a feasible solution. The HRP series of robots including HRP-4 demnostrated even further advancement with walking up stairs and single leg balancing [4]. One of the largest platform advances was Boston Dynamics Atlas which at the time was one of the most advanced and capable humanoids at the DARPA Robotics Challenge [5] [6]. It was by far the most commonly used platform on which teams built their stacks and competed. Atlas was improved and was seen as the most dynamically capable robots up until the 2020s. Nowadays, a number

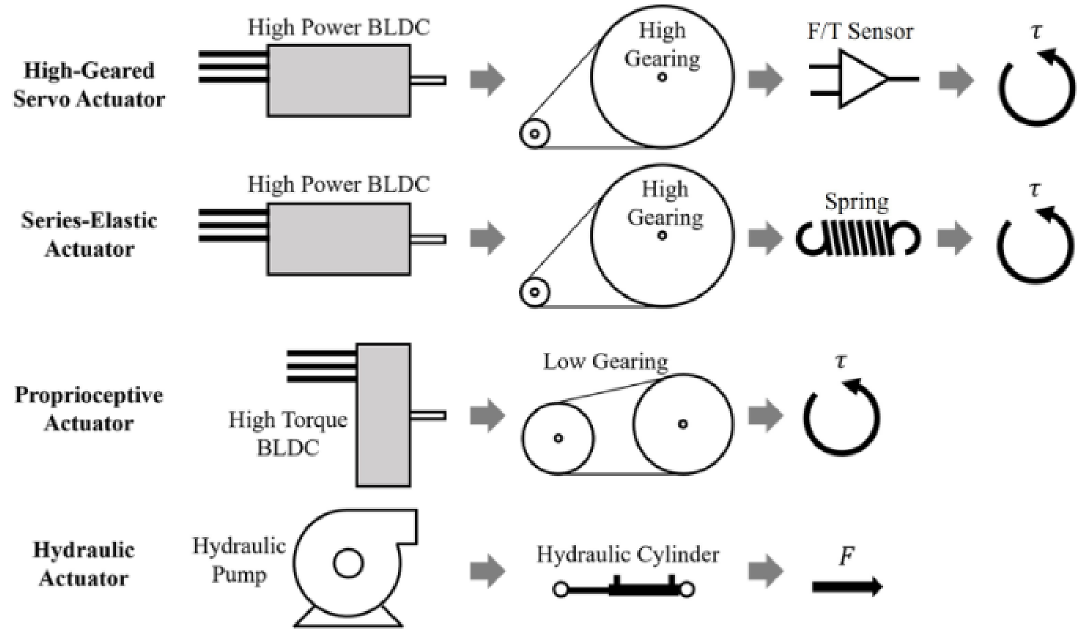


Figure 1.1: Traditionally, most robots have been controlled using high gear ratio servos. Only until recently has there been a shift to compliant actuation. Compliance can be implemented in a number of different ways including series elatics, proprioceptive, or hydrualic

of commerically available platforms including the Unitree H1, Figure AI Figure 02, Tesla Optimus, and the Apptronik Apollo [7] [8] [9] [10]. All of these platforms have demonstrated not only stable walking, but manipulation to some extent.

In order to make robots safer around humans and handle unexpected interactions with the environment, a general trend in robotics has been the notion of adding more compliance to robotic systems. Since using rigid body makes it easier for dynamics calculations, most have resorted to compliance at the actuation level although some still use mechanical springs [11] [12] [13]. Actuator compliance can be done in a number of ways as shown in Figure 1.1. Some robots have developed a physical spring between the output shaft of the motor and the link to have a spring in series or a Series Elasttic Actuator (SEA) [14]. Others have favored the use of hydraulics wherein fluids are used to generate motion and the compressibility of the fluid acts as a cushion against impacts [15]. By far, the most popular method nowadays is by using lower gear ratio motors named Quasi Direct Drive (QDD) or proprioceptive motors [16] [17] [18] [19] [20] [21] [22]. By using a current measurement as a proxy for torque, a lower level torque control algorithm at the joint level can, on the fly, change dynamic properties like stiffness and damping in software. This opens the door to a number of possibilities as a much wider range of operating modes can be programmed into the actuator. They are also much cheaper to manufacture and maintain relative while still remaining as generalizable as possible by allowing faster development without being constrained to a single set of mechanical springs.

Meanwhile, control algorithms for humanoids have split mainly into two separate approaches: model-based or those that use explicitly defined mathematical descriptions of the physics and dynamics to control motion and model-free, more frequently associated with AI, that chooses to allow an algorithm to learn the underlying system and determine the necessary control purely through data. Often times, there is cross pollination between the two theories to form some combination, however, this isn't quite commonplace yet. The current state of the art model-free approaches leverage neural networks through transformer architectures to develop control at each state. The challenge for most model-free systems historically has been the simulation to real world gap wherein a controller trained purely on simulation data yielded drastically poorer performance in the real world [23]. Recently, however, there have been large gains both to algorithms and to the ability to parallelize learning sessions in large worlds that have managed to bridge the gap [24]. Due to the lower cost and higher availability of actuators and platforms, it is now also feasible to run more training in the real world [25] [26]. Model-free approaches, however, tend to be harder to debug as the controller is encoded into a non-human interpretable set of weights trained off data.

Despite this, model based approaches have still proven to be quite capable, especially for those without the required tools to gather all the data. Since the approach is based on mathematical models, locomotion can be inutitively described and inform hardware decisions

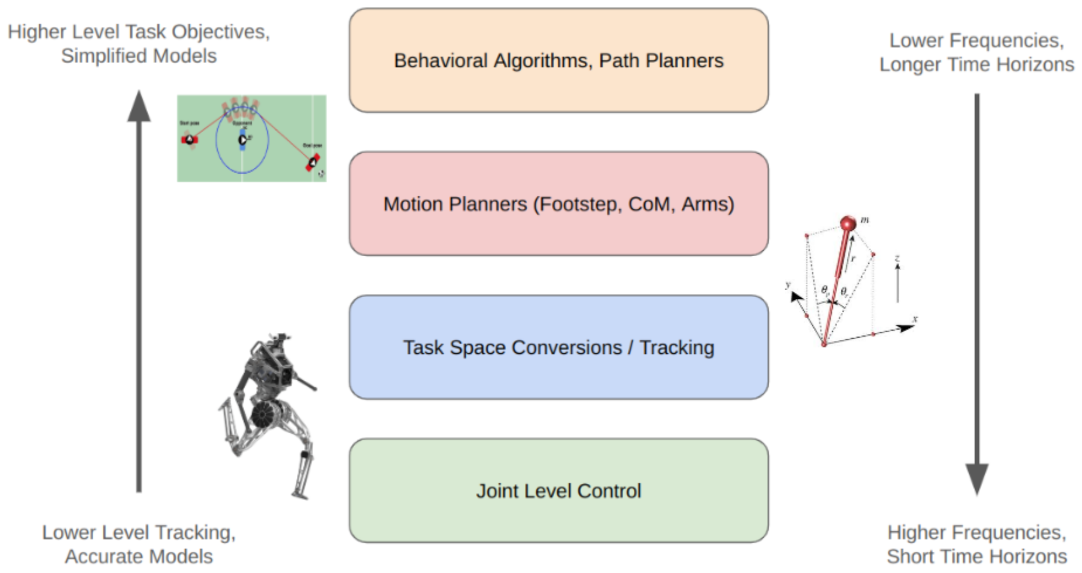

Higher Level Task Objectives, Simplified Models
Behavioral Algorithms, Path Planners
Lower Frequencies, Longer Time Horizons
Motion Planners (Footstep, CoM, Arms)
Task Space Conversions / Tracking
Joint Level Control
Lower Level Tracking, Accurate Models
Higher Frequencies, Short Time Horizons

Figure 1.2: A higher level controller will require a simplified model and longer time horizon while lower level controllers need to be more accurate

[27]. Succesful implementations include the following across multiple robots [28] [29] [30] [31] [32]. Since any model based controller requires explicit definition of models, this approach often requires large amounts of hand tuning. Often times, this results in the need for highly trained experts in order for the controller to work. Nonetheless, this work will focus on model based control.

The current state of the art seeks to encapsulate a hierarchical approach wherein lower level models are used at the bottom and higher level models are used at the top as shown in Figure 1.2 [33]. The lower level controllers tend to run at faster frequencies, but shorter time horizons while higher level algorithms run at slower frequencies and longer time horizons. This is due to the fact that the lower level is designed to react and track commands given higher up. As long as the higher level provides commands, the lower system will track them as best as possible. Even more so, for a successful hierarchical stack implementation, the lower level is responsible for matching as close as possible the simplified model that is being used at the next level above. The reason being is that for most high level control algorithms, the models have been simplified to the point where the control is an exact solution. That is given the model, the solution is not simply optimal, but rather the only solution to the problem. As a result, the better the lower level can maintain those assumptions and thus match the model, the better the control will be tracked, the better performance that controller will have. Problems arise when the lower level cannot compensate for the model and what is being controlled is drastically different from expectations.

In order to push the capabilities of humanoids to move faster, make correct decisions in a changing environment, and push the boundaries of how capable a platform can be in the real world, ARTEMIS was entered into Robocup, an autonomous soccer competition [34]. While soccer may not be an end goal in itself, it still demonstrates many capabilities that are required other more useful tasks. Walking on grassy, not flat, squishy surfaces improves locomotion standards, especially when subjected to multiple adversarial disturbances in the form of the soccer ball and other robot feet. Other robots and humans present dynamic and even adversarial obstacles that the robot must contend with. As a reult, performing well at soccer will lead to a better platform overall.

1.3 Contributions

The following work aims to contribute novel research in the following ways:

- Modularization of a humanoid locomotion stack
- Development of a dynamic soccer kick
- Application of higher level soccer behavior stack to win Robocup 2024 in Eindhoven, Netherlands

1.4 Organization

The remainder of the manuscript serves to outline improvements made to a locomotion stack on the torque controlled humanoid ARTEMIS. Chapter 2 discusses the main theory behind locomotion including development of the gait, swing leg trajectories, and whole body control. Chapter 3 discusses implementation details as well as the ARTEMIS platform. Chapter 4 provides an overview of the stack used to compete in Robocup 2024 including the behavior tree, localization, path planning, and vision model. Lastly, Chapter 5 offers concluding thoughts and future works that can be foreseen.

CHAPTER 2

Theory

2.1 Overview

To effectively control a humanoid robot, a hierarchical, model-based control stack was developed. Figure 2.1 shows a functional overview of all the components involved. Starting from the base, an esimator reports the state of the robot including its world frame position, the joint states, and contact state for the other controllers to act on. From these states, the control stack starts with a gait that syncs all temporal timings and ensures that the feet and body remain coordinated with each other. A footstep planner, working off of a Linear Inverted Pendulum (LIP) model, then plans for the next footstep position to stabilize the robot. Next up, swing leg planners ensure that the task space specifications for the feet can reliably reach desired touchdown location. A center of mass (CoM) planner then develops CoM task space trajectories to track while a whole body controller ensures that higher level objectives are tracked by joint level torque commands. Finally, commands at the motor level are tracked using joint level PD control and feedforward torque control.

As was discussed in Section 1.2, the stack heavily relies on the hierarchical nature of control wherein the higher level controllers use simplified models to make task level decisions such as the timings of the contacts and the placement of the feet. The main high level model that will be used and reinforced throughout the development of the stack is the Linear Inverted Pendulum (LIP) model. It is crucial that lower level controllers attempt to maintain all the assumptions set out by this high level model in order for the developed control strategies to work effectively. Meanwhile, to ensure that joint level commands can be accurately tracked based on the current state of the robot, a rigid body dynamics model is used at the lower level whole body controller. This ensures that simple yet effective trajectories can be developed over longer time horizons while still being correctly tracked by the robot.

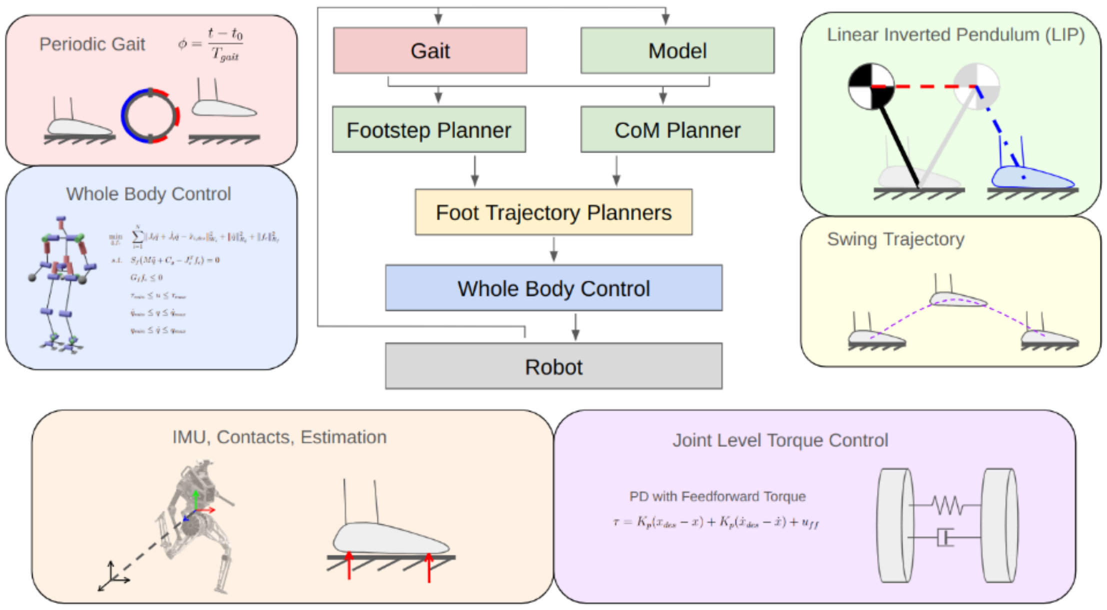

Periodic Gait
$\phi = \frac{t - t_0}{T_{gait}}$
Gait
Model
Linear Inverted Pendulum (LIP)
Footstep Planner
CoM Planner
Whole Body Control
Foot Trajectory Planners
Whole Body Control
Swing Trajectory
Robot
IMU, Contacts, Estimation
Joint Level Torque Control
PD with Feedforward Torque
$\tau = K_p(x_{des} - x) + K_p(\dot{x}_{des} - \dot{x}) + u_{ff}$

Figure 2.1: Conceptual overview of the proposed humanoid stack. Starting from the robot hardware on the left, joint and sensor information is acquired from actuators and the IMU. It is then processed through an estimator and filtered to generate a generalized state measurement. Meanwhile, planners for the arms, footsteps, and CoM compute trajectories with inputs from higher level commands, either from users or other components. At the lowest level, a whole body control tries to track all the task objectives and provides the final control outputs q_{des}, $\dot{q}_{des}$, and u_{des}. These values can be used for joint level torque control to control the robot.

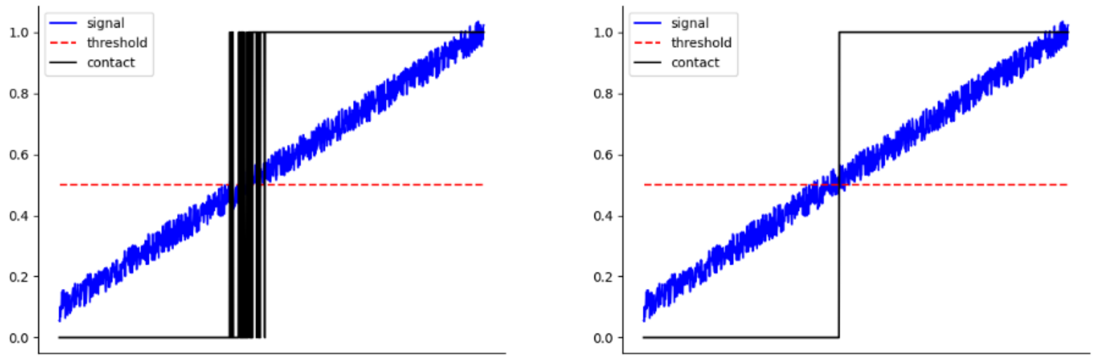


Figure 2.2: Motivation for contact debouncing. Without a proper contact debouncer on the left, the contact signal fluctuates rapidly on contact transitions. In order to fix this, a debouncer was introduced to hold the signal until enough consecutive samples changed the state as shown on the right

2.2 Estimation

The estimator used in this stack is a modified version of the Contact-aided Invariant Extended Kalman Filter [35]. Using an on board IMU to propagate a floating base model, it uses contact locations as measurement updates to correct the global frame position, orientation, and velocity. The main output of the estimator is the current generalized state, q_g, and generalized velocity, $\dot{q}_g$, of the robot. In this work, the overall implementation did not differ greatly from previous works and thus it will not be discussed further.

One of the main requirements for a functioning locomotion stack is to accurately determine the contact state of the robot at any given time. In this work, the robot is assumed to have contact two sensors, one at the toe and one at the heel to measure the contact forces in the z direction. A foot is labeled as in contact if either of the force sensors exceeds a threshold.

The contact sensor signal may be noisy which may induce flickering when near the threshold as evidenced by Figure 2.2. When this occurs, the contact state will oscillate and cause

Algorithm 1 Contact Debouncing Algorithm

```
1: Input: $\delta t, c, T_{debounce}, t_{in\_contact}, t_{no\_contact}$
2: Output: $\hat{c}$
3: Initialize: $\hat{c} \coloneqq False,$
4: if $c$ then
5:     $t_{in\_contact} \mathrel{+}= \delta t$
6: else
7:     $t_{no\_contact} \mathrel{+}= \delta t$
8: end if
9: if $t_{in\_contact} \geq T_{debounce}$ then
10:     $\hat{c} \coloneqq True$
11:     if $c$ then
12:         $t_{no\_contact} \coloneqq 0$
13:     end if
14: end if
15: if $t_{no\_contact} \geq T_{debounce}$ then
16:     $\hat{c} \coloneqq False$
17:     if $!\ c$ then
18:         $t_{in\_contact} \coloneqq 0$
19:     end if
20: end if
```

false detection of liftoff or touchdown events, ruining the timing of the walking gait. As such, it is necessary to add a signal debouncer to hold the signal at its current state until the noisy signal has spent enough consecutive time above the threshold.

For any given contact sensor, the thresholded contact measurement, c is put through the debouncing algorithm shown in Algorithm 1 to produce an estimated contact signal, $\hat{c}$. The algorithm works by incrementing the consecutive amounts of time spent in either contact or

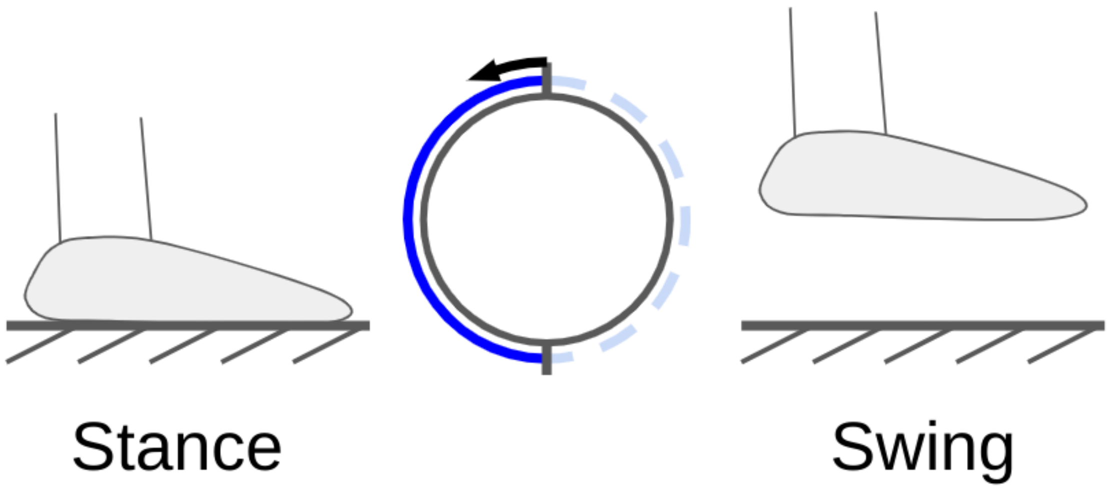


Figure 2.3: Conceptual overview of the gait. Since most locomotion attributes are cyclical, each gait can be represented by a circle with portions of the circle dedicated as subphases. In the case of legs, there are only two subphase: stance and swing

not in contact denoted by $t_{in\_contact}$ and $t_{no\_contact}$, respectively. These valeus are then checked against a minimum time threshold, $T_{debounce}$, to compare whether or not the signal has spent enough consecutive time in a given state. Once that has been acheived, the debouccned measurement can be reliably set. If the state has been confirmed through this thresholding, it is important to reset the opposing consecutive to show that the signal must spend at least $T_{debounce}$ seconds in the other state to once again change. $T_{debounce}$ was set to about 50 ms. As shown by Figure 2.2, the signal is much cleaner and can be used to estimate the contact state.

2.3 Gait

Most forms of locomotion are cyclical processes wherein a set of contact timings is repeated over and over [36]. During walking over flat terrain, a single leg will transition from stance to swing and back to stance in a predictable, repeated manner. Mathematically, this can be

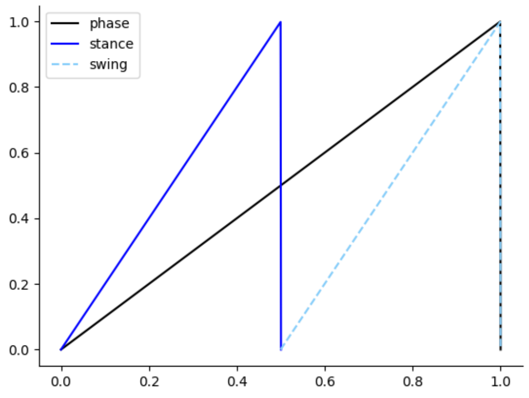


Figure 2.4: The gait can be graphically presented by monotonically increasing from 0 to 1 before being reset back to 0 upon completion. The subcycles immediate switch to the next subcycle upon completion as well.

acheived with the following formula describing the phase:

$$\phi = \frac{t - t_0}{T_{gait}} \tag{2.1}$$

where $\phi \in [0, 1]$ is the gait phase variable, t_0 is the starting time of the current gait cycle and T_{gait} is the gait period. The phase variable, ϕ increases monotonically from 0 to 1. When the variable reaches 1, t_0 is modified to start another gait back at $\phi = 0$. The phase is a dimensionless quantity that can be interpreted as the percentage completion of the cycle.

Each gait can be split into subcycles each with their own subphases. The gait will sequentially cycle through each subcycle before restarting at the first subcycle. In the case of a single leg, the two subcycles are stance and swing. Each variable can be specified as follows:

$$\phi_{stance} = \frac{t - t_0}{T_{swing}} \qquad \phi_{swing} = \frac{t - t_0}{T_{stance}} \tag{2.2}$$

where ϕ_{stance} is the stance phase variable, T_{stance} is the stance time period, ϕ_{swing} is

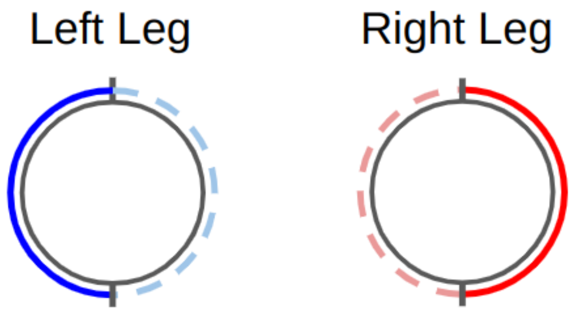


Figure 2.5: With symmetric walking, the goal is to always have the robot supported by a single leg. Each leg has its own gait cycle phase shifted exactly half a gait cycle apart. Thus, when one leg is in contact, the other is travelling through the air to the next touchdown point.

the swing phase variable, and T_{swing} is the swing time period. Once the gait of a single leg completes the stance phase, it immediately transitions to swing phase and vice versa. Thus, an open loop gait schedule can be constucted with desired locomotion timings. This quantity can be used to also measure how much time remaining in any given subcycle. Typical progression of the variables can be seen in Figure 2.4 where a single leg experience stance then swing. While the overall phase continues to progress, the subcycles shift to the next subcycle upon completion at $\phi_{stance} = 1$ or $\phi_{swing} = 1$.

In order to facilitate symmetric walking, two gaits, one for the left leg and one for the right leg are developed. The main goal is for the robot to always be supported by a single leg. This results in offsetting the gaits exactly half a gait phase apart such that when the left leg is in contact, the right leg is in swing and vice versa. As a consequence, the timing parameters, T_{swing} and T_{stance} are the same for both legs and $T_{swing} = T_{stance}$. This concept is demonstrated in Figure 2.5. In this manner, while one leg is maintaining contact forces to support the body, the other leg is free to move to the next stabilizing footstep placement.

While an open loop gait cycle, theoretically, is enough for walking, real world events often leads to undesired behavior if left uncorrected. One such case is when one foot contacts the

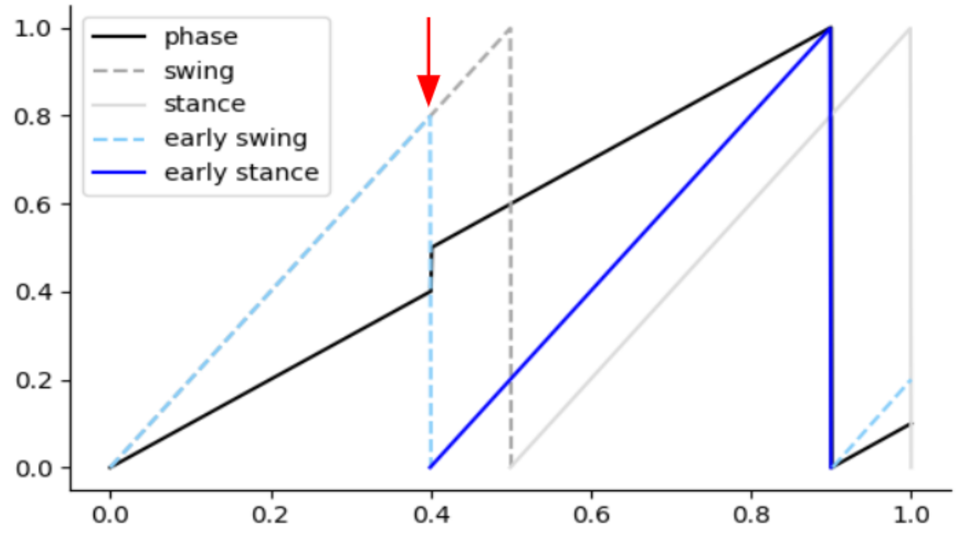


Figure 2.6: An early contact if left unhandled can cause the foot to remain in swing and further push exert an uncessary z force. This will propagate to the next gait cycle and compound the effect, ruining the Linear Inverted Pendulum model assumption. Instead, the gait should be adjusted by resyncing the entire gait to transitino to stance on an early touchdown event.

ground before it is predicted in an event known as ”early touchdown.” If left unchecked, the foot, still believing it is in swing, continues to drive into the ground and imparts uncessary contact forces on to the system. For example, if the right foot experiences an unknown early touchdown, then it will try still move to its desired next footstep location. As a result, the righ foot will drive into the ground causing a positive z force at the contact location on to the robotic system. The robot will then gain angular momentum to start leaning toward the left side. Since no timing corrections are made, the left foot will now experience an even earlier touchdown event causing even more of a disturbance. This effect is an unstable system that exponentially spirals out of control.

To solve the problem, when a swing foot experiences early touchdown, the corresponding gait must be fastforwarded immediately transition into stance. On the other foot, the stance

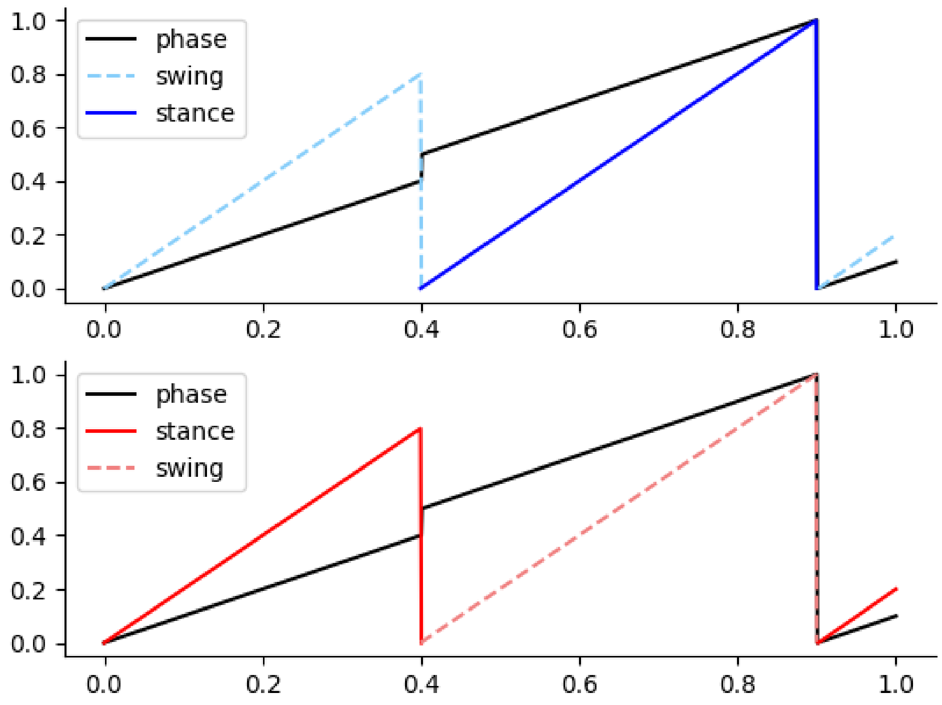


Figure 2.7: On the detection of an early touchdown in the gait on top in blue, the swing leg must immediately transition to stance while the stance leg in the bottom plot in red is commanded to liftoff. This prevents timing errors from propagating in the future.

foot must be commanded to liftoff in order to maintain single support consistency. Figure 2.6 shows an example of a single early touchdown on a single leg and how if left uncorrected, how successive early touchdown events could make problems even worse. Ideally, the early contact events are not of large scale so that the overall gait is predicted with relative accuracy and preempted motions have enough time to fully take place. But with enough care, both legs can be resynced with little effort as shown Figure 2.7.

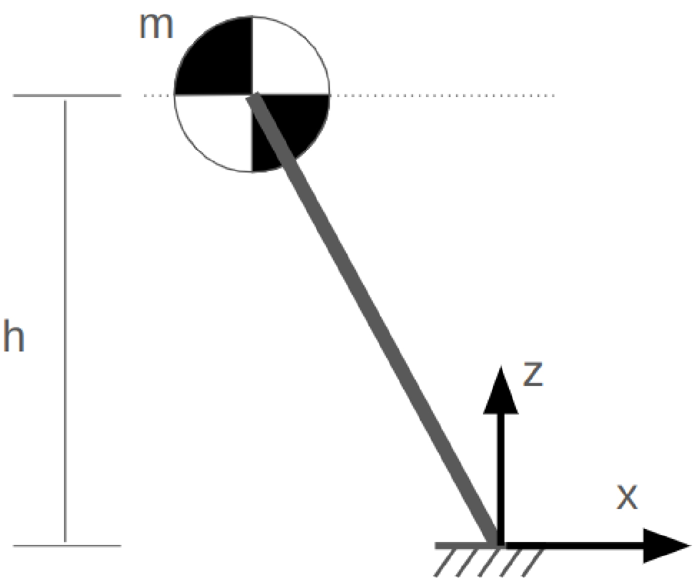


Figure 2.8: The linear inverted pendulum model has a constant height and is a simplified model that approximates walking dynamics well

2.4 Footstep Planning

The main goal of the foostep planning algorithm is to determine where to place the next footstep in order to stabilize the robot. The planner will utilize the linear inverted pendulum model and solely focus on where to place the foothold and assume that the footholds are always reached instantaneously. The problem of how to get each foot to the next location is left to the swing leg planners.

Before discussing the footstep planner, it is first necessary to introduce the linear inverted pendulum (LIP) model which will serve as the basis for the control. The model has served as the backbone for many walking approaches since it approximates human walking well [37] [38]. The main assumptions for the model are to lump all the masses of the robot into a single point mass connected by a rigid link to a point of rotation centered about a single contact point. By consequence, it assumes that the robot has massless legs and that the contact point does not slip. Also, the model assumes that there is an instantaneous change in contact point when transitioning from one foothold to the next. To reduce the complexity of the math, a linearization technique can be used if the height of the pendulum remains fixed. The equation of motion is as follows:

$$\ddot{x} = \frac{g}{h}x = \omega^2 x \qquad \omega = \sqrt{g/h} \tag{2.3}$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ g/h & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{2.4}$$

where x is the position of CoM relative to the contact point, $\dot{x}$ is the velocity of CoM, $\ddot{x}$ is the accelertation of CoM, g is the gravitational constant, h is the height of pendulum, and ω is the natural frequency of the pendulum.

While most algorithms choose to develop foot positions in response to the CoM linear momentum, it has been shown recently that stabilizing relative to the angular momentum about the contact point is a much better proxy for CoM control [30]. As such, the first goal is transform the equations of motion into a form with the angular momentum using the following:

$$L = mh\dot{x} \qquad \dot{L} = mh\ddot{x} = mgx \tag{2.5}$$

$$\begin{bmatrix} \dot{x} \\ \dot{L} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ mg & 0 \end{bmatrix} \begin{bmatrix} x \\ L \end{bmatrix} \tag{2.6}$$

where L is the angular momentum of the CoM about the contact point and $\dot{L}$ is the time derivative of the CoM angular momentum.

The analytic solution for this system is as follows:

$$\begin{bmatrix} x(T) \\ L(T) \end{bmatrix} = \begin{bmatrix} \cosh(\omega(T-t)) & 1/mh\sinh(\omega(T-t)) \\ mh\omega\sinh(\omega(T-t)) & \cosh(\omega(T-t)) \end{bmatrix} \begin{bmatrix} x(t) \\ L(t) \end{bmatrix} \tag{2.7}$$

where t is the current time, and T is the swing time period. Since the ultimate goal of the foostep planner is to find the desired contact position relative to the CoM for the next step to acheive the desired velocity, the second equation is further examined:

$$L(T) = mh\omega\sinh(\omega(T-t))x(t) + \cosh(\omega(T-t))L(t) \tag{2.8}$$

Remember that $x(t)$ is the CoM position relative to the contact point while a footstep position is the contact point realtive to the CoM. Since they are realtive just by a simple transformation, $x(t)$ should be solved for. Upon examination of the resulting equation, only has $L(t)$ and $L(T)$ are left as unknown. Fortunately, $L(t)$ is a measureable quantity based on the current system state. Only $L(T)$ must be caluclated as a desired value.

For the longitduinal direction, the desired angular momentum is just a function of the desired linear velcoity in the forward direction:

$$L_{x,des} = mhv_{x,des} \tag{2.9}$$

Meanwhile for the lateral direction, it can be shown that an oscillation around the center point must acheived by reaching the same CoM position in two footsteps [30]. Thus, the desired angular momentum is as follows:

$$L_{y,des} = mhv_{y,des} \pm mhW\frac{\omega\sinh(\omega T_{swing})}{1+\cosh(\omega T_{swing})} \tag{2.10}$$

where W is the desired step width and T_{swing} is the swing period.

With a chagne of variable of $x(t) = p(t)$ for clarity, the equation for each footstep is as follows:

$$p(t) = \frac{L_{des}(T_{swing}) - \cosh(\omega(T_{swing} - t))L(t)}{mh\omega \sinh(\omega(T_{swing} - t))} \tag{2.11}$$

In order to develop an appropriate CoM trajectory, the system dynamics in Equation 2.3 are simply integrated.

2.5 Swing Leg Trajectories

In the previous section, the LIP model only worried about where to position the next footstep, not the problem how to get there. During walking in the physical world, however, each contact point cannot instantaneously travel from point to point to establish a new contact. Instead, the foot must be lifted from the current contact frame to the next using a trajectory. The trajectory must be smooth and low enough acceleration such that the working assumption can be maintained, yet aggressive enough to correctly track desired footstep commands for the next touchdown location. An added constraint must be that they should be computationally inexpensive as this must be computed online for each timestep that the leg must travel through.

As such, polynomial trajectories have been simple, yet effetive tools for swing leg trajectory generation [39] [40]. When they are formed as boundary problems with initial and final conditions, they offer smooth and computationally efficient trajectories for spatial swing leg quantities (x, y, z). To simplify the effects of different timesteps on the trajectory, the polynomials are normalized with respect to the dt. A unittless phase value $\tau \in [0, 1)$ is then used as the input for each trajectory. This simplifies the specification as any timestep and swing period can be attained by post mulitplication and scaling without changing the problem formulation. As a side effect, the phase variable ϕ_{swing} developed in Section 2.3 can be directly used.

2.5.1 Cubic Polynomial (x,y)

The cubic polynomial is the simplest swing leg trajectory that can be implemented while satisfying initial and final position and velocity constraints. This allows the swing leg to start at the current foot location travel to the final footstep location with zero velocity conditions. This is mainly used for the (x,y) position as further constraints cannot be imposed. For the z direction, two cubic splines have been used where one is the lift portion of the step and the other is downswing of the leg. However, for this work, a higher order polynomial simplified the process with only slight increase in computation. The cubic polynomial for position with its derivatives for velocity and acceleraton are as follows:

$$p(\tau) = c_3\tau^3 + c_2\tau^2 + c_1\tau + c_0 \tag{2.12}$$

$$v(\tau) = 3c_3\tau^2 + 2c_2\tau + c_1 \tag{2.13}$$

$$a(\tau) = 6c_3\tau + 2c_2 \tag{2.14}$$

The boundary conditions are to start at the current foot position p_0 and end at the desired final foot position p_f with zero takeoff and touchdown velocity. For generalization, the boundary conditions for an intial velocity v_0 and final velocity v_f are shown:

$$p(0) = p_0 \qquad\qquad p(1) = p_f \tag{2.15}$$

$$v(0) = v_0 \qquad\qquad v(1) = v_f \tag{2.16}$$

Based on the transformation into the unitless τ variable, the matrix relating any new updated values for either starting or ending foot position is a constant matrix. Thus, the inverse of this matrix can be computed apriori and each subsequent call is very inexpensive.

$$c_0 = p_0 \quad c_1 = 0 \tag{2.17}$$

$$\begin{bmatrix} 1 & 1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} c_3 \\ c_2 \end{bmatrix} = \begin{bmatrix} p_f - p_0 - v_0 \\ v_f - v_0 \end{bmatrix} \tag{2.18}$$

$$c_0 = p_0 \quad c_1 = 0 \tag{2.19}$$

$$\begin{bmatrix} c_3 \\ c_2 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} p_f - p_0 - v_0 \\ v_f - v_0 \end{bmatrix} \tag{2.20}$$

After solving for the coefficients, they can be used in Equation 2.14 to calculate the position and velocity of the swing leg foot position given the current swing phase variable ϕ_{swing}.

2.5.2 Fifth Order Polynomial (x,y)

While a cubic polynomial can be used, upon exmaniation of the curve it generates in Figure 2.9, it is still changes quite aggressive near the takeoff (beginning) and touchdown (end) events. In order to reduce the amount of impact forces imparted, particularly on early touchdown events, a smoother, higher order function should be used. In a very similar manner to to the cubic polynimal, a quintic spline boundary problem can also be specified. Due to the higher order terms, the boundary problem also now includes specifications for the initial acceleration a_0 and the final acceleartion a_f which are set to 0.

$$p(\tau) = c_5\tau^5 + c_4\tau^4 + c_3\tau^3 + c_2\tau^2 + c_1\tau + c_0 \tag{2.21}$$

$$v(\tau) = 5c_5\tau^4 + 4c_4\tau^3 + 3c_3\tau^2 + 2c_2\tau + c_1 \tag{2.22}$$

$$a(\tau) = 20c_5\tau^3 + 12c_4\tau^2 + 6c_3\tau + 2c_2 \tag{2.23}$$

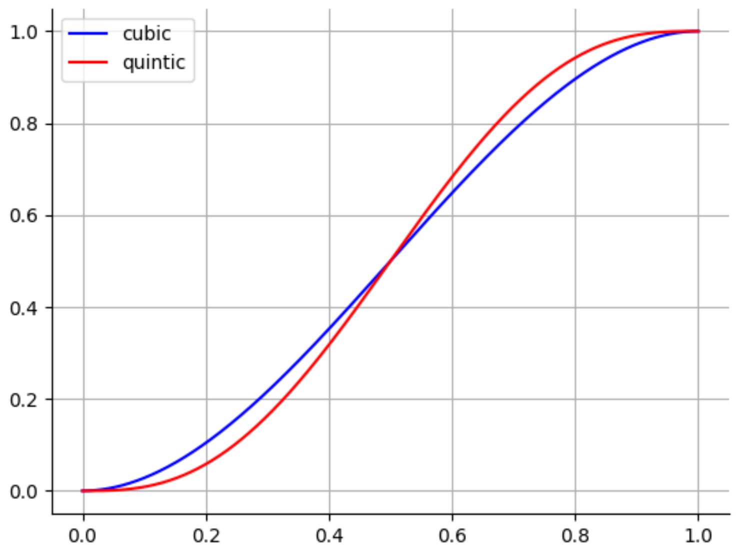


Figure 2.9: Sample swing leg trajectories for cubic (blue) and quintic (red) polynomials

$$p(0) = p_0 \qquad p(1) = p_f \tag{2.24}$$

$$v(0) = v_0 \qquad v(1) = v_f \tag{2.25}$$

$$a(0) = a_0 \qquad a(1) = a_f \tag{2.26}$$

Similar to before, the matrix inversion to solve the linear system of equations can be precomputed ahead of execution time to becom more computationally efficient.

$$c_0 = p_0 \quad c_1 = v_0 \quad c_2 = \frac{1}{2}a_0 \tag{2.27}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 5 & 4 & 3 \\ 20 & 12 & 6 \end{bmatrix} \begin{bmatrix} c_5 \\ c_4 \\ c_3 \end{bmatrix} = \begin{bmatrix} p_f - p_0 - v_0 - \frac{1}{2}a_0 \\ v_f - v_0 - a_0 \\ a_f - a_0 \end{bmatrix} \tag{2.28}$$

$$c_0 = p_0 \quad c_1 = v_0 \quad c_2 = \frac{1}{2}a_0 \tag{2.29}$$

$$\begin{bmatrix} c_5 \\ c_4 \\ c_3 \end{bmatrix} = \begin{bmatrix} 6 & -3 & 0.5 \\ -15 & 7 & -1 \\ 10 & -4 & 0.5 \end{bmatrix} \begin{bmatrix} p_f - p_0 - v_0 - \frac{1}{2}a_0 \\ v_f - v_0 - a_0 \\ a_f - a_0 \end{bmatrix} \tag{2.30}$$

Upon comparison in Figure 2.9, the fifth order spline is much less aggressive at the boudnaries and is a more suitable swing trajectory to use.

2.5.3 Sixth Order Polynomial (x,y,z)

The main constraint on constructing a z swing leg trajectory is that the foot usually must rise above a specified height, the step height, in order to break friction contacts, but also to avoid smaller obstacles on the floor. Thus, to construct a suitable trajectory, an interpolation is required to have an added constraint wherein at some point in the trajectory, a specific value must be met. To simplify calculations the midpoint at $\tau = 0.5$ was chosen. Since another constraint is added, a sixth order polynomial was chosen. In this way, the path is constrained to a midpoint value, or $p(0.5) = p_{mid}$. A similar process can be used to develop the coefficients neecssary to implemnt the polynomial:

$$p(\tau) = c_6\tau^6 + c_5\tau^5 + c_4\tau^4 + c_3\tau^3 + c_2\tau^2 + c_1\tau + c_0 \tag{2.31}$$

$$v(\tau) = 6c_6\tau^5 + 5c_5\tau^4 + 4c_4\tau^3 + 3c_3\tau^2 + 2c_2\tau + c_1 \tag{2.32}$$

$$a(\tau) = 30c_6\tau^4 + 20c_5\tau^3 + 12c_4\tau^2 + 6c_3\tau + 2c_2 \tag{2.33}$$

$$p(0) = p_0 \qquad p(0.5) = p_{mid} \qquad p(1) = p_f \tag{2.34}$$

$$v(0) = v_0 \qquad v(1) = v_0 \tag{2.35}$$

$$a(0) = a_0 \qquad a(1) = a_f \tag{2.36}$$

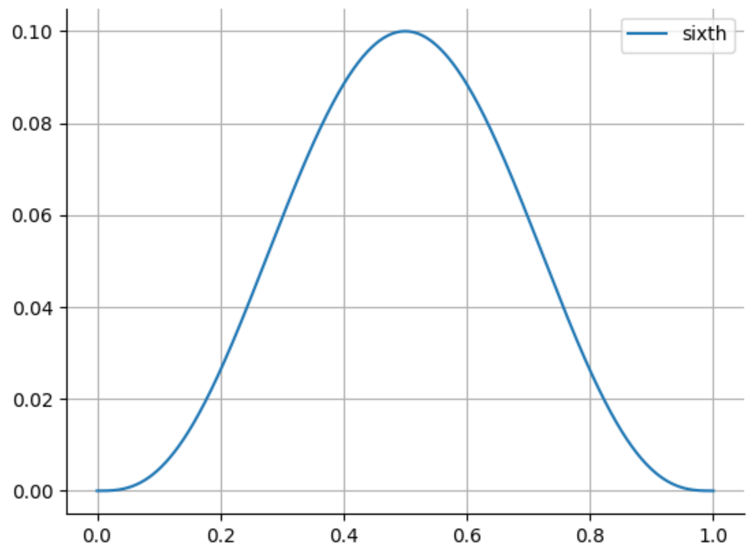


Figure 2.10: A sample sixth order trajectory for the z component of the swing leg trajectory

$$c_0 = p_0 \quad c_1 = v_0 \quad c_2 = \frac{1}{2}a_0 \tag{2.37}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ \frac{1}{2}^6 & \frac{1}{2}^5 & \frac{1}{2}^4 & \frac{1}{2}^3 \\ 6 & 5 & 4 & 3 \\ 30 & 20 & 12 & 6 \end{bmatrix} \begin{bmatrix} c_6 \\ c_5 \\ c_4 \\ c_3 \end{bmatrix} = \begin{bmatrix} p_f - p_0 - v_0 - \frac{1}{2}a_0 \\ p_{mid} - p_0 - \frac{1}{2}v_0 - \frac{1}{2}^3 a_0 \\ v_f - v_0 - a_0 \\ a_f - a_0 \end{bmatrix} \tag{2.38}$$

$$c_0 = p_0 \quad c_1 = v_0 \quad c_2 = \frac{1}{2}a_0 \tag{2.39}$$

$$\begin{bmatrix} c_6 \\ c_5 \\ c_4 \\ c_3 \end{bmatrix} = \begin{bmatrix} 32 & -64 & -10 & 1 \\ -90 & 192 & 27 & -2.5 \\ 81 & -192 & -23 & 2 \\ -22 & 64 & 6 & -0.5 \end{bmatrix} \begin{bmatrix} p_f - p_0 - v_0 - \frac{1}{2}a_0 \\ p_{mid} - p_0 - \frac{1}{2}v_0 - \frac{1}{2}^3 a_0 \\ v_f - v_0 - a_0 \\ a_f - a_0 \end{bmatrix} \tag{2.40}$$

As Figure 2.10 shows, the trajectory is able to reach a certain midpoint value before returning to the desired end location. This can be seen as a foot stepping over obstacles on each swing leg. In the future, this can also be used if the foot is desried to reach a point in either x or y direction as well, as is the case for a kick.

2.6 Whole Body Control

The whole body control problem is a popular scheme to control humanoids at the lowest level to generate output joint accelerations and then converting to joint torques use an inverse dynamics model [41] [42] [43] [44] [45] [46]. The main concept behind whole body control is to use a task to describe a space of objectives to track. This task can be any space as long as a Jacobian and Jacobian time derivative exist. Starting from the definition of the Jacobian and applying a time derivative:

$$\dot{x} = J\dot{q} \tag{2.41}$$

$$\ddot{x} = \dot{J}\dot{q} + J\ddot{q} \tag{2.42}$$

where $\dot{x}$ is the task space velocity, $\ddot{x}$ is the task space acceleration, $\dot{q}$ is the generalized velocity of the robot, $\ddot{q}$ is the generalized acceleration of the robot, J is the task space Jacobian, and $\dot{J}$ is the time derivative of the task space Jacobian. Using this form, task space accelerations can be tracked if a desired value is known. For most cases, a PD controller around the desired task space position and velocity works well:

$$\ddot{x}_{des} = K_p(x_{des} - x) + K_d(\dot{x}_{des} - \dot{x}) \tag{2.43}$$

where $\ddot{x}_{des}$ is the desired task space acceleration, K_p is the proportional error gain, x_{des} is the desired task space position, x is the current task space position, K_d is the damping velocity gain, $\dot{x}_{des}$ is the desired task space velocity, and $\dot{x}$ is the current task velocity. Thus, an error to be minimized can be developed:

$$error = \|J\ddot{q} + \dot{J}\dot{q} - \ddot{x}_{des}\|_W^2 \tag{2.44}$$

Before showing the whole body problem, first, the full rigid body dynamics model must be introduced. Rigid body dynamcis only assume that robot links be rigid and that the actuated torque is the actual output torque at the joint. To describe the system using this form, a generalized state q and generalized velocity $\dot{q}$ must first be defined as in Featherstone [47].

$$q = \begin{bmatrix} x \\ \psi \\ q_j \end{bmatrix} \quad \dot{q} = \begin{bmatrix} v \\ \omega \\ \dot{q}_j \end{bmatrix} \tag{2.45}$$

where $x \in \mathbb{R}_{3\times1}$ is the world frame position, $\psi \in \mathbb{R}_{4\times1}$ is the world frame quaternion, $q_j \in \mathbb{R}_{n_j\times1}$ is the joint positions, $v \in \mathbb{R}_{3\times1}$ is the world frame velocity, $\omega \in \mathbb{R}_{3\times1}$ is the world frame angular velocity, $\dot{q}_j \in \mathbb{R}_{n_j\times1}$ is the joint velocities. The standardized robotics rigid body equation of motion (EoM) for a robot with n_q number of generalized states, n_v number of generalized velocities, and n_c number of contacts with the environment is defined as:

$$M(q)\ddot{q} + c_g(q, \dot{q}) = \tau + J_c^T(q) f_c \tag{2.46}$$

where $q \in \mathbb{R}_{n_q\times1}$ is the generalized state vector, $\dot{q} \in \mathbb{R}_{n_v\times1}$ is the generalized state vector, $M(q) \in \mathbb{R}_{n_q\times n_q}$ is the inertia matrix, $\ddot{q} \in \mathbb{R}_{n_q\times1}$ is the joint accelerations, $C_g(q, \dot{q}) \in \mathbb{R}_{n_q\times1}$ is the nonlinear vector due to gravity and coriolis effects, $\tau \in \mathbb{R}_{n_q\times1}$ is the vector of joint torques, $J_c(g) \in \mathbb{R}_{n_c\times n_q}$ is the contact jacobian, and $F_c \in \mathbb{R}_{n_c\times1}$ is the contact forces. This model is the best approximation of the robot dynamics and is slightly expensive to calculate. However, since whole body control is the lowest level before joint level tracking, it must be as accurate as possible to correctly track task quantities.

Since there are often more desired degree of freedoms across all tasks than there are degrees of freedom in the robot, a compromise must be struck to complete a solution. This can either be solved using a null space projection of the Jacobian on each set of successive tasks or through a weighted optimization to try to satisfy desired values implicitly [41]. In this work, the implicitly weighted version is used as shown below:

$$\min_{\ddot{q},f_c} \quad \sum_{i=1}^{N} \| J_i\ddot{q} + \dot{J}_i\dot{q} - \ddot{x}_{i,des}\|^2_{W_i} + \|\ddot{q}\|^2_{R_{\ddot{q}}} + \|f_c\|^2_{R_f} \tag{2.47}$$

$$s.t. \quad S_f\big(M\ddot{q} + C_g - J_c^T f_c\big) = 0 \tag{2.48}$$

$$G_f f_c \leq 0 \tag{2.49}$$

$$\tau_{min} \leq u \leq \tau_{max} \tag{2.50}$$

$$\dot{q}_{min} \leq q \leq \dot{q}_{max} \tag{2.51}$$

$$q_{min} \leq \dot{q} \leq q_{max} \tag{2.52}$$

where the optimization problem seeks to solve for the desired joint accelerations $\ddot{q}$ and desired contact forces f_c. The cost function is comprised of weighted task space acceleration error norms to ensure that tasks are tracked relative to the importance given by their associated weight W_i. Meanwhile, regularization terms on the optimization variables are placed to ensure that only the minimal amount of input is required to find a solution.

The constraints meanwhile are given as the previously discussed rigid body dynamics to create dynamically consistent solutions. A friction cone constraint matrix G_f ensures that all forces applied are realistically bounded by the amount that friction would allow. Finally, limits are placed on the joint torques u, the joint positions q, and the joint velocites $\dot{q}$ to ensure that any solutions are physcially possible to acheived on the robot. An expanded look at the matrices that comprise the constraints in a convex form is show below:

$$\begin{bmatrix} -S_f C_g \\ \tau_{min} - S_a C_g \\ l_{fc} \\ \dot{q}_{min} - \dot{q}_m \\ q_{min} - (q_m + \dot{q}_m \Delta t) \end{bmatrix} \leq \begin{bmatrix} S_f M & -S_f J_c^T \\ S_a M & -S_a J_c^T \\ 0 & C_{fc} \\ \Delta t I & 0 \\ \frac{1}{2}\Delta t^2 I & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ f_c \end{bmatrix} \leq \begin{bmatrix} -S_f C_g \\ \tau_{max} - S_a C_g \\ u_{fc} \\ \dot{q}_{max} - \dot{q}_m \\ q_{max} - (q_m + \dot{q}_m \Delta t) \end{bmatrix} \tag{2.53}$$

After the solution has been found, joint torque commands τ can be found using inverse dynamics [48]:

$$\tau = M(q)\ddot{q} + c_g(q, \dot{q}) - J_c^T(q) f_c \tag{2.54}$$

In addition to torque commands, position and velocity joint commands must also be generated to enable better tracking. For position commands, the most commonly used approach is inverse kinematics (IK). Since joint chains with more than 4 degrees of freedom becoming increasingly hard to develop analytic solutions, nuemrical solutions have been developed to solve them. In this work, the Weighted Damped Least Squares method [49] is used to solve for joint position give the end effector positions of each of the limbs (left foot, right foot). These joint positions are then fed to the lower level controller. The main update equation of the numerically IK has been written below:

$$\Delta\theta = \tilde{J}^T(\tilde{J}\tilde{J}^T + \lambda^2 I)^{-1} e, \quad \tilde{J} = WJ \tag{2.55}$$

For velocity commands, the psuedo Jacobian method [50] is used as platforms are not guaranteed to have invertible Jacobians. If a 6 DoF arm was used with an invertible Jacobian, the inverse could easily be calculated. Most of the time this is not the case as even with the correct amount of degrees of freedom, usually there exist singularities which cause inverse calculations to become numerically unstable. The output joint velocities are checked to be within some limit in order to avoid these instabilities.

$$\dot{\theta}_{des} = J^{\dagger}\dot{x} = J^{T}(JJ^{T})^{-1}\dot{x}_{des} \tag{2.56}$$

2.7 Joint Level Control

Proprioceptive acuators are the paradigm of motor chosen to the run the stack [16]. As such the lower level joint control utilizes joint position, joint velocity, and joint torque commands sent from higher level planners and calculations. A low level torque controller runs on the platforms in this thesis which uses the current as a proxy for the torque to specify motor control:

$$u_{des} = K_p(\theta_{des} - \theta) + K_d(\dot{\theta}_{des} - \dot{\theta}) + \tau \tag{2.57}$$

CHAPTER 3

Artemis Platform

3.1 Platform Overview

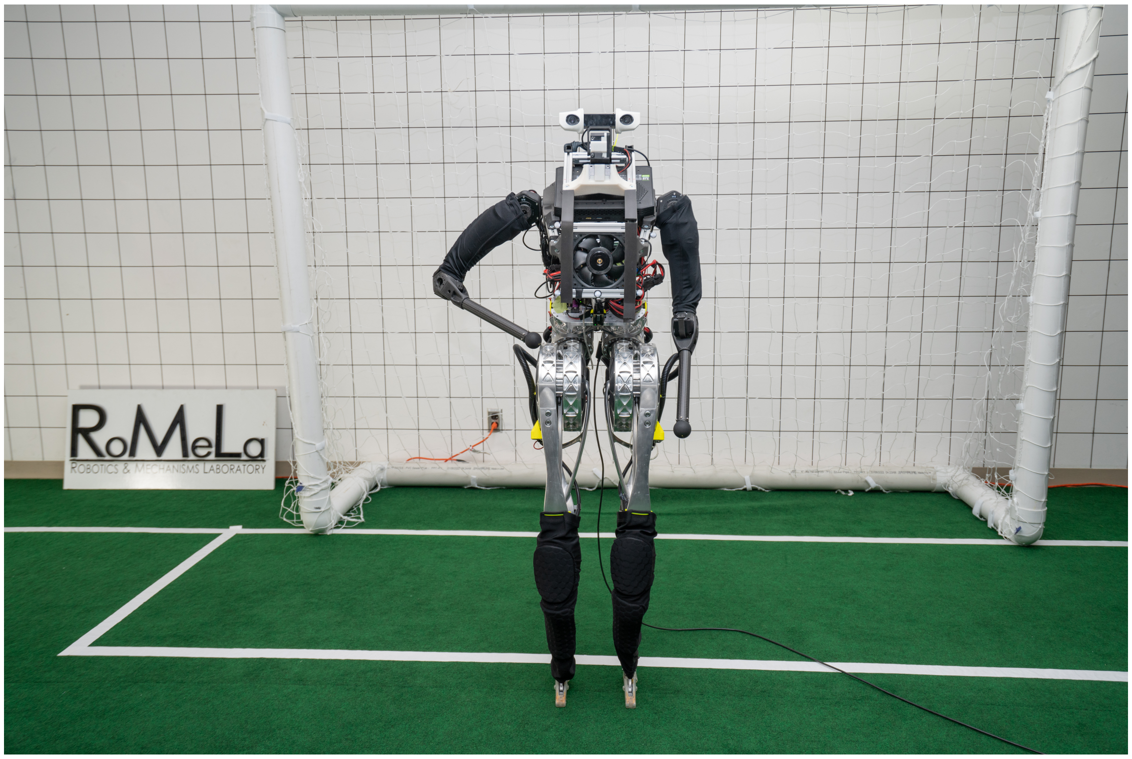


Figure 3.1: ARTEMIS or the Advanced Research Platform with Enhanced Mobility and Improved Stability is a highly capable 20 DoF proprioceptively actuated humanoid

In order to verify the proposed locomotion stack, The Advanced Robotic Technology with Enhanced Mobility and Improved Stability, or ARTEMIS (Artemis), as shown in Figure 3.1, was used. Artemis is a 20 degree of freedom humanoid robot with custom quasi-direct drive

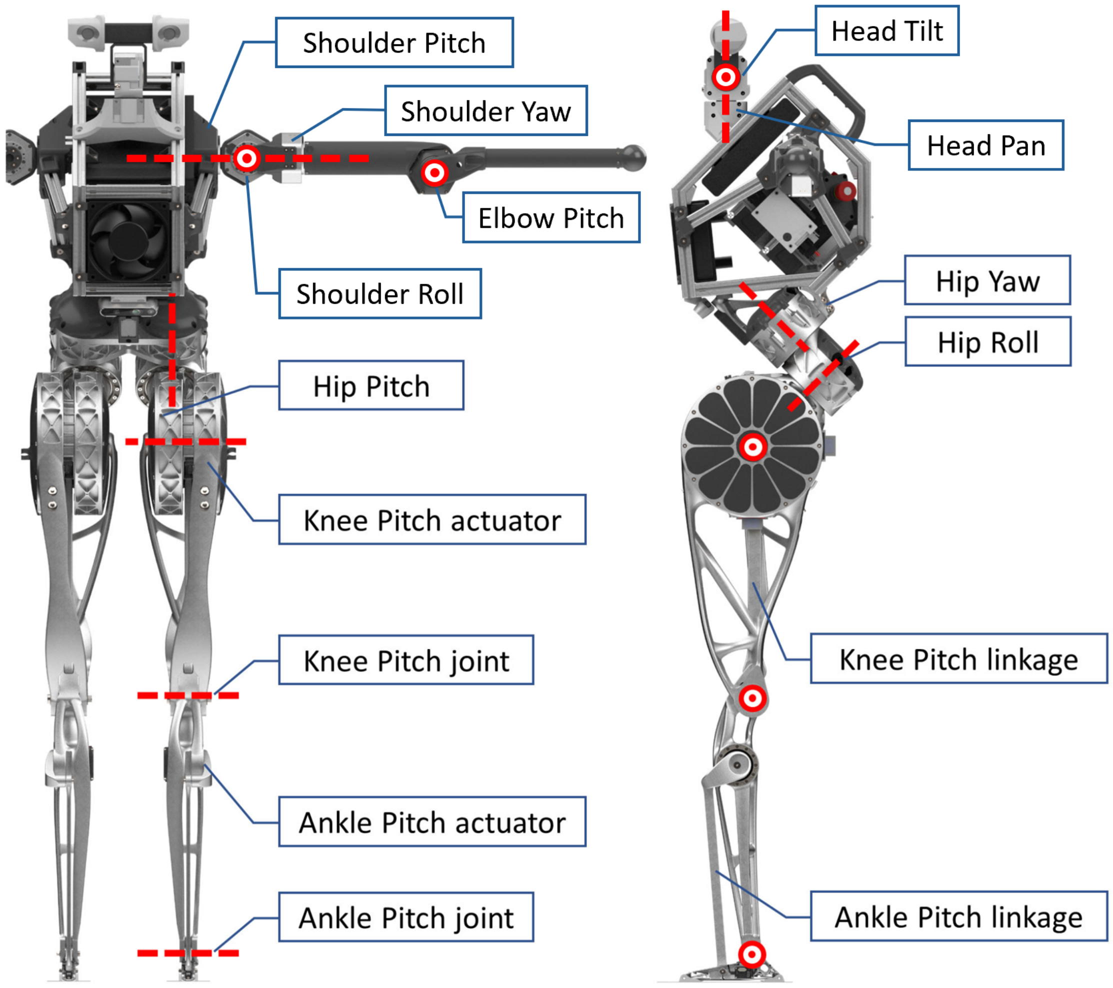


Figure 3.2: Joint configuration for Artemis. The platform has two 5 DoF legs, two 4 DoF arms, and one 2 DoF neck for a total of 20 joints

actuators fully developed at RoMeLa. The main design philosophy behind Artemis was to create a highly dynamic platfrom capable of performing explosive and dynamic movements unseen in previous humanoids. This is achieved by minimizing weight and inertia, especially any distal quantities away from the body, in order to maximize dynamic capability [51]. Large amounts of weight were saved by making the output shaft of each actuator the actuator housing of the next joint at the hip. This provides a highly compact package where the motor housings are the sole structural components of the platform. By also using custom

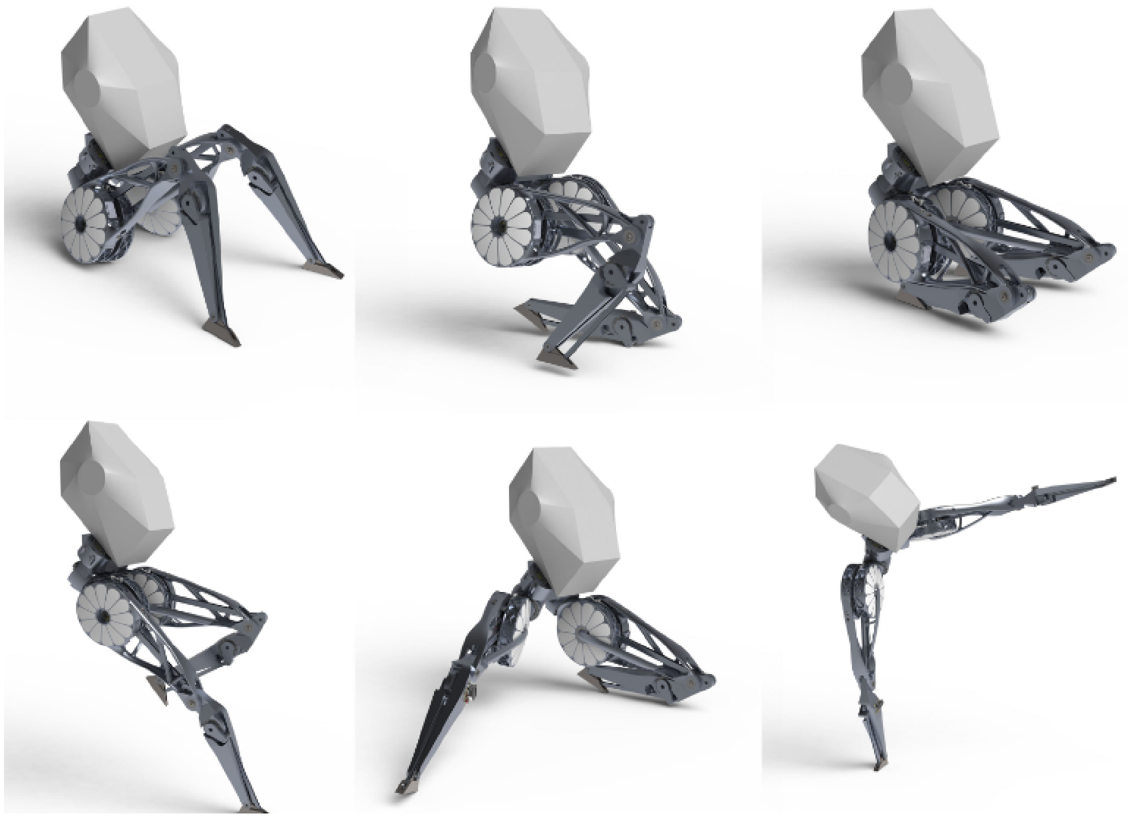


Figure 3.3: Artemis was designed to have a large range of motion, almost comparable to that of an actual human

Back-drivable Electromagnetic Actuators for Robots, or BEAR actuators, Artemis is able to withstand high impact forces while providing clear torque fidelity on actuation output.

Artemis has two 5 degree of freedom (DoF) legs, two 4 degree of freedom arms, and 2 degrees of freedom in the neck (yaw, pitch) as shown in Figure 3.2. Having only 5 degrees of freedom in the legs differs from the standard of humanoid platforms as it is missing the pitch roll actuator. This was done on purpose to reduce distal mass as dynamic fast paced movements were prioritized. While this does save a considerable amount of inertia at the ankle, this prevents ARtemis from fully controlling the the CoM state when one leg is contact.

Another major design consideration was to allow Artemis to exhibit a much larger range of motion in its legs compared to traditionally designed humanoids. As can be seen by Figure

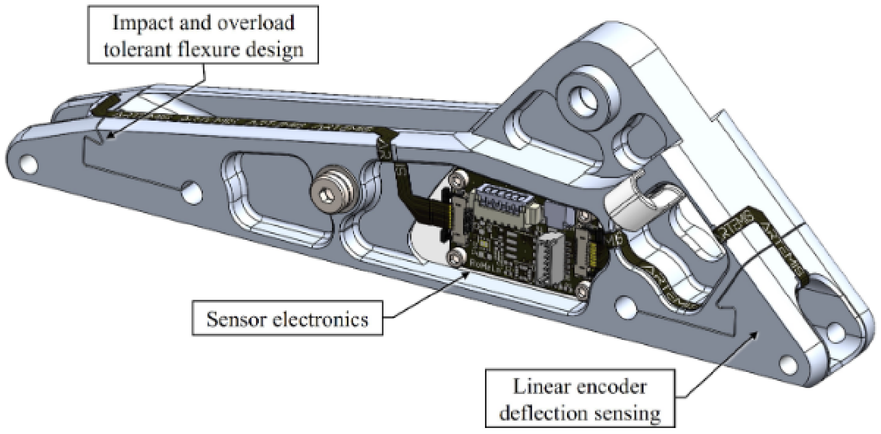

Impact and overload tolerant flexure design
Sensor electronics
Linear encoder deflection sensing

Figure 3.4: Artemis has flexure based contact sensors that allow accurate, timely detection of touchdown events. The flexures also bottoms out to prevent damage from larger impact forces

3.3, Artemis has quite a large workspace for its legs and can reach very similar poses to that of an actual human being. This allows for more intuitive task specification as human poses can be more easily translated to commands on the platform.

Meanwhile, the platform is equipped with a sensor suite containing the StereoLabs Zed 2i RGB-D stereo camera, the Lord Microstrain 3DM-CV7 IMU, and custom flexure based contact sensors. Each contact sensor, one at the toe and another at the heel, measures the deflection of an aluminum member that can be converted through calibration into a force as shown in Figure 3.4. The contact sensors have a hard stop where the deflection leads to compression against a structure to prevent plastic deformation from occurring. This provides very reliable and timely contact information about the robot that is robust to weaknesses of other methods such as errant forces during swing or breaking under large impact forces. Additionally, Artemis can be equipped with forward and rear facing depth cameras for further environmental awareness, although these are absent during the Robocup competition to follow regulations.

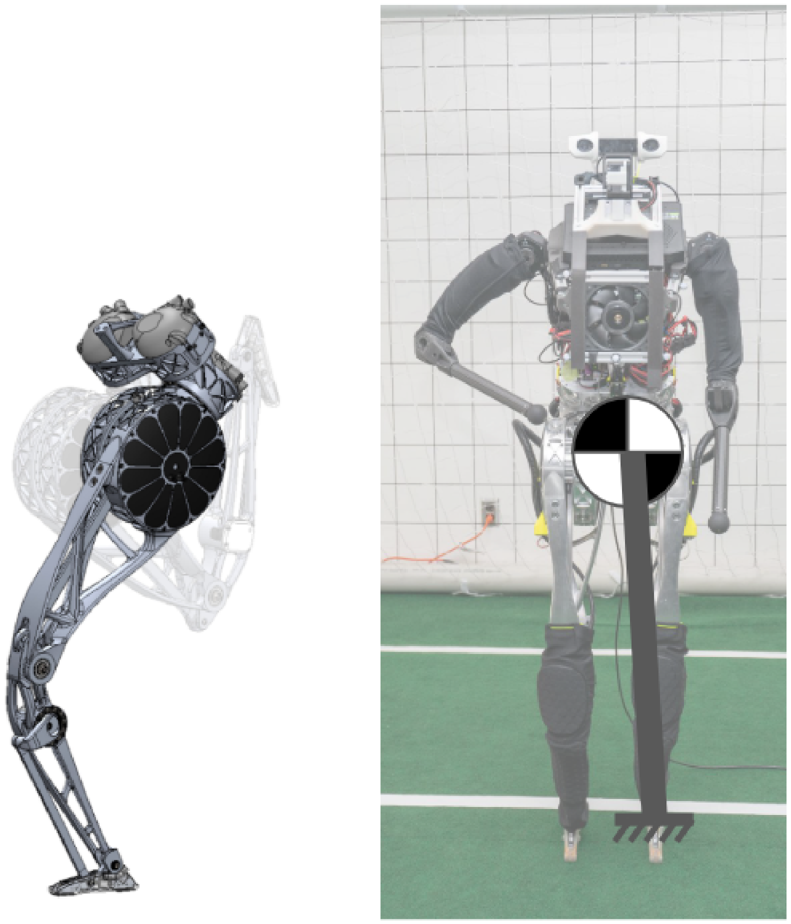


Figure 3.5: Since Artemis only has a 5 DoF leg (no ankle roll), the platform behaves very similarly to the LIP model in real life due to the lower distal leg inertia and the ankle naturally rolling about the contact point

All these design considerations were conducted in an effort to also match the LIP model as shown by Figure 3.5. By more closely designing hardware to match the underlying template models, the control approach that uses that model will be more effective.

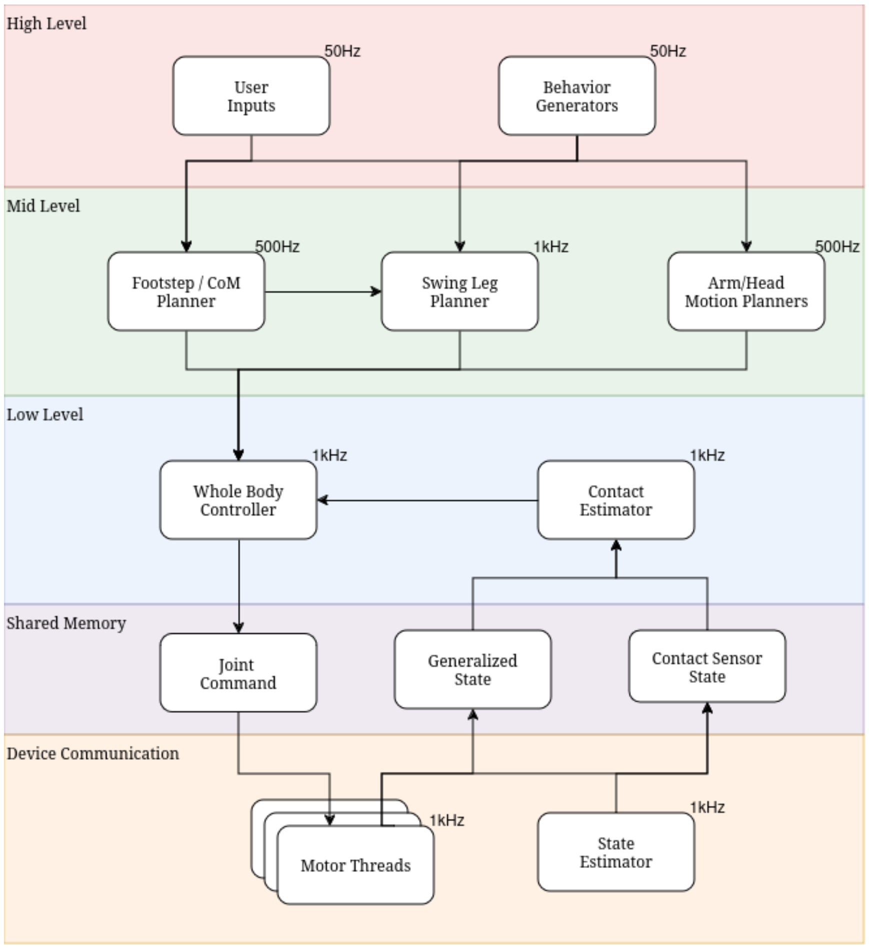


Figure 3.6: Overall diagram for the implementation of the Artemis locomotion stack

3.2 Artemis Implementation Details

The software stack that depicted in Figure 3.6 was implemented on Artemis to make the robot walk and kick. Due to the simplication of the techniques outlined, threads were able to run quickly at 500Hz - 1kHz. Each individual block listed constituted a separate thread that was run in parallel with the rest of the stack.

To facilitate communication across all threads, a custom shared memory library built off of POSIX interfaces found in Boost C++ libaries [52]. While it is only explicitly shown as a bridge between the whole body controller and the motor threads, each interface between any given threads used shared memory. By having each component separated by shared memory, modules can be designed, implemented, and tested relatively in a decoupled manner. In addition, components can be reused across different robots given the same underlying interfaces do not change.

In order to facilitate communication with other higher level libraries, each thread was wrapped in a ROS 2 node [53] with topics exposed to report the state of the robot and to offer interfaces for higher level commands like desired velocity. Due to the sensitive timings that torque control require, it was a decision design to disallow external ROS 2 topics from directly specifying joint comands to the motors directly. The amount of error handling necessary to ensure that two controllers were not commanding to the motors at the same time was deemed too costly to implement during this work.

The BEAR actuators are controlled using the CBEAR SDK from Westwood Robotics [54]. This is a C++ library that facilitates serial communication over the RS485 devices to read/write joint position, velocities, torques among other registers. A motor thread runs at 1kHz and follows direct force commands as tracking can be improved when using joint level feedback with target trajectories [55].

All kinematics and dyanmics measurements are computed using the Pinocchio C++ library [56]. This library offers very efficient computation of most dynamic quanities relevant to the control of a humanoid including the centroidal momentum matrix [57]. For the convex optimization of the whole body controller, the ProxQP C++ library was used for its ease of use and speed [58].

The most difficult part about the locomotion stack is tuning the whole body controller as often times it is difficult to determine exactly which task needs to be prioritized in relation to the other tasks. Table 3.1 shows the gains found to stablize the stack both in double support or balancing and when the robot was walking in continuous single support. During balancing, the parameters were relatively straightforward as there are a more than enough degrees of freedom between the two legs to satisfy any CoM commands required to prevent

Task	Double Support (Balancing)	Single Support (Walking)
	(10, 10, 100)	(1, 1, 100)
CoM Linear Momentum	(30, 40, 100)	(0, 0, 100)
	(25, 10, 15)	(10, 10, 15)
	(10, 10, 1)	(20, 20, 20)
CoM Angular Momentum	(0, 0, 0)	(0, 0, 0)
	(10, 10, 1)	(10, 10, 10)
	(80, 80, 80)	(20, 20, 40)
Body Orientation	(300, 300, 200)	(150, 150, 100)
	(40, 50, 20)	(30, 30, 10)
	(0, 0, 0)	(80, 80, 80)
Swing Foot Position	(0, 0, 0)	(1500, 1500, 1000)
	(0, 0, 0)	(5, 5, 20)
	(0, 0, 0)	(1, 10, 10)
Swing Foot Orientation	(0, 0, 0)	(0, 500, 500)
	(0, 0, 0)	(0, 25, 25)
	(1000, 1000, 1000)	(1000, 1000, 1000)
Contact Position	(0, 0, 0)	(0, 0, 0)
	(0, 0, 0)	(0, 0, 0)
	(10, 10, 10, 10)	(10, 10, 10, 10)
Arm Joint Positions	(100, 100, 100, 100)	(100, 100, 100, 100)
	(20, 20, 20, 20)	(20, 20, 20, 20)
	(10, 10)	(10, 10)
Head Joint Positions	(100, 100)	(100, 100)
	(20, 20)	(20, 20)

Table 3.1: Weights and gains used for the whole body controller

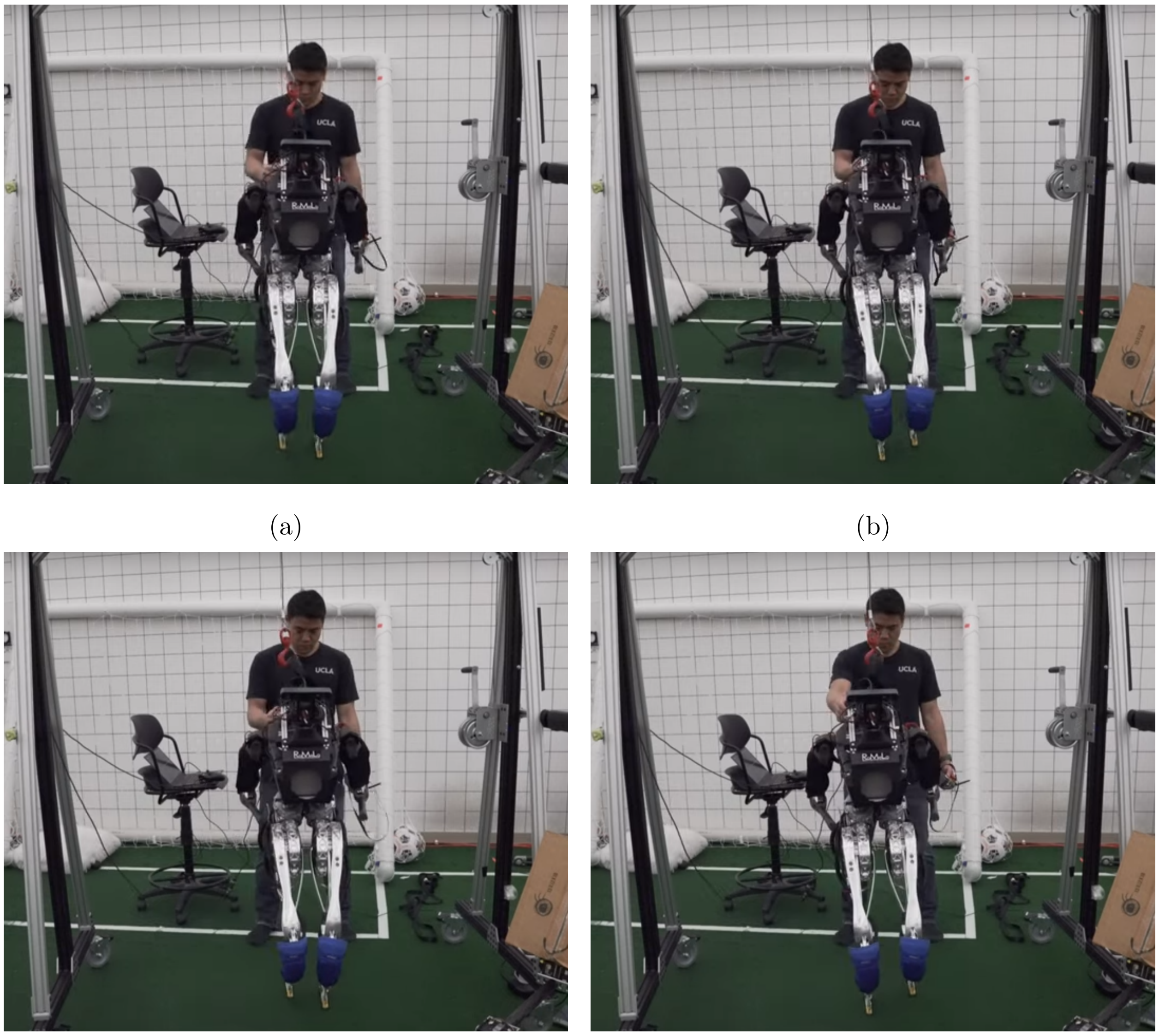

(a) (b) (c) (d)

Figure 3.7: Artemis walking in the lab with minimal intervention from a human bystander

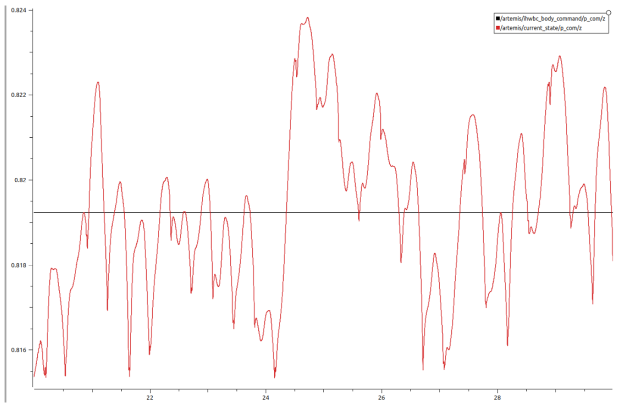
the robot from falling.

While walking, however, multiple tasks need to be tracked using only a single 5 DoF leg providing contact forces as control inputs. The main starting point was that it is well known that the CoM angular momentum should be minimized during walking [59]. Through multiple experiments, it was found that the prioritizing tracking the CoM z height and the footstep swing location were the most crucial in stabilizing the platform. Since Artemis is always at an unstable equilibrium without the ability to resist roll motion about the ankle, it is imperative that the touchdown location be as close as possible to the desired location as calculated by the footstep planner. As a consequence, in order to make the footstep planner as effective as possible, the CoM z height must be kept as close as possible to a constant to match the assumptions of the underlying LIP model.

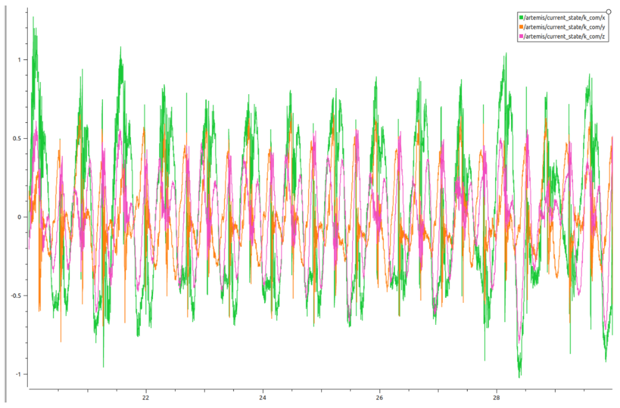
3.3 Walking

With the software stack thus outlined, Artemis performed stable walking with a human only standing by for safety of the robot as shown in Figure 3.7a-3.7d. Artemis was able to walk stably for a recorded 20 minutes before it needed to be stopped to recharge batteries. Even through small disturbances applied to the chest and the legs, the platform did not fall over.

As was mentioned during tuning of the whole body, the quantities that mattered the most



(a)



(b)

Figure 3.8: The whole body controller tracked CoM quantities well. Data is shown over 10 seconds of walking. (a) T z com height in particular was tracked within a 1cm of noise as shown by the actual CoM height in red near the commanded height in black. (b) The angular momentum, on the other hand, oscillated around zero and was minimized

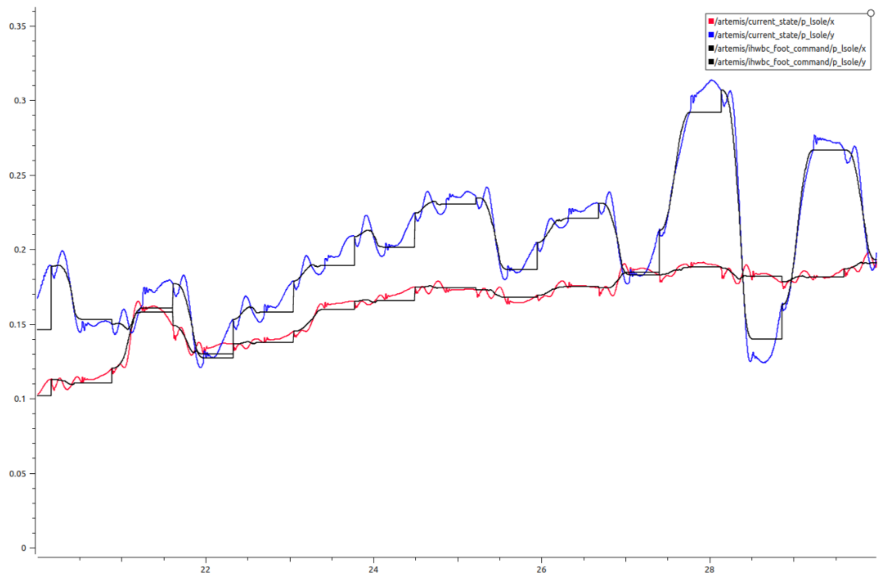


Figure 3.9: Meanwhile footstep comamnds were tracked well throughout walking to ensure the platform stabilized through proper footstep placement. The data is conducted over 10 seconds with the black lines denoting commanded values, the blue line representing the left foot x direction and the red line representing the left foot y direction. The data was similar for the right foot

to enable the stabilizing through footstep calculation were to track the CoM z height and the footstep (x,y) position while minimizing the CoM angular momentum. Figure 3.8a-3.8b shows tracking of the related CoM quantities by the whole body controller over 10 seconds of walking. The CoM z height was maintained with a 1cm of the desired value while the CoM angular momentum was minimized to a maximum value of 1.3 $kg \cdot m^2/s$. Meanwhile, Figure 3.9 shows the tracking of the left foot by the whole body controller. Throughout the sampled time, the tracking follows very closely and only deviates 2-3cm from the desired values. The values for the right foot looked very similar and were left out for brevity. This ensures that the feet consistently reached the desired footstep locations and maintained the assumptions for the footstep planner to remain valid.

A point that was stressed during the discussion of the gait in Section 2.3 was the fact that

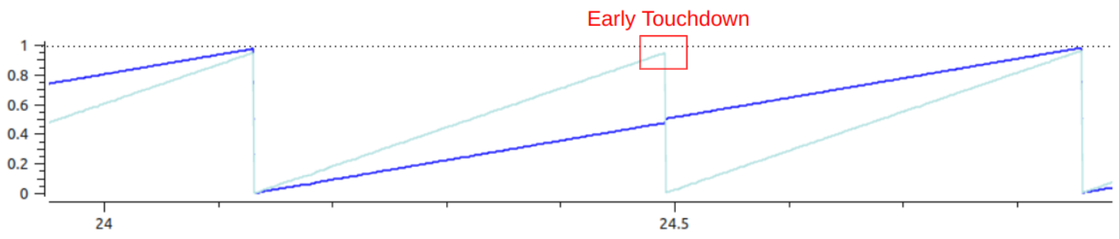


Figure 3.10: Artemis was able to readjust the gait timings in order to resync on each early touchdown event. As can be seen, the gait cycle never reaches completion by meeting a $phi = 1$ value.

the gait needed to be able to handle early touchdowns in order prevent escalating desyncing of the gait. Figure 3.10 shows the result of the gait handling the early touchdown event. Walking was performed without this contact feedback initially and would never stabilize as Artemis still believed the newly contact support was still in the air. This would cause the leg to drive further into the ground, leading to large z forces on the platform. Each step would exacerbate the problem as the timing continued to get further and further mismatched.

3.4 Kicking

Kicking was implemented in order to satisfy later goals of competing in automous soccer. By using the same tools that have already been discussed, a simple, yet effective kick can be developed. The sixth order polynomial that was discussed in Section 2.5 mainly used for the z height swing leg trajectory can also specified to extend the foot out in the x (forward) direction to make contact with a ball. If the ball position is known, the midpoint can be adjusted to match as close to that position as possible only limited by kinematics. Ball positions outside of this range require the robot to walk closer in order to properly kick.

Kick power and how hard the ball is contacted is also tunable solely through the specification of the midpoint. By kicking ”through the ball” to a point past the surface of the

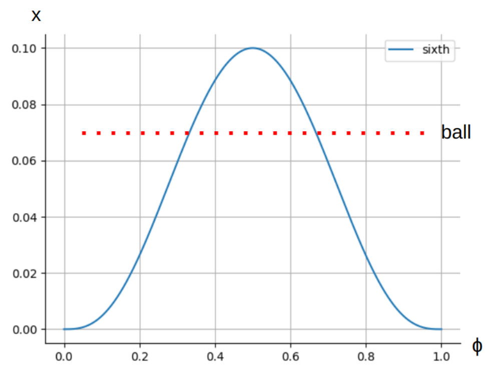
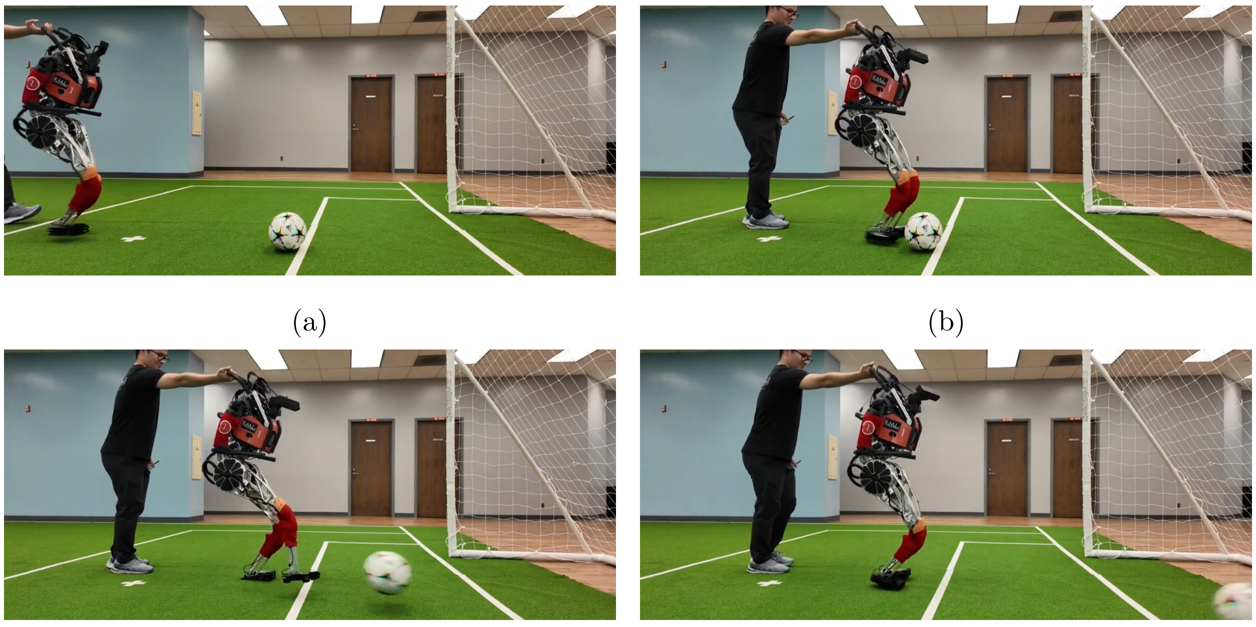


Figure 3.11: A strong kick was implemented for Artemis using seventh order polynomial splines to detemine how far to extend the foot in the x (forward) direction based on the ball position.

ball as demonstrated in Figure 3.11, the foot carries more velocity and ends up with a more powerful kick. This is a relatively crude, but effective method in order to increase power. Unfortunately, the distance travelled by the kick does not correlate linearly with the amount of specified depth due to the noise of the ball position measurement, the esimated state of the robot, and tracking of the swing leg on impact with the ball. Since this prove too difficult to properly calibrate, two sets of kicks were specified: a weak kick, with a depth of 5cm past the ball surface, used to relocate the ball and a strong kick, with a depth of 10cm past the ball surface, to strongly kick the ball forward.

Kicking was successfully performed both in a controlled lab environment and at the later competiton as shown in Figure 3.12a-3.12d. During testing, the strong kick ended up being able to travel nearly three quaters of the field without intervention. This distance, however, varied greatly based on the thickness of the grass playing field, how centered on the ball the kick contact point was, and the type of ball used. Holding all those variables constant, however, produced a reliable and repeatable strong kick.



(a) (b) (c) (d)

Figure 3.12: Artemis kicked a ball on a test field to verify the behavior. (a) Artemis wakling up to the ball, (b) Artemis preparing to kick, (c) the swing foot extends through the ball position, and (d) returning back down at the footstep location

CHAPTER 4

Robocup 2024 Eindhoven

4.1 Overview

One of the major ways to prove the capability of a robotic platform is to apply it to a real world, high pressure environment. Competitions like the DARPA Robotics Challenge (DRC) and Robocup are proving grounds to ensure that algorithms not only work, but that they are reliable under harsh conditions where real world variables come to play. Additional levels of reliability and robustness are required from both hardware and software in order to succeed.

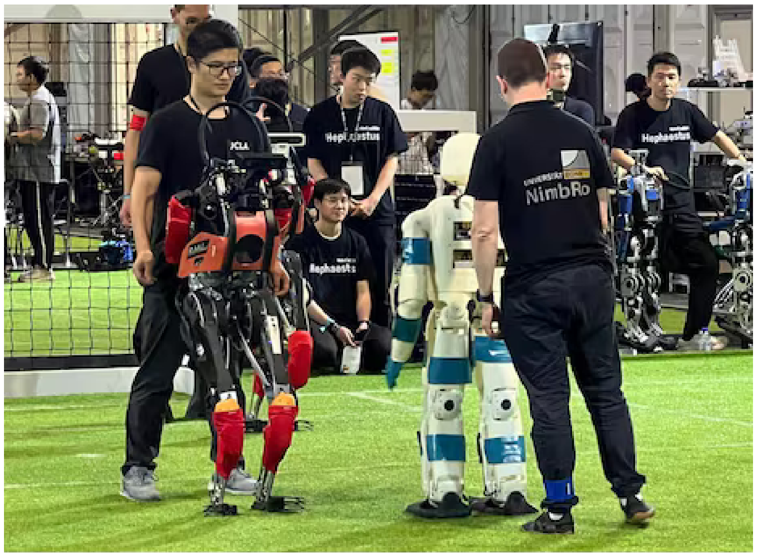


Figure 4.1: Robocup pits two teams of autonomous robots to face off in full games of soccer. The only allowed humand intervention is to pick robots off the field after failures to be reset on the sidelines.

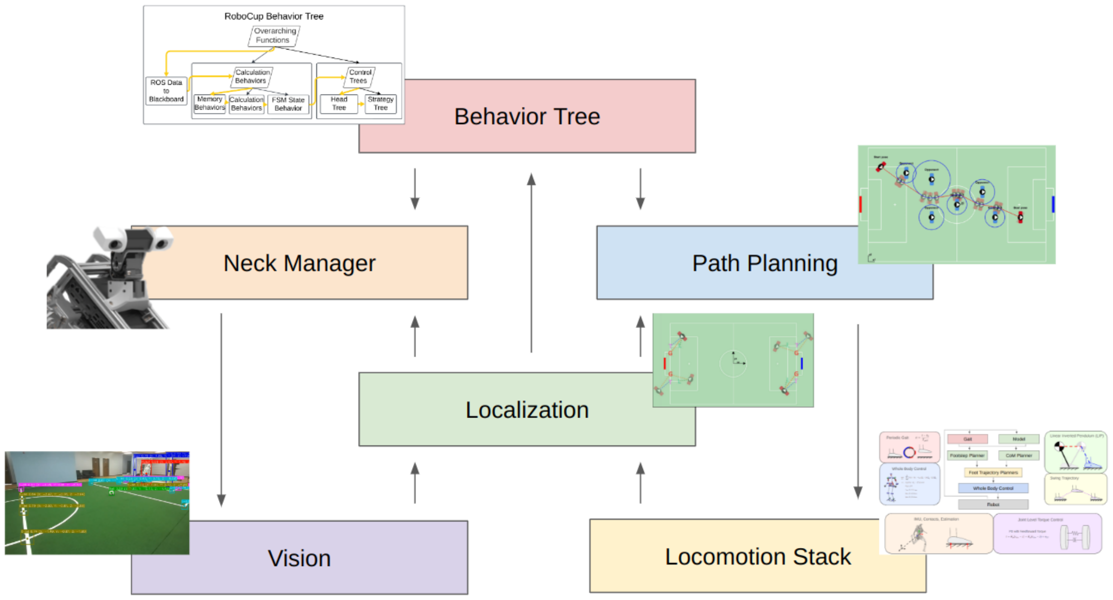


Figure 4.2: An automous soccer stack to compete in Robocup. The stack is fully capable of playing an autonomous game of soccer with little intervention from humans

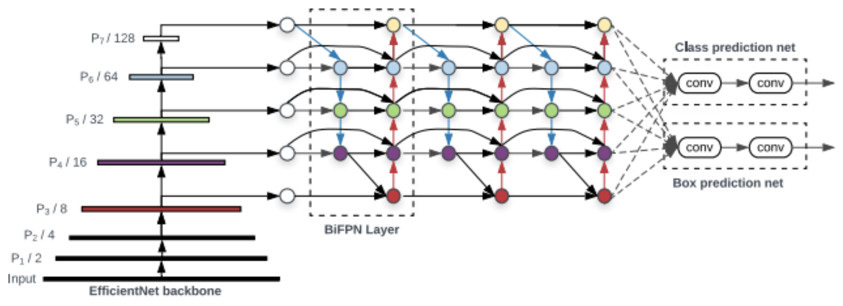
Robocup is an annual competition wherein teams of autonomous robots must play a full game of soccer on simulated turf without human intervention. From detecting the ball to deciding where to move next, the robots act on their own with the only allowed communication to an electronic scoreboard. Humans standby only in case of failure and to prevent damage to the robot.

This section outlines the software stack that the Robotics and Mechanisms Laboratory (RoMeLa) participated with during Robocup 2024 held in Eindhoven. As can be seen from Figure 4.2, a similar hierarchical controller approach was used wherein higher level controllers pass task objectives to lower level controllers to accomplish.

At the bottom of the stack, the vision algorithm is responsible for detecting key landmarks including the ball in order to inform any upstream controller decisions. This information is passed to the localization algorithm which informs the rest of the controllers about the state of play. In order to better inform the vision and localization algorithms, a neck manager alternates between focusing on the ball and scanning the field based on the current desired actions. A high level behavior tree then determines the appropriate soccer actions to execute from moving to the ball, kicking, and even stopping an opponent from scoring. From those soccer decisions, a path planner always tries to reach the specified goal position by developing x, y, and yaw velocity commands assuming motion on a plane. Finally, the commands are sent to the locomotion stack outlined in 2.

4.2 Vision

(a)



(b)

Figure 4.3: (a) The Stereolabs Zed 2i RGB-D stereo camera was used on Artemis with (b) a custom trained detection model based on Yolov8

Camera frames containing red, green, blue, and depth (RGB-D) signals were sampled at 60 Hz from the Stereolabs Zed 2i stereo camera as shown in Figure 4.3a using the manufacturer supplied Zed SDK [60]. The RGB portion of the images are then fed through a perception model based on the Yolov8 architecture [61] similar to Figure 4.3b trained on a custom dataset gathered over practice sessions and the actual competition. The one shot detection algorithm outputs detections for the ball, field markers, and other robots in pixel coordinates with depth measurements relative to the camera frame. These raw measure-

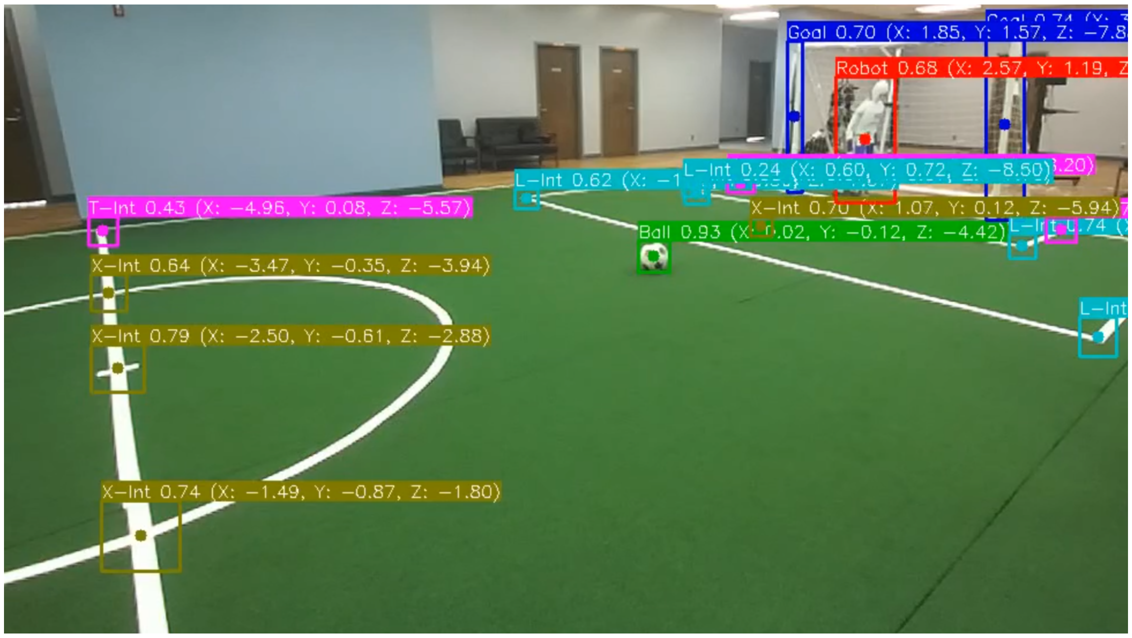

Goal 0.70 (X: 1.85, Y: 1.57, Z: −7.8
Robot 0.68 (X: 2.57, Y: 1.19, Z
L−Int 0.62 (X: −1
L−Int 0.24 (X: 0.60, Y: 0.72, Z: −8.50)
T−Int 0.43 (X: −4.96, Y: 0.08, Z: −5.57)
X−Int 0.70 (X: 1.07, Y: 0.12, Z: −5.94)
Ball 0.93 (X: .02, Y: −0.12, Z: −4.42)
X−Int 0.64 (X: −3.47, Y: −0.35, Z: −3.94)
L−Int
X−Int 0.79 (X: −2.50, Y: −0.61, Z: −2.88)
X−Int 0.74 (X: −1.49, Y: −0.87, Z: −1.80)

Figure 4.4: Robocup vision detecting landmarks, the ball, and other robots

ments are then converted into robot frame measurements based on the camera mounting geometry and the current neck joint angles. In order to improve the robustness of the detection algorithm, filters to check whether the ball was on green turf or within the bounds of play were performed to reduce the number of false positives. These filtered landmark data are then passed to the localization algorithm for determining the world frame game state.

An example of the output of the detection model can be seen in Figure 4.4. Each color represents a unique type of landmark with multiple instances of each type being detected at the same time. Each detection contains an (X,Y) pixel location as well as a Z depth measurement from the front of the camera face plane.

4.3 Localization

In order to effectively determine the state of the game, the localization algorithm relies on a multi-step approach outlined in Figure 4.5. First, a geometric based algorithm triangulates

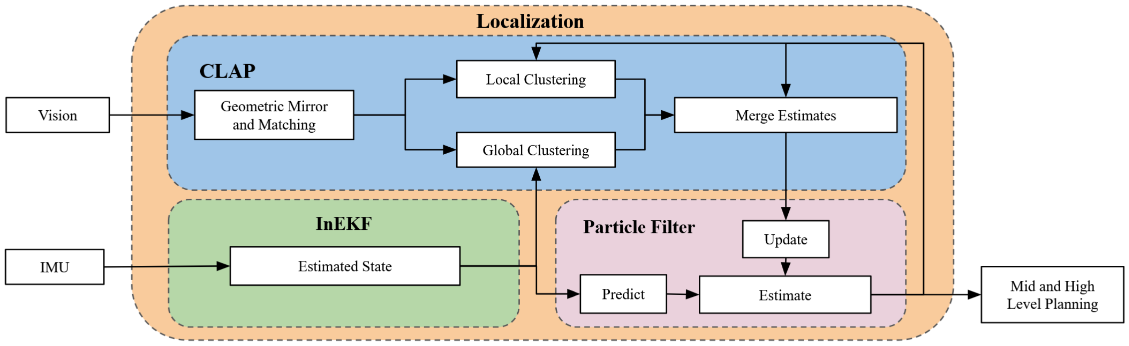


Figure 4.5: Localization used to determine where the robot is on the field

the closest best estimate of the robot in the world frame based on landmark data from vision and the predicted set of landmark positions given the current state estimate. As seen from Figure 4.6, depending on the view of the camera, the number and types of landmarks currently seen by the robot can vary greatly. Since there are no unique landmarks bordering the field, multiple spots on the field can yield the same measuremenmts. This results in a mirroring effect where any triangulation attempt has many solutions. Furthermore, landmarks measurements tend to be noisy or even occluded by obstacles such as other robots and humans. A sample of landmark data was measured to be off by as much as 0.3 meters when measured 1 meter away from the robot, with further measuremeents experiencing even larger noise. As a result, measurements that are either too far away from the robot or are estimated to be outside of the field of play are thrown out as outliers.

To combat this, for each pair of landmarks detected, an estimate of the true location of the robot on the field is found. Then, a clustering algorithm determines the most likely state of the robot from the collection of these estimates. For robustness and to prevent converging to a mirrored location, a global clustering algorithm is run at a slower rate on the past inertial data from the IMU to ensure that the estimate remains consistent. If these estimates differ too greatly, it signals that the geometric approach has converged on a location different to the actual state and must be reset. The estimate is then run through a particle filter with the IMU estimate in order to smooth out any discontinuities in the reported state.

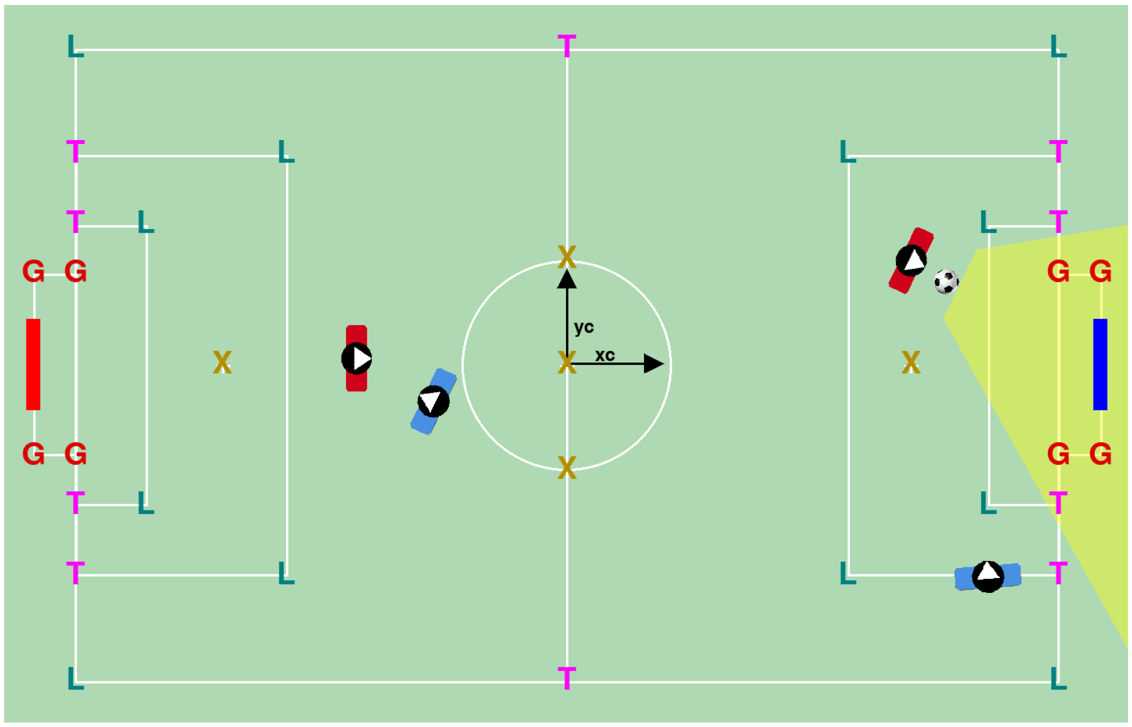


Figure 4.6: All possible landmarks on the field that localization needs to match to. The yellow region shows the field of the view of the robot and the predicted landmarks that should be within the camera frame

A test of the localization algorithm can be seen in Figure 4.7 where the black line is the solely the IMU estimate and the proposed algorithm is labeled in blue. The test ran a single continuous rectangular path from one goal box to across the field at the other goal box and back over multiple trips. The IMU estimate predictably experiences drift and a rotated shift in the estimated path due to the unobservability of the robot yaw. Meanwhile, the proposed algorithm, while slightly noisy, converges around the true rectangular path. This ensures that higher level decision making and path planning can reliably rely on an accurate game state. Further details can be found in the paper dedicated to localization [62].

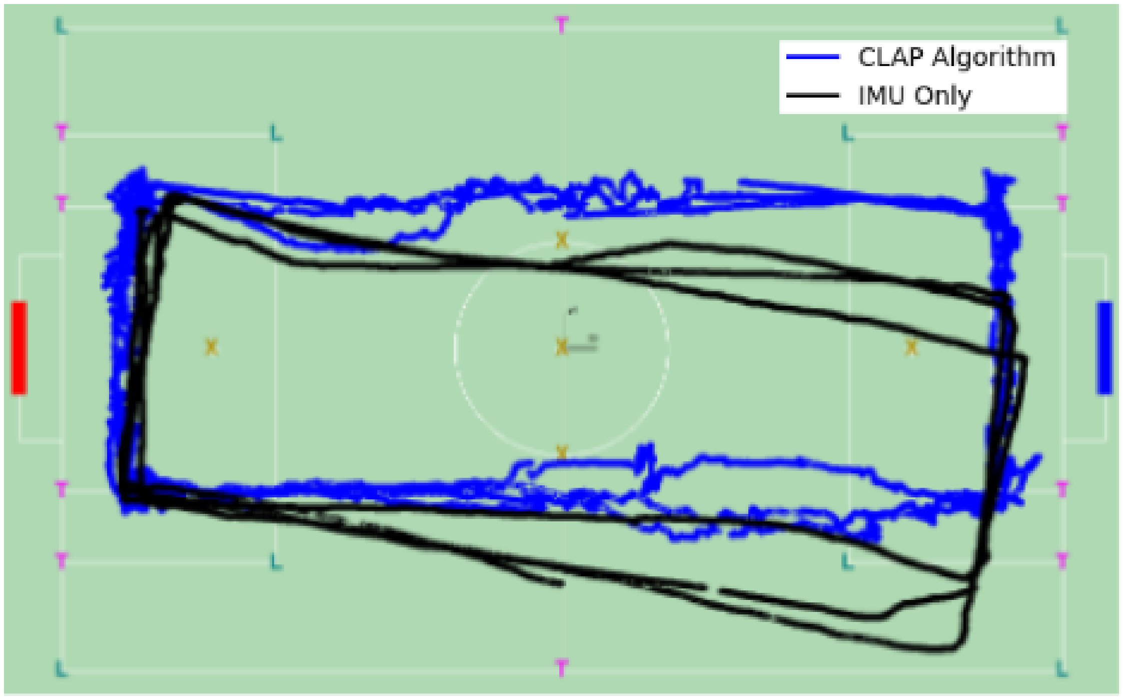


Figure 4.7: The robot was run from one goal box to the other in a rectangle path to test the accuracy and robustness of the algorithm.

4.4 Behavior Tree

From the current game state, a behavior tree was implemented using PyTrees [63] to make key soccer decisions such where to go next, when to kick the ball, and how to respond to the electronic scoreboard, the GameController [64]. Behavior trees are powerful decision making tools that have been implemented to handle decision making including play soccer [65] [66] [67] [68]. At first, both robots are assigned either the position of the goalie or the striker to simply behavior. The goalie is mainly configured to prevent opponents from scoring by defending a half and waiting for the ball to enter a specified scoring area. Once in that area, the goalie pursues the ball. If the opponent is near the ball, the goalie will try to as quickly as possible, kick it away from its own goal as there is high likelihood of the opponent scoring. Meawhile, if an opponent is further away, the goalie can take its time to kick the ball down field to an awaiting striker to score. Once the ball has been cleared, the goalie will return to

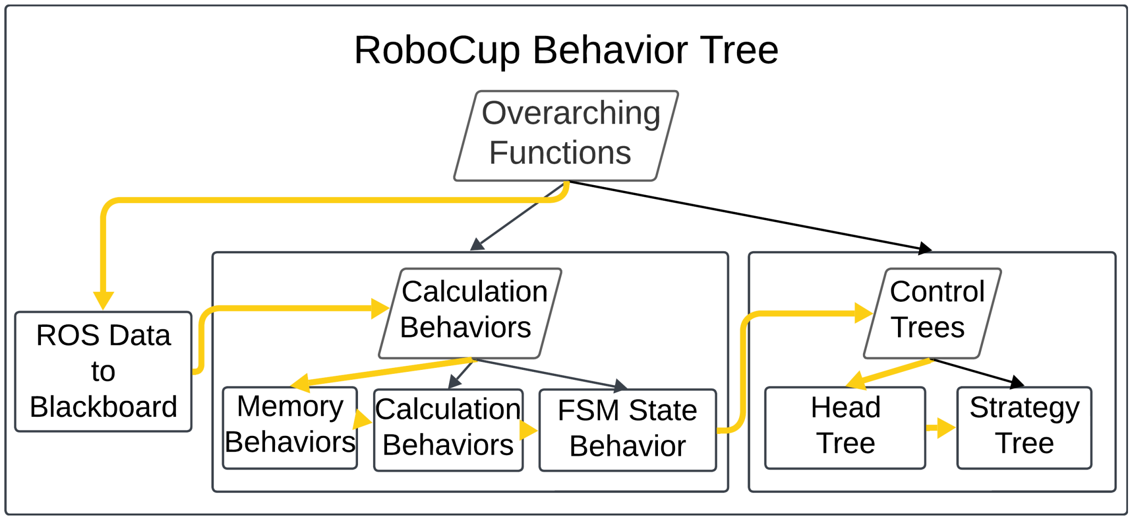


Figure 4.8: Behavior tree used to determine soccer decisions during a game

its defending spot.

Meanwhile, the striker's only objective is to score the ball. Once the ball has entered a specified area of the field, the striker will pursue the ball with desired end orientation facing the goal. If an opponent is deemed to be blocking the kick path of the ball, then the striker will try to kick around the opponent in order to set up for a clearer, higher probability shot on goal. As best as possible, the striker tries to push the ball down the field toward the opponents' goal to eventually score.

Outside of the general soccer playing behavior, behavior for game stoppages must be also put in place for when the ball goes out of play or when a goal is scored and a kickoff is required. These events are signaled from the GameController and are simply handled by prescribing desired goal positions to wait for play to resume. For example, if a corner kick is to be performed, the goalie will try to line up facing the appropriate corner on the goal box line. Similarly, the striker will choose a position closer to the defending goal, but near the corner to act as a first line of defense when play resumes. After which, the path planner and game playing behavior will take over on continuation of play. A paper with the specifics of the behavior tree is currently being produced.

4.5 Neck Manager

A key subsection of the behavior tree to point out is the need to tackle where to face the camera during the game of soccer. A naive approach would be to always either search for the ball or if the ball is found, track the ball. This leads to a player whose head is mostly aimed toward the floor. Since it is not guaranteed that suitable landmarks can be found near the ball, the localization of the game state suffers greatly as landmarks are key to understanding the state of the robot on the field.

In order to fix this, the robot was tasked to look up and scan the field at intervals in order to capture the required landmark data before resuming tracking the ball. This allows the localization algorithm to continuously maintain an accurate estimate while still satisfying the need to locate the ball. The ball only becomes a priority when the robot needs to get close in an effort to kick. At this point, the scanning behavior is disabled until the ball is either deemed far enough away or is lost by vision.

4.6 Path Planning

After receiving both end goal states from the behavior tree and the current game state from localziation, the path planner develops a path and velocity commands in order to avod obstacles on the way to the desired state. To speed up computation, a multistep approach, as shown in Figure 4.9, is used wherein a obstacle free, shortest path is first computed, smoothed using an interpolation function, and finally passed through an MPC to develop velocity commands.

Based on each of the opponent positions and the current robot state, a simple graph can be constructed using the vertices of polygons around each obstacle as shown in Figure 4.10. The objective is to find the shortest path that connects the start and end points using the vertices of the polygons. Since graph search algorithms such as Dijkstra's or A* are realtively

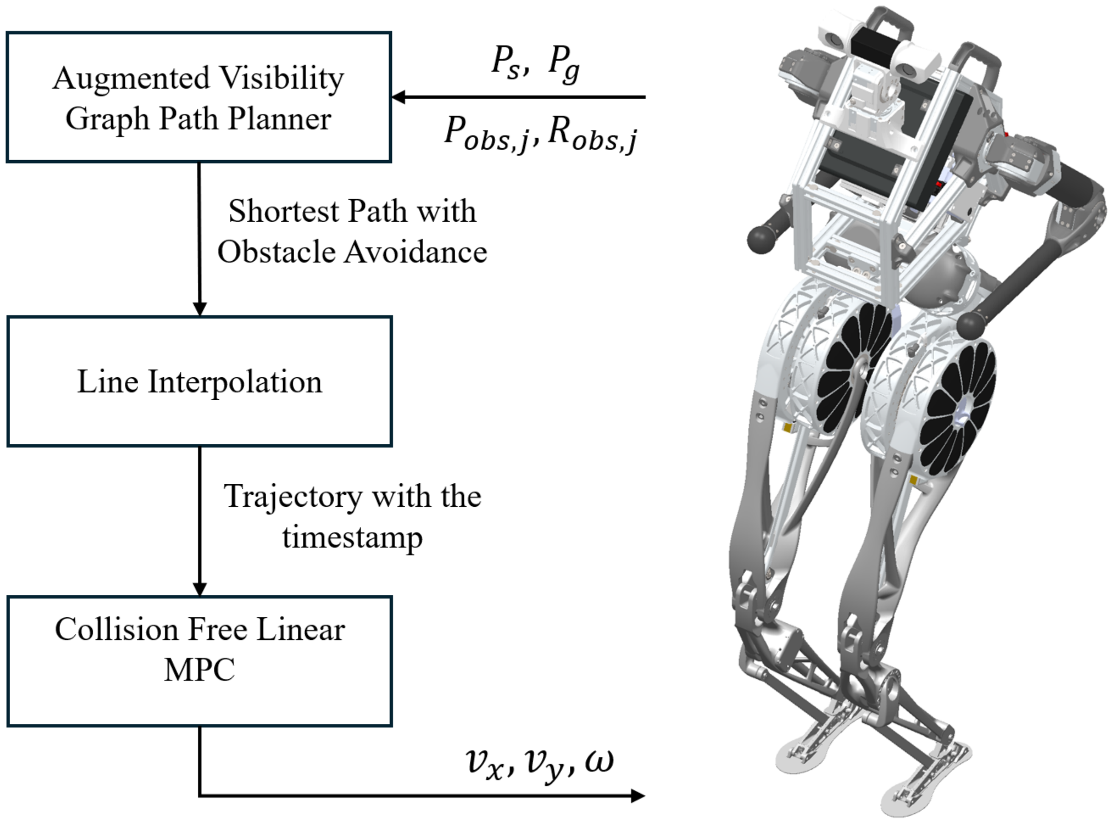

Augmented Visibility Graph Path Planner
$P_s,\ P_g$
$P_{obs,j}, R_{obs,j}$
Shortest Path with Obstacle Avoidance
Line Interpolation
Trajectory with the timestamp
Collision Free Linear MPC
v_x, v_y, ω

Figure 4.9: Midlevel ensuring obstacle avoidance on the way to the goal

mature and efficient [69], this allows for an obstacle free path to be developed at a relatively low computational cost.

After the segmented path has be found, it is interpolated using a continuous spline to allow for smooth resampling. The resampled path is then fed into a collision free nonlinear MPC that using a planar model [70] to ensure the developed path can be accurately tracked using velocity commands. Constraints are placed in the MPC in order limit linear velocity, turning speed, acceleration and to ensure that the robot remains in the field of play at all times.

While many steps were taken to prevent collision with obstacles in the formulation of the approach, due to delays and noise in the game state estimate, obstacles could actually be too close for the MPC respond quickly enough given the acceleration limits. As a result a further

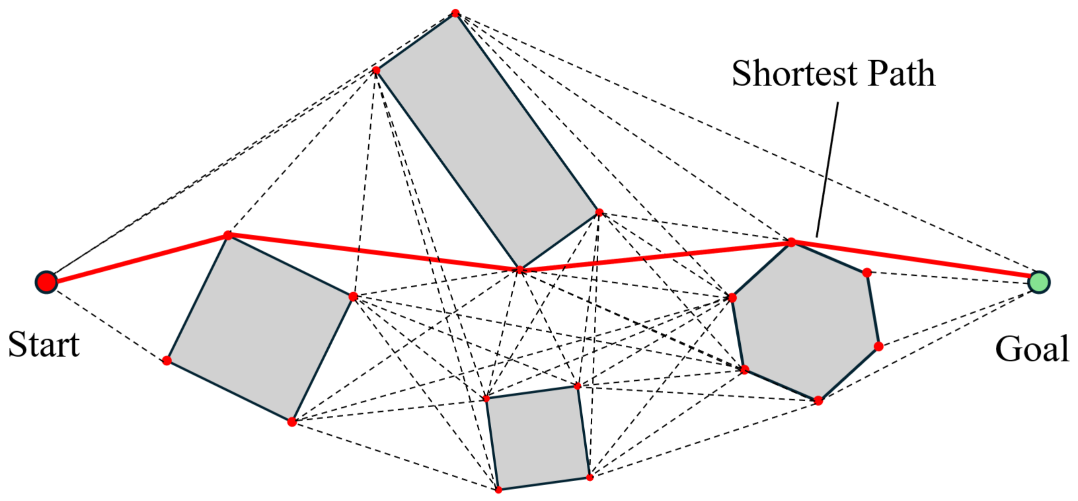


Figure 4.10: Polygons are centered around each obstacle with the goal of finding a the shortest path from start to finish by passing through the vertices

limiter is applied on the calculated velocities that limits the speed as inverse function relative to the distance to the nearest obstacle in the field of view of the robot. This speed limit varies, but is capped to a maximum of 0.2 m/s at the closest threshold in order to prevent the robot from unintentionally careening into an opponent and incurring a foul. Additional precautions then had to be taken in order to allow the robot to overtake an opponent by altering the path and avoid being completely stranded by the speed limiter.

An example of the resulting path and velocity trajectories from the approach can be seen in Figure 4.11. As can be seen, the path smoothly avoids all obstacles to reach the desired goal. This path can be applied in multiple situations whether they be to move toward the ball to kick, reposition to a more defensible position, or even take kickoff or corner kicks. Further details can be found in a detailed path planner paper [71].

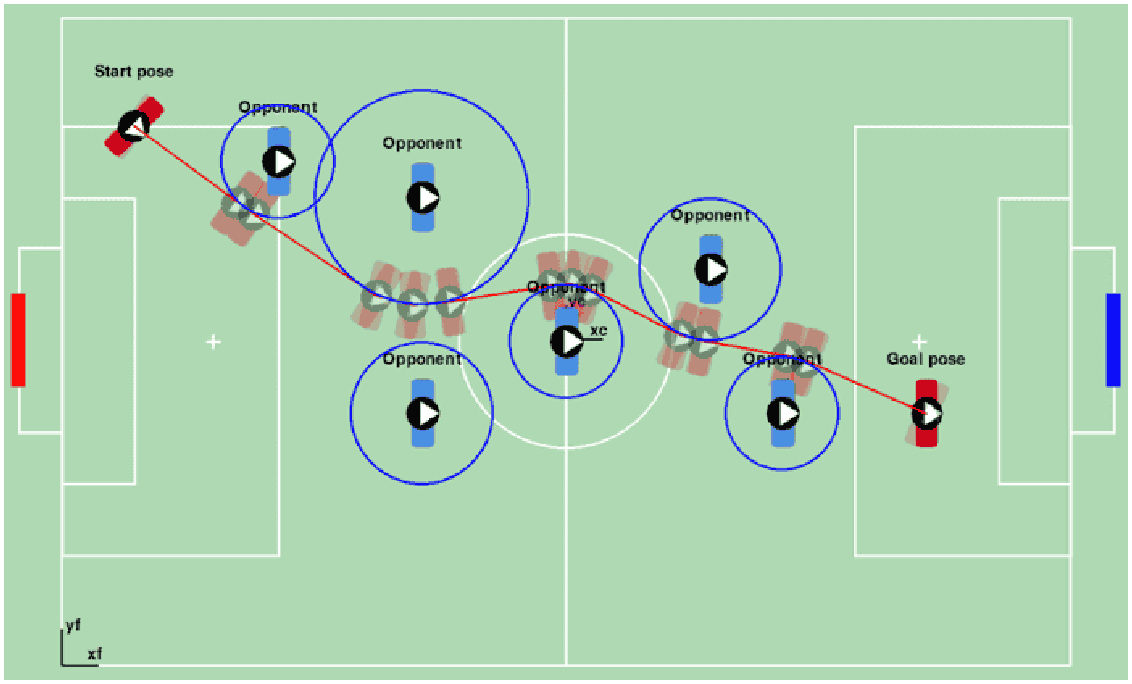
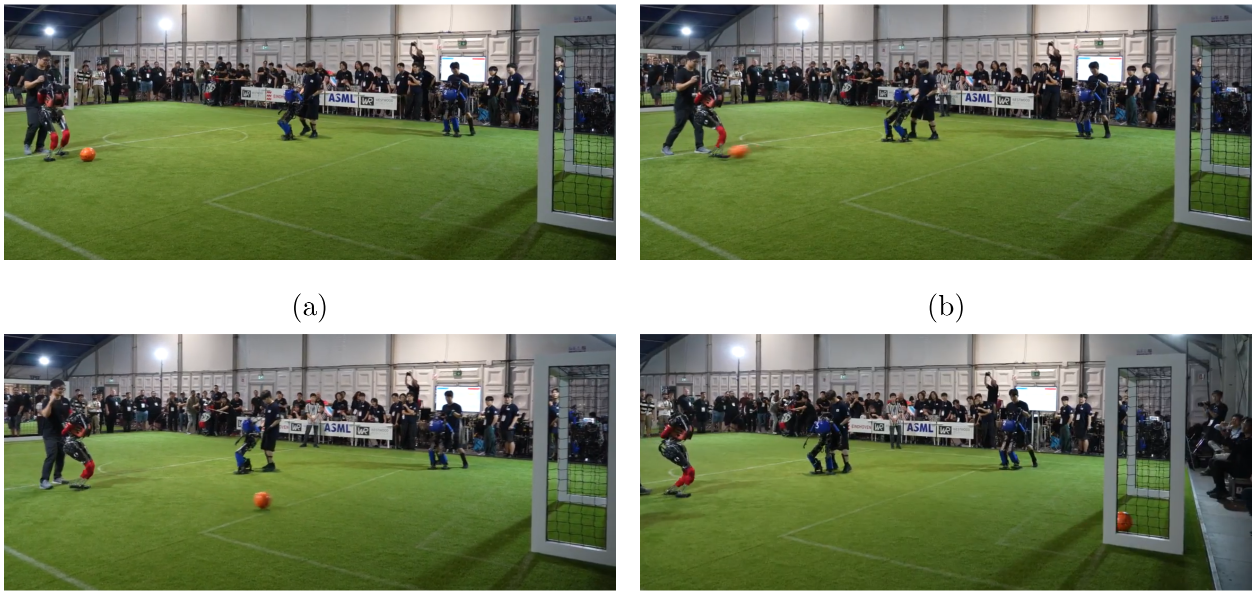


Figure 4.11: Midlevel ensuring obstacle avoidance on the way to the goal

4.7 Results

Using the stack outlined, Team RoMeLa at Robocup 2024 was able to win first place with a score of 45 goals over 6 matches without a single loss in competitive play. Artemis was by far the fastest robot in terms of linear speed, reaching up to 1.2 m/s during competitive play. Even when walking near opponents, Artemis was able to navigate safely to the ball while incurring minimal penalties.

There were quite a few problems that were required to be adjusted on the fly during the competition. To start, new training datasets had to adjust to harsh floodlights on the field and to capture soccer balls that were previously unlisted as potential ball candidates. The vision model, however, was robust enough to handle these new conditions with relatively few changes. Meanwhile, the field was constructed using wooden plywood sheets supported by wooden slats which caused a considerable amount of bounciness to the floor plane, especially in the spaces between supports. While other robots struggled to walk correctly with these



(a) (b) (c) (d)

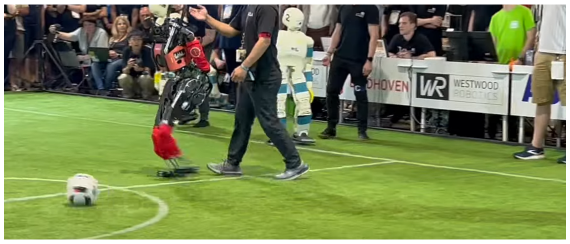
Figure 4.12: During the semi-final match, Artemis demonstrated an accurate long kick nearly half the field length. The shot progression of (a) lining up for the shot, (b) executing the kick, (c) ball travelling, and (d) a goal scored all happened in about 3 seconds

new floor conditions, Artemis was still able to maintain fast locomotion speeds across the field. Finally, multiple fields were set up side by side which drastically increased the number of incorrect landmarks being detected. For example, when looking towards the goal to score, the robot could see not only the correct goal and field markings, but at least two other goals and their entire field of landmarks. Due to the robustness considerations taken in the localization algorithm, however, Artemis rarely was confused during the game of play and managed to correcly position itself in the desired locations.

One of the major advantages was the strength of the kick which, as shown by Figure 4.12a-4.12d, was able to consistenly score from long distances. Artemis recorded the longest kick of the tournament from nearly the half field mark. This provided a considerable advantage relative to other teams who were limited in kicking strength.

One of the more exciting moments during the competition occurred when the goalie was able to pass ahead from one end of the field to an open area on the other end for the striker to successfully score a goal. This sequence happened relatively continuously and was one of the goals that most similarly reflected realistic play in a game of soccer. While this behavior wasn't explicitly programmed in for the two robots to cooperate, it demonstrates the potential capability to run in game set plays in order to take advantage of weaknesses in the defensive coverage.

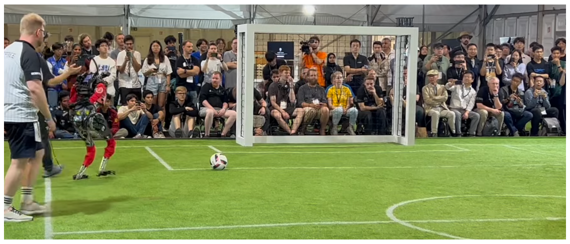
(a)



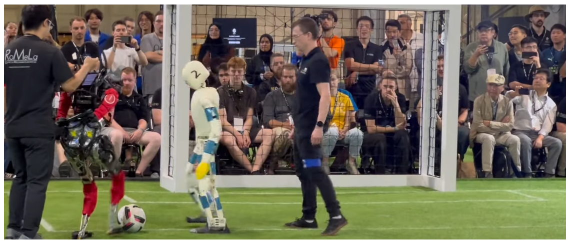
(b)

(c)

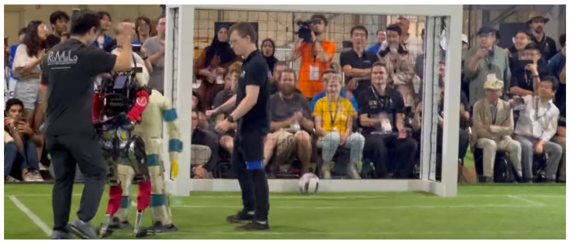
(d)



(e)



(f)



(g)

Figure 4.13: The two Artemis platforms on the field at the same time were able to perform a coordinated attack by having the goalie pass ahead to an open area on the field and have the striker pursue the ball for a goal. The play progression was (a) goalie passes the ball, (b) ball is en route and the striker is headed toward the predicted direction, (c) (d) (e), the striker sees the ball and plans a path to attack the goal, (f) the striker arrives and kicks the ball, and (g) a goal is scored

CHAPTER 5

Conclusion and Future Works

Humanoid locomotion and soccer behavior has advanced far past the days of old. With the introduction of the work presented in this thesis, a more dynamic platform has acheived competitive play at a level previously unseen. Many researchers from Robocup 2024 voiced that this past year's competition drew far more crowds and excitement than any of the previous years. In large part, by increasing the speed of play from platform capability to the execution time of the stack to make decisions, robot soccer has become far more alluring for the innocent bystander. This, however, it just the start of what can truly be acheived at this level.

Despite these advancements, robots are still a long way off from being able to compete with humans on an even playing field. Toward the end of the Robocup 2024 competition, a friendly match was played between a team of various robots and a team of the conference organizers as shown in Figure **??**. The robots still lacked the same level of locomotion speed, ball handling dexterity, quickness of decision making, and prediction of events that were more likely to happen next. As a result, the robots often seemed like they were lagging behind what the human players were doing, always moving to spots of where the ball had been as opposed to trying to meet the ball where it was going to end up. A large portion of this has to do with the fact that ball prediction and future state of play has not been implemented at the AdultSize humanoid level as of yet. Other leagues are have tried to adopt strategies to do this [72], but it is still an area of development untouched at the full humanoid size level.

Figure 5.1: Toward the end of the competition, robots faced off against humans in a match. The humans were still far ahead of the robots in terms of speed, dexterity, and decision making

Arguably, the most exciting part of the past competitions has been the increased speed of play whether due to faster locomotion, quicker reactions, or stronger kicks. All these areas are ripe for improvements. On the side of locomotion, the work presented here, while stable and fast, could be greatly improved through multiple methods. Firstly, gait modifications to include more complex adjustments in timing would offer the most initial improvement. Currently, with the fixed swing leg timings, the only way for the robot to stabilize is by placing the feet in accordance to the LIP model. While this works, it constrains how much disturbance rejection the robot can handle. Meanwhile, if a variable timing scheme were implemented [73], the robot would be able to walk more stably at faster speeds.

Currently, the Artemis platform speed is limited due to kinematics. Since the robot always tries to take off and land with feet parallel to the ground, this causes Artemis to hit joint limits at the knee or at the hip depending on the height of the platform and the desired stride length to reach a desired velocity. A sample of the robot nearing its kinematics during

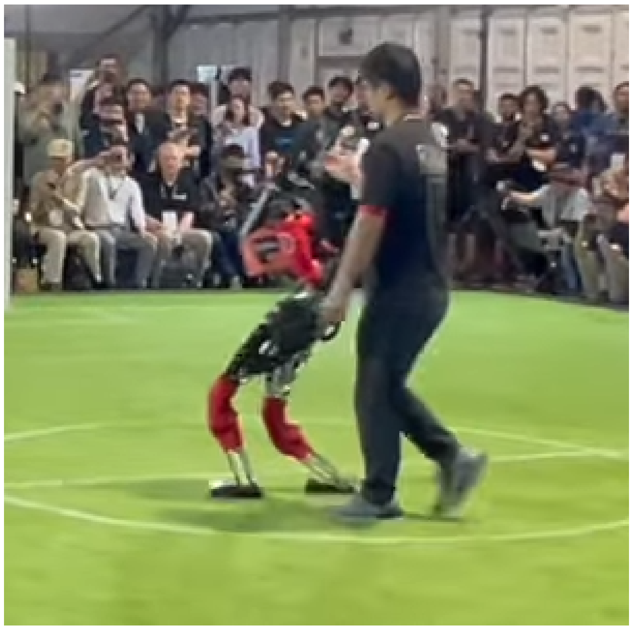


Figure 5.2: During longer strides, Artemis become kinematically limited. To reach faster speeds, either shorter timings, non parallel foot landing sequences, or running must be performed.

walking can be seen in Figure 5.2. A naive approach was to have the gait frequency be faster to decrease the stride length for a given velocity. This, however, works up to a limit as the natural pendulum of the system becomes disrupted. The body height can also be lowered to give a larger kinematic workspace for the legs to operate, but is not a viable long term solution as the body height affects all other areas of the stack. Consequently, attempts should be made to either include constructing liftoff and touchdown trajectories that allow for toe-heel transitions or to have a more stable form of running in which the robot no longer has to maintain single support at all times. Through running, the support leg no longer has to maintain an kinematically unfavorably positions.

Another area of improvement would to use the CoM in a more complex way than simply following the LIP model. This would require a full rework of the CoM and footstep planners, however, multiple approaches have proved successful on other robots. Divergent Component of Motion (DCM) has been a popularized method to control only the asymptotically unstable portion of the dynamics [74]. Using the centroidal momentum matrix [75] has also been a key aspect in humanoid control and could lead to some benefit particular when airborne [22]. Futhermore, using an Model Predictive Controller with particular attention to propagating known forces and moments of limbs could help stabilize such a large disturbance that is due to moving their own limb [76]. Even though including more complex models might reduce the solve frequency, it could prove valuable in helping to produce more expressive behavior that soccer requires.

On the autonomous soccer side of things, the advancement of gameplay should be focused on increasing the speed of play. This can be done with the addition of more teamplay. This year's stack utilized a set play which involved both robots and coordinating a single pass at kick off to gain an advantage. Most of the time though, both robots are confined to their own portions of the field in order to prevent robots from interfering or even damaging a teammate. This is the first step that should be take in order to have faster games. Teammate recognition and prediction of where motion will occur is a must. This can be expanded even further into an adversarial context in which prediction can then be used on opponent players. This can help players make more informed long term decisions particularly when weighing to go on the attack or to sit back and defend. More complex coordination behavior will be possible once this is in place. Instead, on every kickoff, teams should ideally recognize when there is only one opponent robot on the field or if there is an open goal with no opponents. In either of those cases, it is beneficial to leave the goal and coordinate on the attack as the likelihood far outweighs the risk that they cannot be beat back on the defense.

A stronger step in kick would make the largest gains in terms of gameplay. Right now, the speed a robot moves almost solely determines the winrate of the robot. Faster robots almost exclusively have an advantage because they are first to arrive and can pratically run circles around opponents. This will only violated if a strong kick capable of scoring at any point on the field is implemented. Then, the faster robots must weigh longer term decisions as even if they are first to move, they could be moving away from a critical defensive position. Thus, a dynamic in step kick would make the most sense for improving the game. The control strategy is something like what is listed above as an improvement and will most likley stem from predicting the forces and moments imparting not only of limbs moving at high speeds, but on predicting the contact forces relative to impact with the ball on kick. Because of the small inertias at the extremities, the ball had a very large part in terms of walking stability after a kick was made. To this end, an estimator of how much reaction force contact with the ball makes will prove vital towards getting this idea off the ground.

Meanwhile, the proximity of slowing down to avoid collisions with other robots on the field, while rudimentary worked very well. Unfortunately, there are still collisions particularly with the arms and hands of other robost. An arm stuck between the legs of another robot will constitute a foul against the robot with the legs. In order to avoid this, a collision avoidance planner needs to take into account more than simply that the robot has a convex cylinder to avoid, but further on a body postural command in order to avoid specific limbs of opponents. This can be coupled with the more complex locomotion planning models mentioned before.

Another large improvement that might not be very hard to implement would be to either decrease the time to boot during the competition or figure a way to have automated stand up without the handler. By far, the most costly penalties throughout the competition was the use of picking up the robot. On each time, the robot not only had to dragged off the field, but required a 30 second penalty under which it was not allowed to enter. While one robot down may be tough, both robots completely sidelined for 30 seconds is nearly disasterous. To mitigate this, the boot sequence should be decreased as it currently takes about a full minute from pick up to get an Artemis back on the field, inclusive of the 30 second penalty. Meanwhile, if it can be avoided entirely by having the robot stand up on its own, that would be the most optimal outcome.

REFERENCES

[1] V. 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*, vol. 63, no. 4, pp. 413–419, 1975.

[2] P. Serafini, E. Guazzelli, B. Schrefler, F. Pfeiffer, F. G. Rammerstorfer, I. Kato, S. Ohteru, H. Kobayashi, K. Shirai, and A. Uchiyama, "Information-power machine with senses and limbs: Wabot 1," *On Theory and Practice of Robots and Manipulators: Volume I*, pp. 11–24, 1974.

[3] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: System overview and integration," in *IEEE/RSJ international conference on intelligent robots and systems*, vol. 3. IEEE, 2002, pp. 2478–2483.

[4] K. Kaneko, F. Kanehiro, M. Morisawa, K. Akachi, G. Miyamori, A. Hayashi, and N. Kanehira, "Humanoid robot hrp-4-humanoid robotics platform with lightweight and slim body," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 4400–4407.

[5] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, pp. 429–455, 2016.

[6] E. Guizzo, "By leaps and bounds: An exclusive look at how boston dynamics is redefining robot agility," *IEEE Spectrum*, vol. 56, no. 12, pp. 34–39, 2019.

[7] Tesla, https://www.tesla.com/AI, 2021, [Accessed 27-10-2024].

[8] Unitree, https://www.unitree.com/h1, 2023, [Accessed 27-10-2024].

[9] Apptronik, https://apptronik.com/apollo, 2023, [Accessed 27-10-2024].

[10] F. AI, https://www.figure.ai/, 2024, [Accessed 27-10-2024].

[11] T. Apgar, P. Clary, K. Green, A. Fern, and J. W. Hurst, "Fast online trajectory optimization for the bipedal robot cassie." in *Robotics: Science and Systems*, vol. 101. Pittsburgh, Pennsylvania, USA, 2018, p. 14.

[12] D. Lahr, V. Orekhov, B. Lee, and D. Hong, "Early developments of a parallelly actuated humanoid, saffir," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 55942. American Society of Mechanical Engineers, 2013, p. V06BT07A054.

[13] "Meet digit." [Online]. Available: https://robots.ieee.org/robots/digit/

[14] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2016, pp. 38–44.

[15] C. Semini, J. Goldsmith, D. Manfredi, F. Calignano, E. P. Ambrosio, J. Pakkanen, and D. G. Caldwell, "Additive manufacturing for agile legged robots with hydraulic actuation," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 123–129.

[16] P. M. Wensing, A. Wang, S. Seok, D. Otten, J. Lang, and S. Kim, "Proprioceptive actuator design in the mit cheetah: Impact mitigation and high-bandwidth physical interaction for dynamic legged robots," *Ieee transactions on robotics*, vol. 33, no. 3, pp. 509–522, 2017.

[17] 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*. IEEE, 2012, pp. 1970–1975.

[18] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. 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, pp. 2245–2252.

[19] S. Rezazadeh, C. Hubicki, M. Jones, A. Peekema, J. Van Why, A. Abate, and J. 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*, vol. 57243. American Society of Mechanical Engineers, 2015, p. V001T04A003.

[20] W. Bosworth, S. Kim, and N. Hogan, "The mit super mini cheetah: A small, low-cost quadrupedal robot for dynamic locomotion," in *2015 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2015, pp. 1–8.

[21] Y. Liu, J. Shen, J. Zhang, X. Zhang, T. Zhu, and D. Hong, "Design and control of a miniature bipedal robot with proprioceptive actuation for dynamic behaviors," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8547–8553.

[22] M. Chignoli, D. Kim, E. Stanger-Jones, and S. Kim, "The mit humanoid robot: Design, motion planning, and control for acrobatic behaviors," in *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2021, pp. 1–8.

[23] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[24] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[25] G. A. Castillo, B. Weng, W. Zhang, and A. Hereid, "Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5136–5143.

[26] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," *arXiv preprint arXiv:2105.08328*, 2021.

[27] R. J. Full and D. E. Koditschek, "Templates and anchors: neuromechanical hypotheses of legged locomotion on land," *Journal of experimental biology*, vol. 202, no. 23, pp. 3325–3332, 1999.

[28] G. Ficht, H. Farazi, D. Rodriguez, D. Pavlichenko, P. Allgeuer, A. Brandenburger, and S. Behnke, "Nimbro-op2x: affordable adult-sized 3d-printed open-source humanoid robot for research," *International Journal of Humanoid Robotics*, vol. 17, no. 05, p. 2050021, 2020.

[29] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *2019 American control conference (ACC)*. IEEE, 2019, pp. 4559–4566.

[30] Y. Gong and J. 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)*. IEEE, 2021, pp. 2832–2838.

[31] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, pp. 429–455, 2016.

[32] J. Li and Q. Nguyen, "Multi-contact mpc for dynamic loco-manipulation on humanoid robots," 2023.

[33] K. Yuan, N. Sajid, K. Friston, and Z. Li, "Hierarchical generative modelling for autonomous robots," *Nature Machine Intelligence*, vol. 5, no. 12, pp. 1402–1414, 2023.

[34] "Robocup 2024." [Online]. Available: https://2024.robocup.org/

[35] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[36] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4399–4406.

[37] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. 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)*, vol. 1. IEEE, 2001, pp. 239–246.

[38] Y. Liu, P. M. Wensing, D. E. Orin, and Y. 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)*. IEEE, 2015, pp. 5710–5717.

[39] E. Cuevas, D. Zaldivar, M. Perez, and M. Ramirez, "Polynomial trajectory algorithm for a biped robot," *arXiv preprint arXiv:1405.5937*, 2014.

[40] M. D. K. Breteler, R. G. Meulenbroek, and S. C. Gielen, "An evaluation of the minimum-jerk and minimum torque-change principles at the path, trajectory, and movement-cost levels," *Motor Control*, vol. 6, no. 1, pp. 69–83, 2002.

[41] L. Sentis and O. 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.* IEEE, 2006, pp. 2641–2648.

[42] J. Ahn, S. J. Jorgensen, S. H. Bang, and L. Sentis, "Versatile locomotion planning and control for humanoid robots," *Frontiers in Robotics and AI*, vol. 8, p. 712239, 2021.

[43] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. 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)*. IEEE, 2018, pp. 1–9.

[44] W.-L. Ma and A. D. Ames, "From bipedal walking to quadrupedal locomotion: Full-body dynamics decomposition for rapid gait generation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4491–4497.

[45] D. Kim, S. J. Jorgensen, J. Lee, J. Ahn, J. Luo, and L. Sentis, "Dynamic locomotion for passive-ankle biped robots and humanoids using whole-body locomotion control," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 936–956, 2020.

[46] J. Yamaguchi, E. Soga, S. Inoue, and A. Takanishi, "Development of a bipedal humanoid robot-control method of whole body cooperative dynamic biped walking," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 1. IEEE, 1999, pp. 368–374.

[47] R. Featherstone, *Rigid body dynamics algorithms.* Springer, 2014.

[48] K. Lynch, *Modern Robotics.* Cambridge University Press, 2017.

[49] C. W. Wampler, “Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, no. 1, pp. 93–101, 1986.

[50] A. S. Deo and I. D. Walker, “Overview of damped least-squares methods for inverse kinematics of robot manipulators,” *Journal of Intelligent and Robotic Systems*, vol. 14, pp. 43–68, 1995.

[51] T. Zhu, “Design of a highly dynamic humanoid robot,” Ph.D. dissertation, UCLA, 2023.

[52] Boost. (2024) Boost c++ libraries. [Online]. Available: https://www.boost.org/

[53] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abm6074

[54] W. Robotics. (2024) Westwood robotics sdk. [Online]. Available: https://www.westwoodrobotics.io/

[55] S. J. Jorgensen, “Towards deploying legged humanoids in human environments,” Ph.D. dissertation, UT Austin, 2020.

[56] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, “The pinocchio c++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives,” in *2019 IEEE/SICE International Symposium on System Integration (SII)*, 2019, pp. 614–619.

[57] D. E. Orin and A. Goswami, “Centroidal momentum matrix of a humanoid robot: Structure and properties,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, 2008, pp. 653–659.

[58] A. Bambade, F. Schramm, S. El Kazdadi, S. Caron, A. Taylor, and J. Carpentier, “Proxqp: an efficient and versatile quadratic programming solver for real-time robotics applications and beyond,” 2023.

[59] H. Herr and M. Popovic, “Angular momentum in human walking,” *Journal of experimental biology*, vol. 211, no. 4, pp. 467–481, 2008.

[60] Sterolabs. (2024) Zed camera sdk 4.2.2. [Online]. Available: https://www.stereolabs.com/en-in/developers/release

[61] R. Varghese and M. Sambath, "Yolov8: A novel object detection algorithm with enhanced performance and robustness," in *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*. IEEE, 2024, pp. 1–6.

[62] G. Fernandez, R. Hou, C. Togashi, A. Xu, and D. Hong, "Clap: Clustering to localize across n possibilities, a simple, robust geometric approach in the presence of symmetries," in *2025 IEEE International Conference on Robotics and Automation*. (Under Review).

[63] PyTrees. (2024) Pytrees 2.2.3. [Online]. Available: https://py-trees.readthedocs.io/en/devel/

[64] R. Federation. (2024) Gamecontroller. [Online]. Available: https://github.com/RoboCup-SPL/GameController

[65] M. Colledanchise, R. Parasuraman, and P. Ögren, "Learning of behavior trees for autonomous agents," *IEEE Transactions on Games*, vol. 11, no. 2, pp. 183–189, 2019.

[66] S. Gugliermo, D. Cáceres Domínguez, M. Iannotta, T. Stoyanov, and E. Schaffernicht, "Evaluating behavior trees," *Robotics and Autonomous Systems*, vol. 178, p. 104714, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889024000976

[67] M. Iovino, E. Scukins, J. Styrud, P. Ögren, and C. Smith, "A survey of behavior trees in robotics and ai," *Robotics and Autonomous Systems*, vol. 154, p. 104096, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889022000513

[68] R. H. Abiyev, N. Akkaya, and E. Aytac, "Control of soccer robots using behaviour trees," in *2013 9th Asian Control Conference (ASCC)*, 2013, pp. 1–6.

[69] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. [Online]. Available: https://doi.org/10.1109/tssc.1968.300136

[70] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," in *2007 American control conference*. IEEE, 2007, pp. 2296–2301.

[71] R. Hou, G. Fernandez, M. Zhu, and D. Hong, "Model predictive control with visibility graphs for humanoid path planning and tracking against adversarial opponents," in *2025 IEEE International Conference on Robotics and Automation*. (Under Review).

[72] A. F. Muzio, M. R. Maximo, and T. Yoneyama, "Deep reinforcement learning for humanoid robot behaviors," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 1, p. 12, 2022.

[73] E. Daneshmand, M. Khadiv, F. Grimminger, and L. Righetti, "Variable horizon mpc with swing foot dynamics for bipedal walking control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2349–2356, 2021.

[74] S. Caron, "Biped stabilization by linear feedback of the variable-height inverted pendulum model," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9782–9788.

[75] D. E. Orin and A. Goswami, "Centroidal momentum matrix of a humanoid robot: Structure and properties," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 653–659.

[76] J. Li, J. Ma, O. Kolt, M. Shah, and Q. Nguyen, "Dynamic loco-manipulation on hector: Humanoid for enhanced control and open-source research," 2023.