Developing Novel Hardware Platforms for Model-Predictive Control

Sofia Kwok

CMU-RI-TR-24-43

August 6

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Dr. Zachary Manchester
Dr. Aaron Johnson
Swaminathan Gurumurthy

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To the friends we made along the way

Abstract

Robot hardware has rapidly become more accessible in the last ten years. However, there is still a dearth of low-cost hardware platform that are open-source and easy to build. With recent developments in accessible manufacturing methods, such as FDM 3D printing, manufacturing and designing parts without using precision machining has become feasible. In this thesis, we address two gaps in open-source hardware platforms that can be filled using relatively few precision machined parts. Both designs are fully open-source, and the controllers for both systems are also available online.

The first hardware platform aims to create a lightweight bipedal system with pitch control, allowing it to recover from perturbations and perform dynamic motions. We do this by adding a reaction wheel actuation system that controls the pitch angle of the robot, fully actuating the system and enabling attitude stabilization. We linearize the dynamics of the system to obtain a linear discrete-time optimization problem that can be solved as a quadratic program. The linear MPC problem is tested on hardware at 240 Hz.

The second part of the thesis covers the design and control process for an aquatic robot that can be easily modeled in a fluid simulator, simplifying the sim-to-real transfer process. To precisely control the robot state, we use a five-bar linkage system. We implement and test this on hardware, and demonstrate that our system is able to accurately and repeatably navigate in a fluid environment.

Acknowledgments

First, I would like to thank my advisor, Dr. Zac Manchester. His support, guidance, and generosity have been instrumental in completing this degree. I am grateful for all of the knowledge he has shared with me over the past two years and for his enthusiasm for "doing cool stuff on hardware".

I would also like to thank the members of the Robotic Exploration Laboratory for their support. This thesis would not have been possible without John, Fausto, JJ, Swami, Arun, Ashley, Mitch, Jacob, and many others who have shared funny stories with me, listened to me complain, and helped me when I was struggling. I have become a better researcher and a better person thanks to you, and my time at Carnegie Mellon would not have been half as interesting or fun without our yap sessions. Special thanks to Sam for being a constant source of joy, my biggest cheerleader, and my best friend.

It would take far too many pages to mention everyone who has made my life fun in Pittsburgh. Thank you to everyone who has gotten dinner with me, bought spontaneous concert tickets, helped me move, looked over my emails, and watched the Great British Bake Off with me. I will miss you all a lot. Special thanks to Matt, Uksang, Conner, Nayana, Peter, Fausto, and John for being my little treat buddies when I needed a pick-me-up in the middle of the work day.

Finally, I would like to thank my family for their support, especially my parents. Their support and love has been critical for my journey these past two years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Currently in robotics research, there are several common hardware platforms for testing new control and learning algorithms. Most of these are commercially available robots, such as the Unitree Go1 quadruped, and have become increasingly affordable in the last decade. However, there is still a limited number of hardware platforms that are feasible for using in a laboratory setting. Many robots are expensive, bulky, and have inaccessible or incorrect documentation for their mass and inertial properties.

In this thesis, we propose two open-source hardware platforms that aim to fill these voids. The first hardware platform aims to solve the under-actuation problem for bipedal robots. The second hardware platform provides a low-cost platform for fluid simulation validation and novel fluid navigation control policies. The design, assembly, and controllers for both projects are covered in this thesis and are also available online. By broadening the scope of available hardware, we hope to make testing and validating new control policies more accessible for research labs, independent projects, and budget-constrained institutions.

1.1 Thesis Structure

This thesis is organized by hardware platform. Chapter 2 covers work on the biped. We address the under-actuation of bipedal systems and their lack of pitch control by integrating a reaction wheel actuation system onto an open-source, lightweight biped. The reaction wheel guarantees stabilization during flight and ground contact phases.

Chapter 3 covers the design and implementation of our novel aquatic robot. We introduce a low-complexity testing platform for validating simulation results and for testing novel control policies in fluids.

Chapter 2

Biped RWA

2.1 Introduction

The requirements for a research hardware platform becomes a problem when taken in conjunction with the downsides of commercially available bipedal systems. Bipeds such as Cassie are typically large, expensive, and difficult to fix after breaking, which is largely due to the large number of precision-machined or cast parts involved in manufacturing. Additionally, many of these bipeds are not open-source, leading to confusion over mass properties or low-level functionality. Using these bipeds in a research laboratory is therefore difficult, as every break or malfunction requires company consultation, and the lack of design transparency makes proper modeling near impossible. Large-scale bipedal systems also often require a crane and multiple humans to position and run code, making them difficult and potentially unsafe to operate in a laboratory setting. Having lightweight and inexpensive robotic systems is critical for testing new dynamic control policies, and makes testing much safer and faster [7].

The solution is to turn to open-source legged designs, which tend to be smaller, lighter, and cheaper. There are many open-source legged quadrupeds available, such as the Stanford-doggo or Squeaky [13][24], and many open-source controllers that have demonstrated impressive agility on hardware [4]. However, there are significantly fewer open-source bipedal designs that are maintained and well documented. Out of these, the legged robotic system from the Open Dynamic Robot Initiative (ODRI)

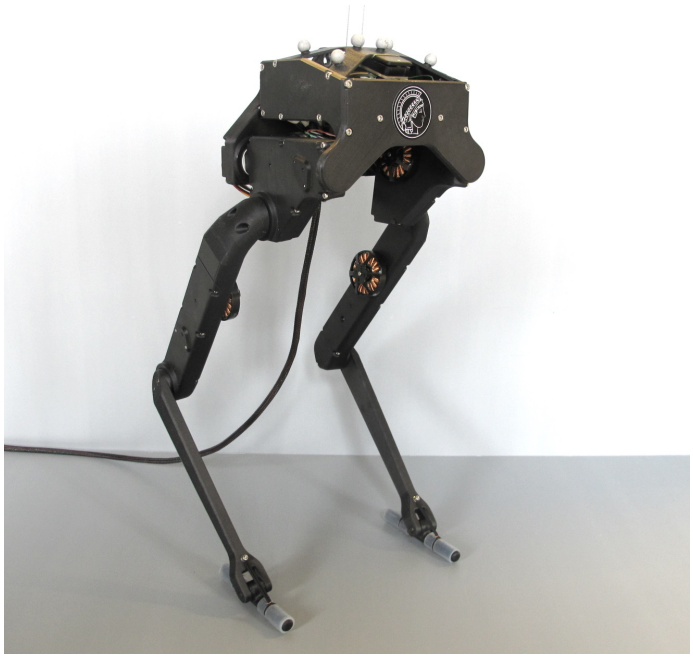

Figure 2.1: Bolt, a bipedal 6DOF robot from ODRI [11]

boasts a lightweight and low-complexity design, providing torque control with an open-source design [11]. It also has a small number of precision machined parts, making it feasible to quickly assemble and run. Many of the primary structural components are 3D printed, making fixing parts easy and low-cost.

However, bipeds without actuated ankles are inherently unstable. Bolt's legs each have 3 DOFs, along with a passive ankle joint. This gives yaw and roll control, but means that Bolt has no pitch control [8]. This limitation, which is common among bipedal robots with no torsos, means that bipeds without alternate ways of producing angular momentum cannot perform truly dynamic maneuvers, such as running or jumping [22]. We can take inspiration from nature to fix this problem. Legged animals stabilize themselves during dynamic movement using unconventional actuation strategies involving their other body parts. Cats and other quadrupeds use their tails, spines, and legs to adjust their attitude or provide a counterweight during dynamic motion [20][18]. Humans and other primarily bipedal primates use their arms to produce angular momentum that keeps them upright while running and jumping [21].

One of the simpler methods to control angular momentum is by using a reaction

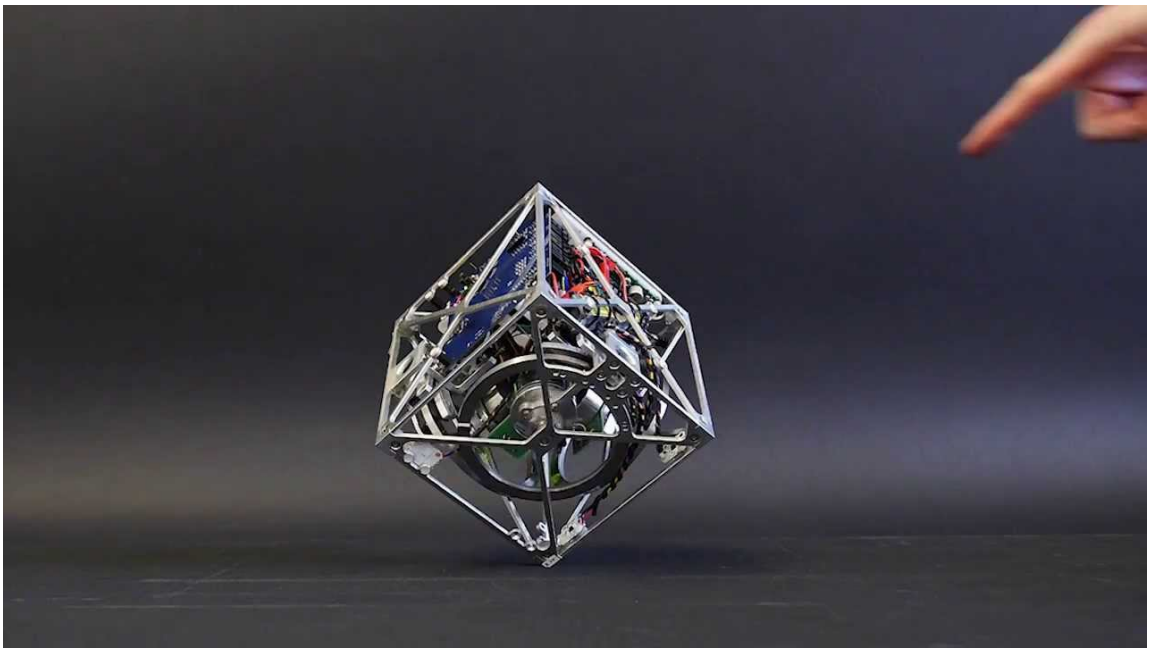

Figure 2.2: The Cubli, a RWA controlled robot [10]

wheel actuator (RWA), which is commonly used on spacecraft to perform attitude control [10]. RWAs also have simpler dynamics than tails, making them a good choice for applications that require rapid computation and linearized dynamics. We demonstrate that RWAs help with perturbation recovery, contribute to a stable gait pattern, and are effective at stabilizing a bipedal system during dynamic maneuvers.

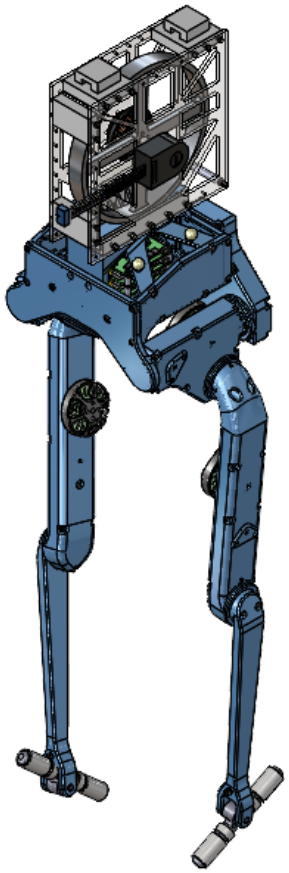

Figure 2.3: Bolt with RWA attached

We also adhere to the mission of the ODRI by publicly releasing the CAD and design specifications of the RWA. This system requires minimal precision machining and is low-cost, lightweight, and made to integrate smoothly with the existing biped. By creating a publicly available pitch control system for bipedal robots, we hope to make this open-source platform more accessible to research labs.

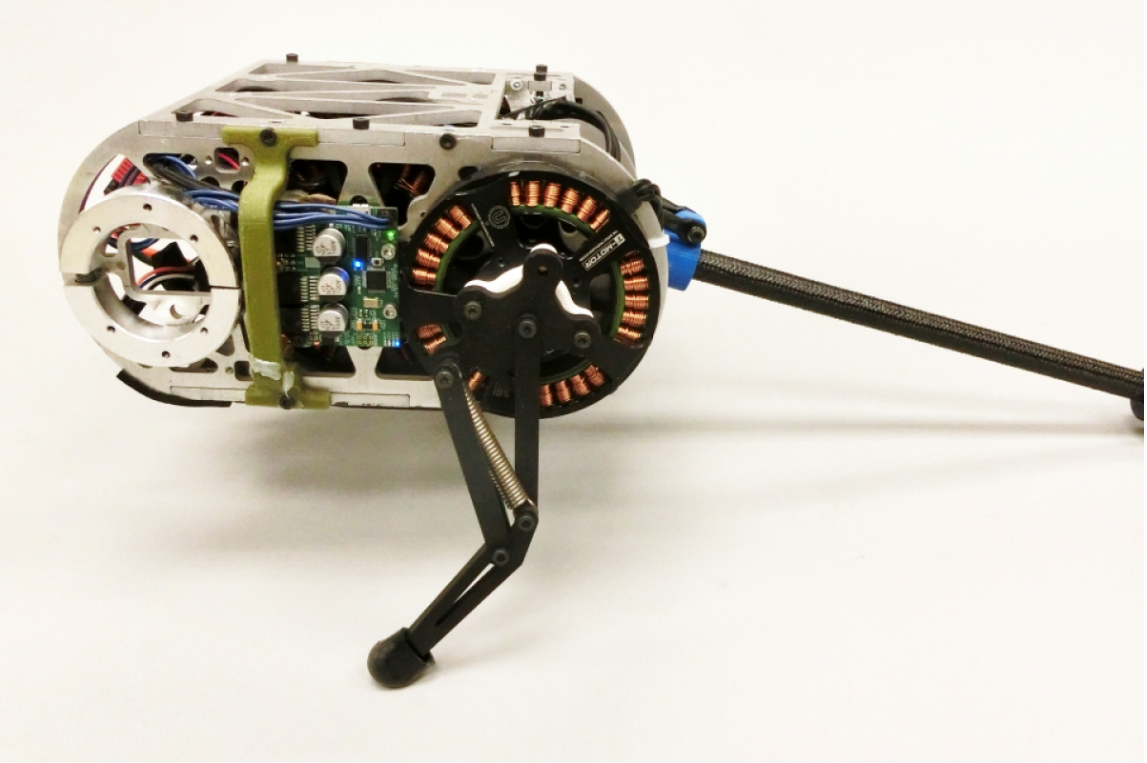

Figure 2.4: The Penn Jerboa, which features a bio-inspired tail design for a bipedal robot

2.2 Related Work

2.2.1 Tails

Tails, which can generate significant angular momentum and are relatively energy-efficient, have been used for attitude stabilization on bipeds and quadrupeds [12][17]. They have also been used for dynamic maneuvers on legged systems, making them an appealing option for stabilization [19]. However, several factors make them not feasible for this system. Using a bio-inspired stabilization method would require an excessive amount of additional hardware, which the lightweight design and low-torque motors of this biped are unable to support. Tails would introduce another layer of control complexity, since their inertial properties can change. Adding an additional limb requires collision checking between the linkage and body, and can introduce another contact mode that leads to unforeseen behaviors during testing.

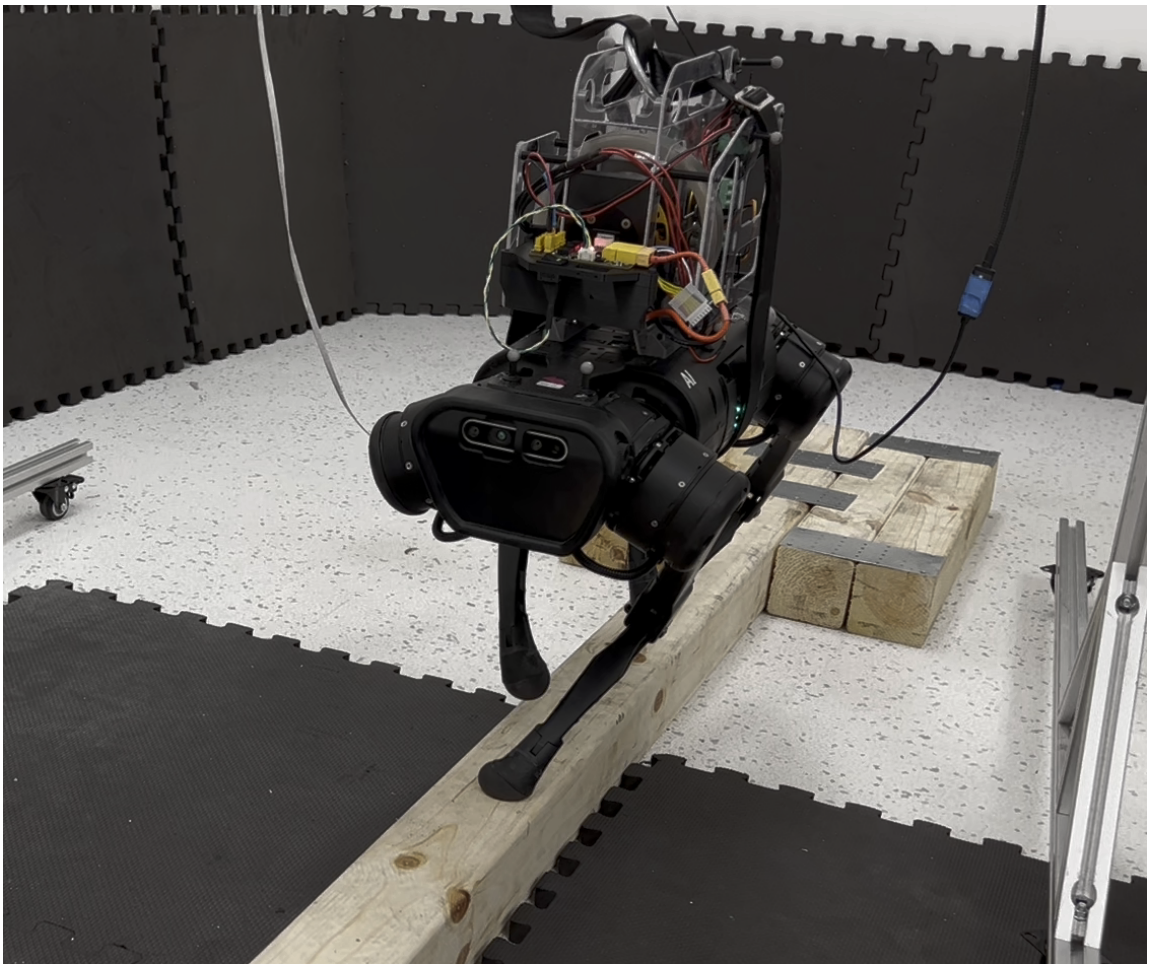

Figure 2.5: A quadruped with a RWA system navigating a balance beam [15]

2.2.2 RWAs

RWAs have previously been used successfully on legged systems for otherwise difficult or impossible locomotion tasks, such as balance beam walking [15]. They have also been used for attitude stabilization when robots lose rotational control during flight or when there are only two feet in contact with the ground [14]. RWAs have been previously used on bipeds to improve walking efficiency, and have been shown to successfully compensate for the added mass on the system [5][6]. However, RWAs have not yet been used on bipedal systems for attitude control during dynamic locomotion.

One downside of RWAs is their tendency to saturate rotor speed. This can be easily prevented by adding a linear feedback term that regulates the angular momentum.

Additionally, since many of the dynamic maneuvers are short-lived, making the pitch correction policy aggressive enough leads to quick acceleration and deceleration of the reaction wheel.

2.2.3 Dynamic Manuevers On Legged Robots

Highly dynamic behavior, such as running and jumping, has been demonstrated on quadrupeds, and on humanoid robots. However, demonstrations of similar behavior on bipeds without torsos has been limited to running and limited hopping. This is because hopping also requires precise knowledge of the takeoff and landing states. Further, because robots during landing experience high-magnitude impulses if their legs are not exactly aligned, bipedal jumping has a very limited success rate. There are very few demonstrations of bipeds performing dynamic maneuvers consistently.

2.3 Background

2.3.1 Quaternions

Unit Quaternions

We use unit quaternions for our attitude representation, as they are singularity-free, unlike Euler angles, and are simpler to normalize than rotation matrices. Quaternions are defined as a four-length vector with a norm of 1, $q \in S^3 \in \mathrm{R}^4$. Conventionally, quaternions are broken into a three-length vector part, $v \in \mathrm{R}^3$, and a scalar part, $w \in \mathrm{R}$. We will use the Hamiltonian notation placing the scalar part first:

$$q = \begin{bmatrix} s \\ \mathbf{v} \end{bmatrix} \tag{2.1}$$

Quaternions can be easily related to axis-angle notation, which defines a rotation $\theta \in \mathrm{R}$ around an axis $\mathbf{a} \in \mathrm{R}^3$:

$$q = \begin{bmatrix} \cos(\theta/2) \\ \mathbf{a}\sin(\theta/2) \end{bmatrix} \tag{2.2}$$

Multiplying two quaternions can be written using the skew symmetric operator for a vector $[\mathbf{v}]^\times$ and the orthonormal matrix operators $[q]_L$ and $[q]_R$, where

$$[q]_L = \begin{bmatrix} q_s & -q_v^T \\ q_v & q_s I_3 + [q_v]^\times \end{bmatrix} \tag{2.3}$$

$$[q]_R = \begin{bmatrix} q_s & -q_v^T \\ q_v & q_s I_3 - [q_v]^\times \end{bmatrix} \tag{2.4}$$

.

Using these conventions, we can express quaternion multiplication as

$$q_1 \times q_2 = [q_1]_L q_2 = [q_2]_L q_1. \tag{2.5}$$

2.3.2 MPC

Previous work on RWAs for legged robots largely uses a simple proportional-derivative controller, ignoring the rest of the body dynamics. Adding a cross-coupling term to relate the leg and the reaction wheel control inputs has been underexplored, but is relatively simple to do and requires only linear dynamics. In order to do so, we must further explore our system dynamics and combine the standard floating-base dynamics with gyrostat dynamics.

Legged Centroidal Dynamics

Most legged MPC controllers are built on a convex centroidal approach. This separates the controller into two sub-problems: center of mass control and footstep planning. We assume that the mass of the body is large enough such that we can neglect the leg masses, simplifying this problem to a single rigid body and two forces from the ground. Given the body mass m and a body frame inertial moment ${}^{B}J_b$, the dynamics equation becomes

$$\dot{x} = \begin{bmatrix} \dot{r} \\ \dot{\theta} \\ N_{\dot{r}} \\ B_{\dot{\omega}} \end{bmatrix} = \begin{bmatrix} N_v \\ \Omega(\theta)^{B}\omega \\ \frac{1}{m}{}^{N}F - g \\ ({}^{B}J_b)^{-1}({}^{B}\tau_f - {}^{B}\omega) \end{bmatrix} \tag{2.6}$$

We assume we know the system state, which includes the center of mass $r \in \mathrm{R}^3$, the attitude of the body $\theta \in \mathrm{R}^3$, the linear velocity ${}^{N}v \in \mathrm{R}^3$, and the angular velocity ${}^{B}\omega \in \mathrm{R}^3$.

The inputs to the system are a force input ${}^{N}F \in \mathrm{R}^3$ and a torque input ${}^{B}\tau \in \mathrm{R}^3$ from the ground reaction forces. We can calculate the force and torque inputs from the ground reaction force components f using the relation

$$u = \begin{bmatrix} {}^{N}F \\ {}^{B}\tau_f \end{bmatrix} = \begin{bmatrix} I_3 & \dots & I_3 \\ R^T[p_1]^x & \dots & R^T[p_n]^x \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \tag{2.7}$$

where I_n signifies an n by n identity matrix. We also place a friction cone constraint

on the ground reaction force to prevent foot slip and approximate the friction cone as a pyramid:

$$-\mu f_z \leq f_x \leq \mu f_z \tag{2.8}$$

$$-\mu f_z \leq f_y \leq \mu f_z \tag{2.9}$$

Gyrostat Dynamics

One of the major advantages of using a RWA is that the motion of the reaction wheel does not change the inertia tensor of the system. This means that we can model our robot as a gyrostat, with multiple rigid bodies whose motion does not change the inertial properties. Assuming that the reaction wheel was manufactured with perfect dynamic balance, it will have a constant inertia in the body frame of the robot that can be written as

$$^B J_r =^B L \tag{2.10}$$

where $^B L$ is the inertia of the wheel expressed in the robot's body frame. The total angular momentum of the RWA is

$$h_r =^B L(^B\omega ^B\psi) \tag{2.11}$$

where $^B\omega \in \mathrm{R}^3$ is the body angular velocity and $^B\psi \in \mathrm{R}^3$ is the angular velocity of the wheel in the body frame. We can then write the total angular momentum of the robot $^B H$ as

$$^B H =^B J_b^B \omega + h_r \tag{2.12}$$

We can derive the equation of motion for the gyrostat by taking the derivative of the angular momentum.

$$^B \tau_f =^B J^B \dot{\omega} +^B \omega \times (^B J^B \omega +^B L^B \psi) +^B \tau_{rw} \tag{2.13}$$

where $^B\tau_f$ is the net body-frame external torque on the robot, $^B J$ is the sum or $^B J_r$ and $^B J_b$, and $^B\tau_{rw}$ is the body-frame net torque from the RWA. We can write this in

terms of ρ, the angular momentum of the reaction wheels using the equation

$$\sum_{i=1}^{k} {}^{B}L_i^B \psi_i = \Lambda_{rw}^T \rho \tag{2.14}$$

where Λ_{rw} is the constant Jacobian matrix. This gives us the expression

$${}^{B}\tau_f = {}^{B}J\dot{\omega} + {}^{B}\omega \times ({}^{B}J^{B}\omega + \Lambda_{rw}^T \rho) + \Lambda_{rw}^T u_{rw} \tag{2.15}$$

where u_{rw} is the input torque to the reaction wheel. This expression can be written in the standard manipulator form,

$$M\dot{v} + C(q, v) = B(q)u \tag{2.16}$$

where we can define our mass matrix M, dynamic bias $C(q, v)$, and control mapping.

$$\begin{bmatrix} {}^{B}J & 0 \\ 0 & I_k \end{bmatrix} \begin{bmatrix} {}^{B}\dot{\omega} \\ \dot{\rho} \end{bmatrix} + \begin{bmatrix} {}^{B}\omega \times ({}^{B}J^{B}\omega + \Lambda_{rw}^T \rho) \end{bmatrix} = \begin{bmatrix} \Lambda_{rw}^T \\ I_k \end{bmatrix} \begin{bmatrix} u_r \end{bmatrix} \tag{2.17}$$

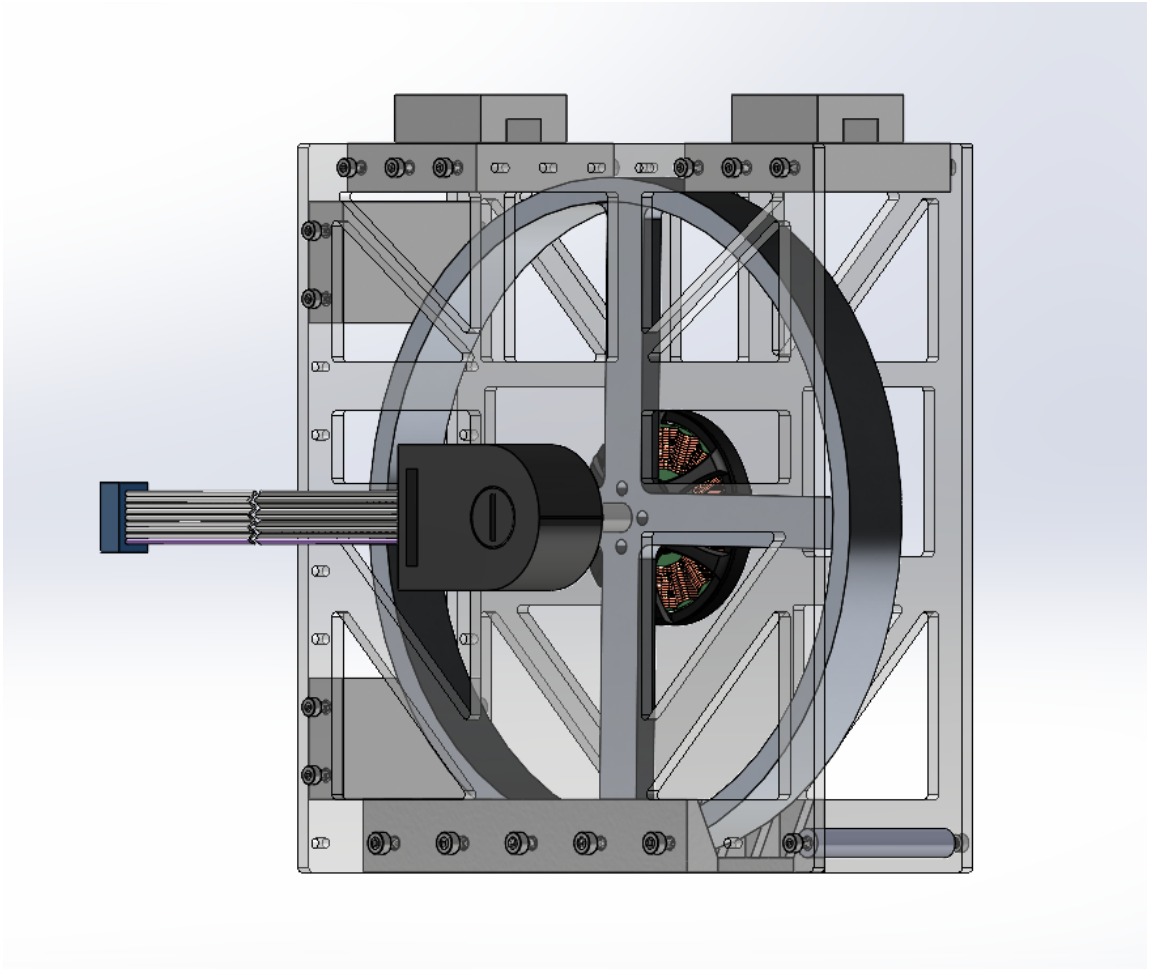

Figure 2.6: Reaction Wheel Assembly

2.4 Hardware

This section elaborates on the mechanical design of the Reaction Wheel Actuator (RWA). The RWA was optimized for weight, and the center of mass of the RWA was designed to be positioned over the original center of mass of the biped. The reaction wheel actuator (RWA) unit is driven by a brushless DC motor, with a continuous maximum current draw of 12 A and a maximum torque of 0.7 Nm. The system has dimensions of 150 x 150 x 46 mm and a total mass of 0.284 kg.

2.4.1 Electronics

The motor driver used to control the RWA are miniaturized versions of the TI micro-controller boards, and are designed by ODRI using a Texas Instruments micro-controller chip. They are capable of dual motor torque control and operate at 10 kHz. The board has CAN and SPI control communication abilities and operates at up to 40 V.

The encoder used for the reaction wheel system is the AEDM-5810-Z12 Broadcom encoder. This module contains the same optical encoder used in the rest of the biped joint motors, making integration simple. This encoder has a maximum velocity of 12,000 RPM.

The motor used for the reaction wheel is similar to the motors used for the rest of the RWA system, but is able to output a continuous torque of 0.7 Nm. This allows the biped to recover from perturbation while standing and control pitch while in the air.

2.4.2 Design

The system is primarily composed of acrylic plates and 3D-printed polylactic acid (PLA). The main components are the reaction wheel, which is machined out of 6061 aluminium, the motor, and the encoder. The reaction wheel and lathe both require precision machining, and the acrylic plates require a laser cutter. Additionally, several 3D printed structural components have been modified to integrate the RWA on top of the original body, as shown in Fig. 2.7. The modified CAD and STL files are included with the rest of the CAD for the RWA.

The CAD and STL files are available online at https://github.com/sofiakwok/biped-rw.

2.4.3 Machining

The main machined components are the reaction wheel and the shaft. The reaction wheel was manufactured out of 6061 aluminium alloy on a CNC milling machine to ensure accurate distribution of mass along the wheel rim. The shaft was manufactured out of high-strength 1045 carbon steel rod on a lathe. The drawings for both are

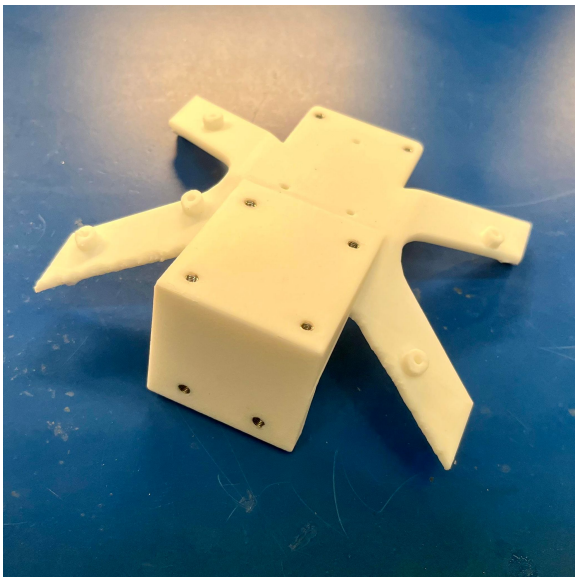

Figure 2.7: The modified top of the body structure

included in Figs. 2.8 and 2.9.

2.4.4 3D Print Preparation

All 3D prints were printed with 80 percent infill on a Bambu Carbon X1 printer, using Bambu PLA. All prints were sliced using a 0.20mm standard preset. To prepare the parts, we use Helicoil threaded inserts. This requires tools for installing M2 and M2.5 Helicoil threaded inserts, as shown in Fig. 2.11. Refer to Figs. 2.12, 2.13, 2.14, and 2.15 for a step-by-step tutorial on how to properly install Helicoils.

2.4.5 Acrylic Plate Preparation

For the side plating, 0.125" acrylic plate was used. Acrylic has the best material properties for our weight-constrained system, and is also easily machined and widely available. The acrylic was cut on a SD-90-1390 Rabbit CO2 laser cutter, which offers location precision of 0.01 mm. The weight cutouts and through holes for screws do not need to be rigorously dimensioned or undergo post-processing. However, the bearing mounts on each plate should be an interference or friction fit.

There are several different approaches to correctly size the bearing mount hole.

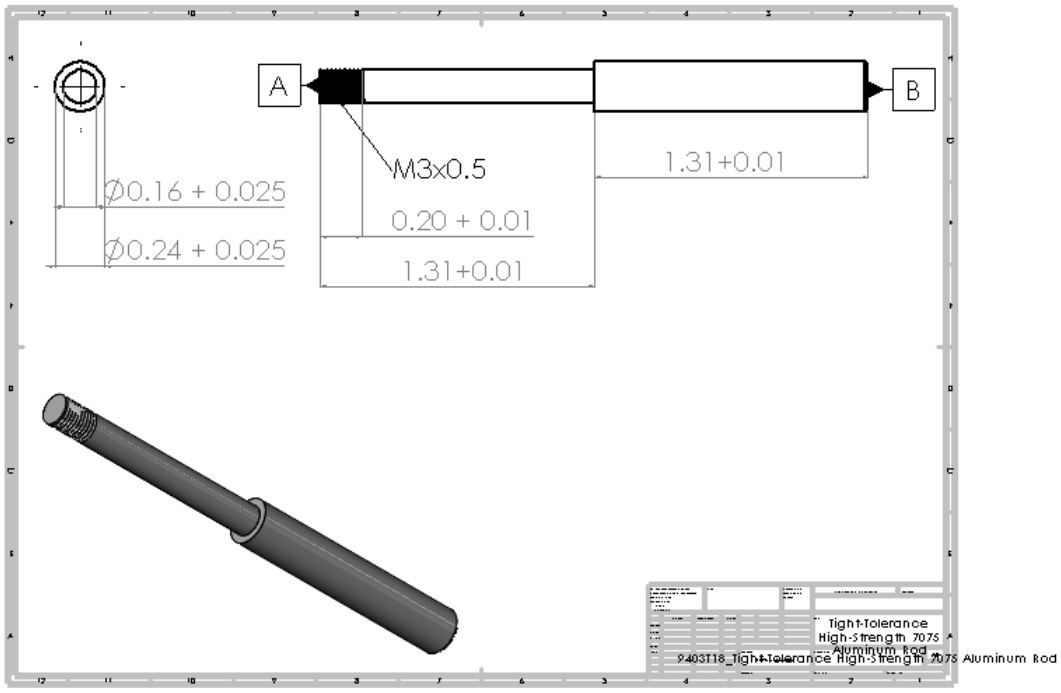

Figure 2.8: Shaft GD&T Drawing

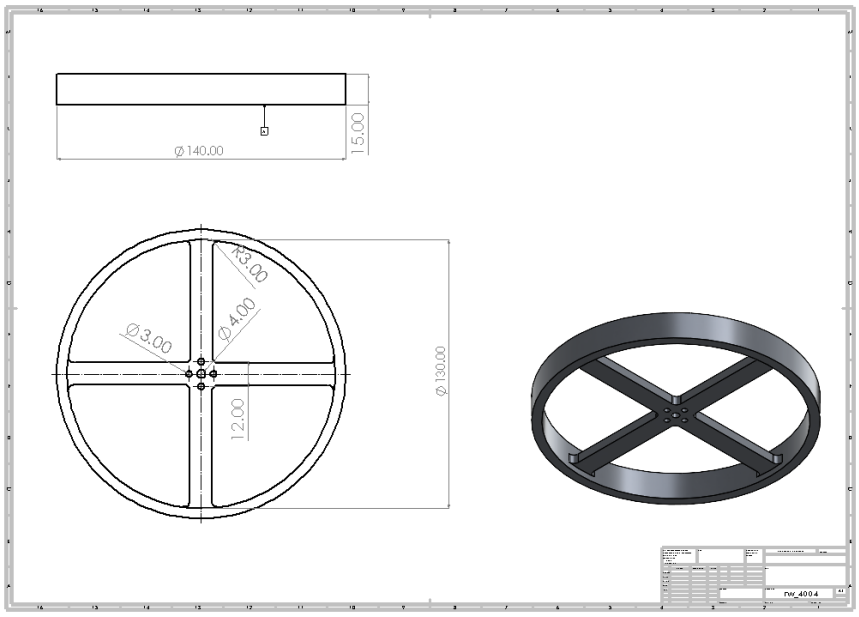

Figure 2.9: Reaction Wheel GD&T Drawing

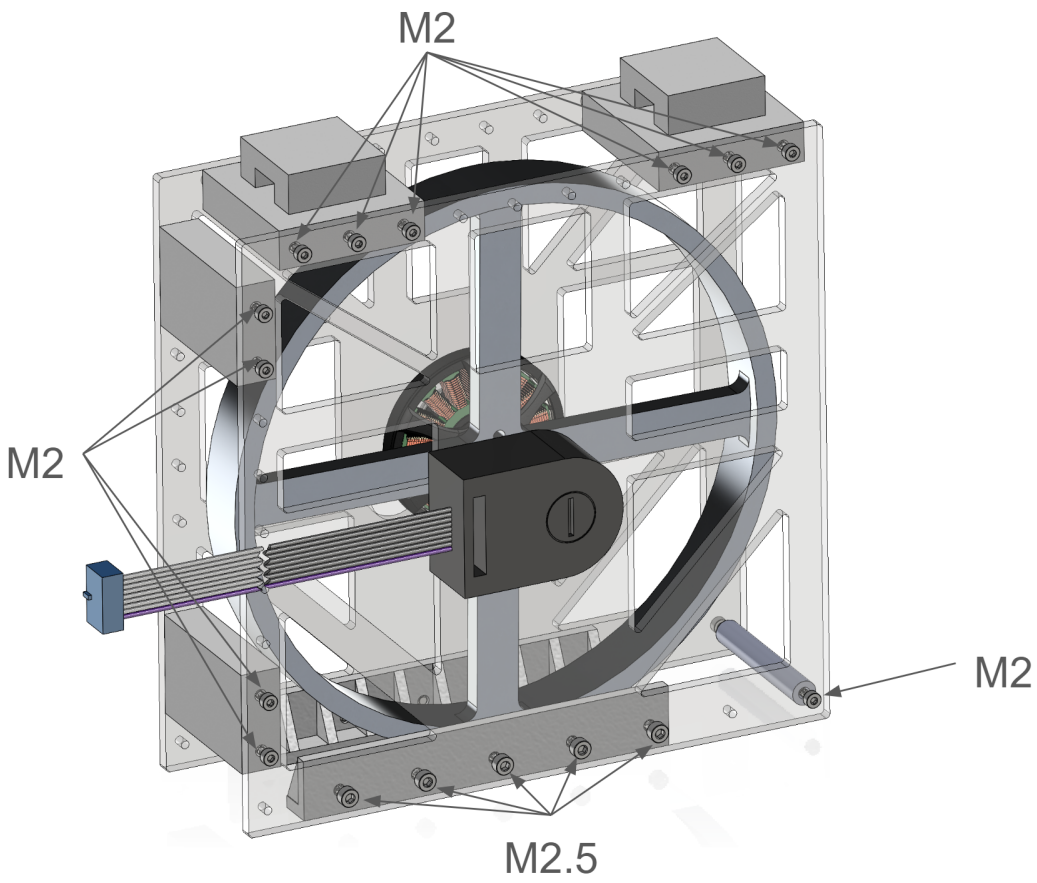

Figure 2.10: Installation locations and sizing for Helicoils

It is possible to do multiple test tolerance cuts with the laser cutter, or deliberately undersize the hole and use a drill press to correctly tolerance the hole. The bearings on both sides are flanged to ensure correct depth insertion. The bearing should be able to be removed and inserted without using a press fit machine. It is a good idea to chamfer the hole edges for easier insertion and removal.

2.4.6 Bill of Materials

Materials and costs at the time of publication are included in Table 2.1. The total cost for all materials, installation tools, and electronics comes out to less than $250 USD.

Figure 2.11: Parts required to correctly install Helicoils

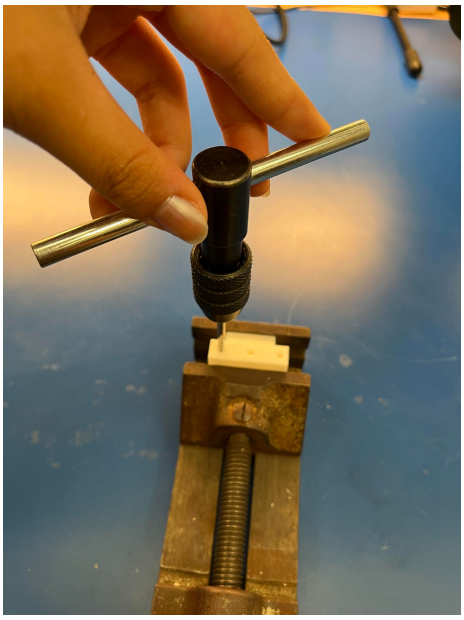

Figure 2.12: Tap the 3D printed part with the correctly size Helicoil tap

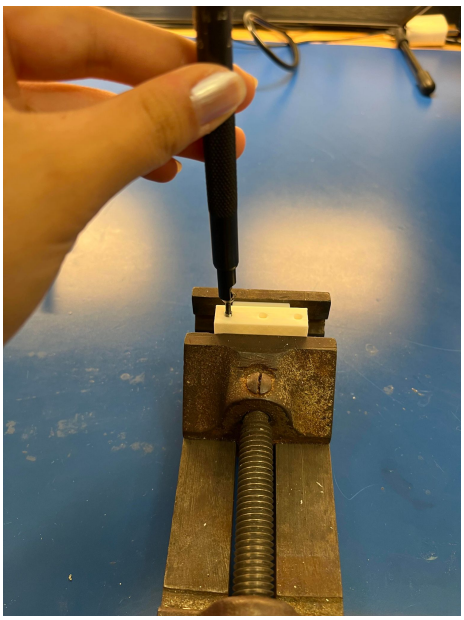

Figure 2.13: Screw the Helicoil inserts into the tapped hole with the correctly sized installation tool

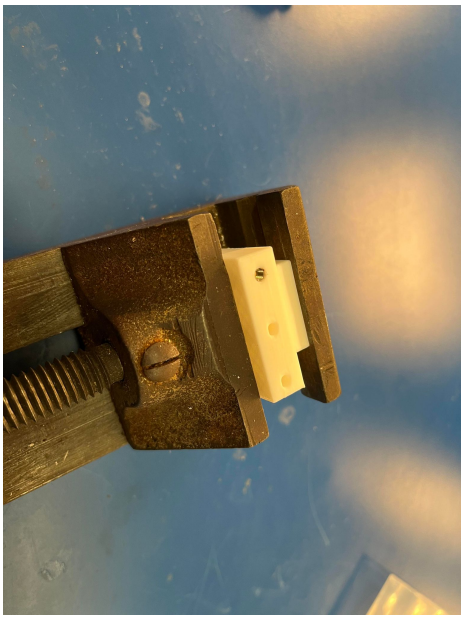

Figure 2.14: Make sure that the Helicoil is positioned one quarter-turn of the installation tool below the surface of the part

Figure 2.15: Remove the tang using the correctly sized tang break-off tool. Make sure that the tang does not stay inside the part

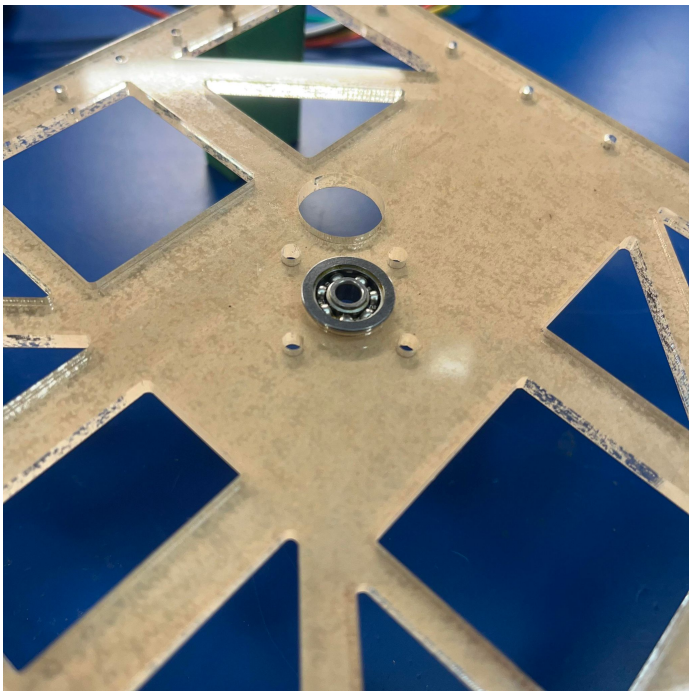

Figure 2.16: Flanged bearing correctly mounted in acrylic

2.4.7 Assembly

Figures 2.17, 2.18, 2.19, 2.20, 2.21, 2.22, 2.23, and 2.24 describe the assembly process for the RWA. For the screws attaching the reaction wheel to the rotor, Locktite 243 was used to prevent loosening during use. The nuts used to fasten the encoder baseplate and to constrain the tapped end of the shaft are nylon-insert locknuts to prevent screws coming loose during use.

2.4.8 Testing

The setup for the RWA is similar to the original Bolt system architecture to enable easy integration. The assembled system can either be tested using a low-level script in master-board-sdk or can be tested with just the motor driver boards in virtual spring mode.

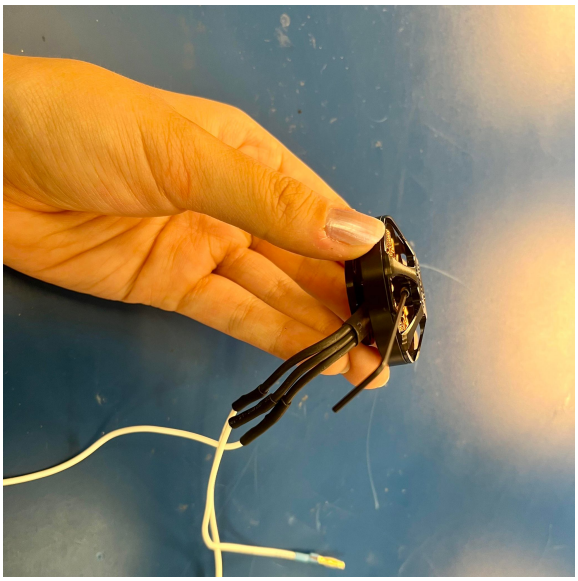

Figure 2.17: Separate the rotor and stator by loosening the set screws

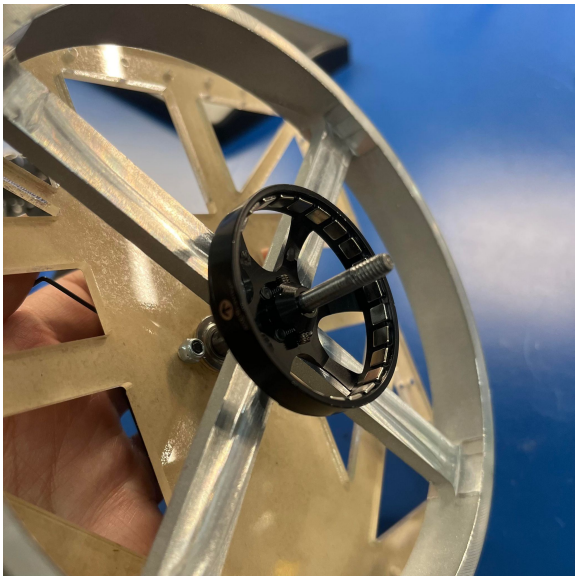

Figure 2.18: Mount the rotor to the reaction wheel

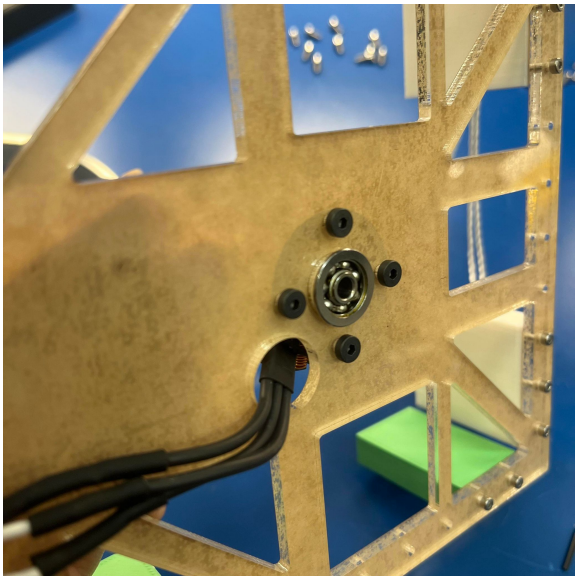

Figure 2.19: Mount the stator to the acrylic plate. Route the stator mounting wires through the cutout as shown in the picture

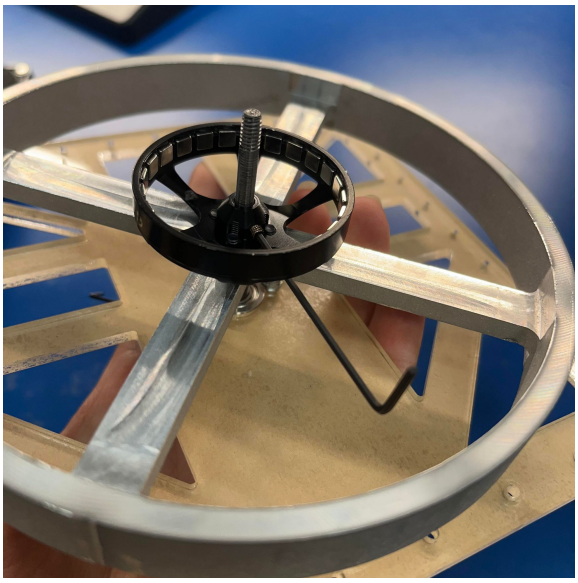

Figure 2.20: Tighten the rotor set screws

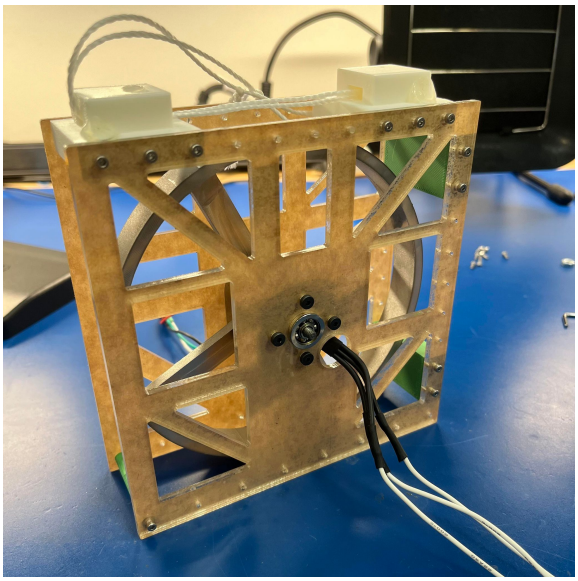

Figure 2.21: Attach rotor and stator together and screw together 3D parts

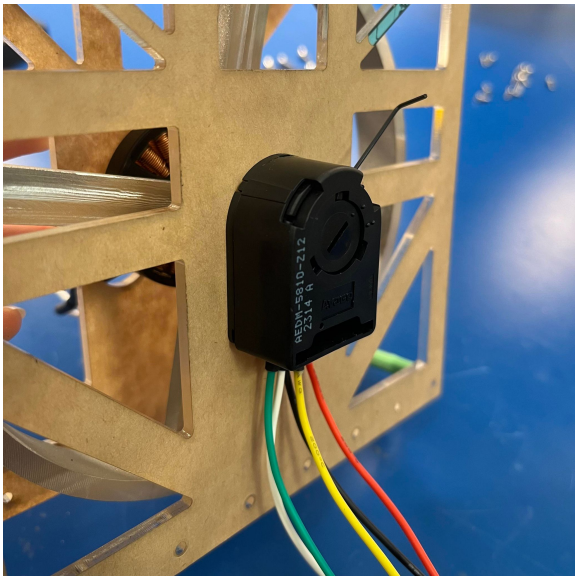

Figure 2.22: Mount encoder according to part documentation

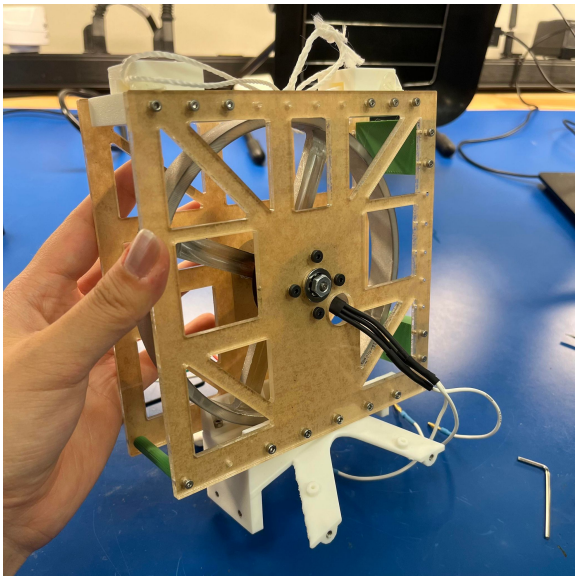

Figure 2.23: Mount RWA to modified biped top

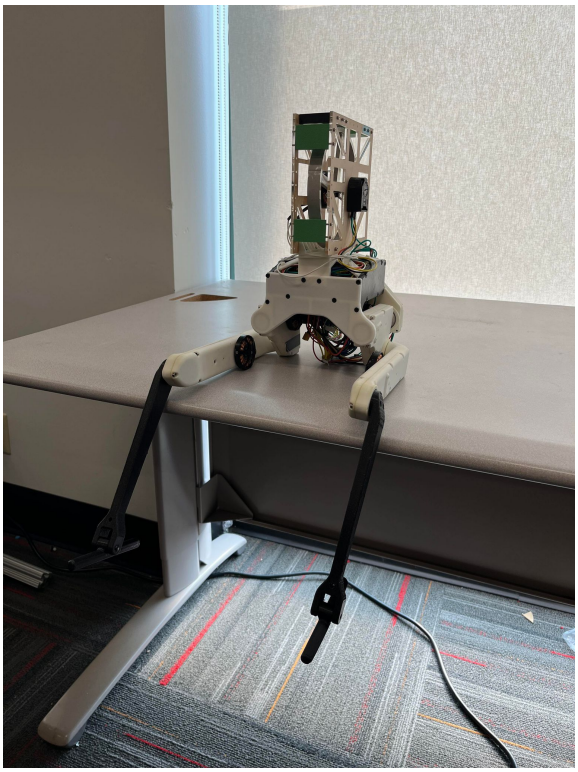

Figure 2.24: Attach RWA to Bolt

Part	Quantity	Cost
Encoder	1	71.63
Motor	1	70
Stock Acrylic	12”x6”	5.70
Stock Aluminium	6”x6”x5/8”	24.75
PLA	1 spool Bambu PLA Basic	19.99
Flanged Bearings	2	26.04
M2 Screws		5.68
M2.5 Screws		7.43
M2 Helicoil Installation Kit	1	13.35
M2.5 Helicoil Installation Kit	1	13.35

Table 2.1: RWA Bill of Materials

2.5 MPC

This section goes into further detail about how we create and solve our optimization problem. First, we incorporate the RWA dynamics into our centroidal model and linearize. We can then discretize our calculated transition matrices to make a convex discrete-time optimization problem. Finally, we solve for the motor torques using the calculated ground reaction forces.

2.5.1 Biped RWA Dynamics

Adding our RWA for pitch control to the centroidal dynamics of our system, we obtain the equations of motion

$$\dot{x} = \begin{bmatrix} \dot{r} \\ \dot{\theta} \\ {}^{N}\ddot{r} \\ {}^{B}\dot{\omega} \\ \dot{\rho} \end{bmatrix} = \begin{bmatrix} {}^{N}v \\ \Omega(\theta)^{B}\omega \\ \frac{1}{m}{}^{N}F - g \\ ({}^{B}I)^{-1}({}^{B}\tau_f + \\ u_r \end{bmatrix} \tag{2.18}$$

where $u_r \in \mathrm{R}$ is the torque input vector for the RWA and ρ is the momentum state of the reaction wheel.

We can linearize the dynamics of our system by assuming that body angular

velocity, attitude change, and RWA velocities are small. This eliminates the Coriolis term for the rotational and gyrostat dynamics. We can then write the rotation representation for the system assuming small body angle changes, giving us a linearized representation for the rotational kinematics and dynamics:

$$\dot{\theta} \approx R_z^T(\theta_y)\,{}^B\omega\,{}^B\dot{\omega} \approx ({}^BI)^{-1}({}^B\tau_f + \tau_\rho) \tag{2.19}$$

This gives us the linearized dynamics

$$\frac{d}{dt}\begin{bmatrix} r \\ \Theta \\ {}^W\dot{r} \\ {}^W\omega \\ \rho \end{bmatrix} = \begin{bmatrix} 0_3 & 0_3 & I & 0_3 & 0_{31} \\ 0_3 & 0_3 & 0_3 & R_z^T(\theta_y) & 0_{31} \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_{31} \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_{31} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ \theta \\ {}^N\dot{r} \\ {}^N\omega \\ \rho \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \\ 0 \\ 0 \end{bmatrix} +$$

$$\begin{bmatrix} 0_3 & \dots & & 0_{31} \\ 0_3 & \dots & & 0_{31} \\ \frac{1}{m} & \dots & \frac{1}{m} & 0_{31} \\ {}^N J^{-1}[p_1]^\times & \dots & {}^N J^{-1}[p_n]^\times & {}^N J^{-1} R^T \Lambda_{rw}^T \\ 0 & \dots & & I \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \\ u_{rw} \end{bmatrix}$$

We can now express our dynamics as a linear function where

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.20}$$

2.5.2 Trajectory Optimization

We can convert our transition and control matrices A and B into discrete-time matrices for hardware. We can then write our control problem as a convex discrete-time quadratic program:

$$\min_{x,u} \sum_{0}^{k-1} \|x_{i+1}^d - x_{i+1}\| + \|u_i\| \tag{2.21}$$

$$\text{subject to } x_{i+1} = A_i x_i + B_i u_i, i = 0, ..., k-1 \tag{2.22}$$

$$c_i \leq C_i u_i \leq \bar{c}_i, i = 0, ..., k-1 \tag{2.23}$$

$$D u_i = 0, i = 0, ..., k-1 \tag{2.24}$$

where x_i, u_i, Q_i, R_i are the robot state, control inputs, cost matrix for state, and the cost matrix for control at timestep i. C_i enforces the linearized friction cone constraints for our ground force reaction vectors, and D_i constrains foot forces to be 0 when a foot is in the swing phase. We can write this as a quadratic program (QP) and solve for the ground reaction forces for feet in contact with the ground. Then, we can calculate the joint torques for a leg in contact with the ground using the relation

$$\tau_i = J_i^T R^T f_i \tag{2.25}$$

where f_i is the ground force, J_i is the forward kinematic Jacobian, R is the world to body rotation matrix, and τ is the joint torque.

2.5.3 Swing Leg Control

For bipedal walking, we assume that one foot is always in contact with the ground, and that it is pinned with no slip. To calculate the swing foot trajectory, we first take our desired foot placement on the xy plane and interpolate between our current position and desired position using a polynomial spline. We then can write the necessary joint torque for the swing leg as

$$\tau = J^T(K_p({}^B p^d - {}^B p) + K_d({}^B v^d - {}^B v)) \tag{2.26}$$

where J is the foot Jacobian, K_p and K_d are our proportional and derivative gain matrices, ${}^B p$ and ${}^B v$ are the position and velocity of the swing foot in the body frame, and ${}^B p^d$ and ${}^B v^d$ are the reference position and velocity along the generated trajectory.

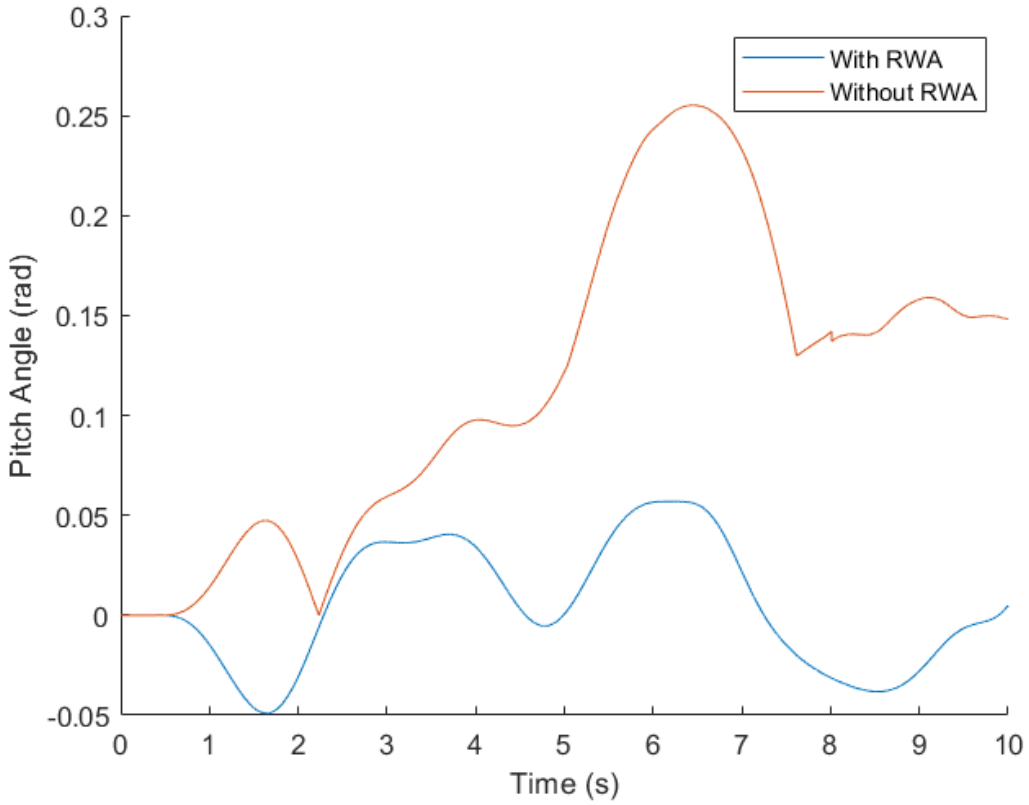

Figure 2.25: Drop Test Recovery

2.6 Results

2.6.1 Simulation Results

Drop Test

We first simulate the biped and RWA in Pybullet. We perform a simple drop test from a height of 0.5 meters using a PD controller to control pitch angle. The model with the reaction wheel is able to recover and stabilize, while the model without a reaction wheel loses balance and tips over.

Disturbance Rejection

We then apply an impulse to the y-axis of the biped body in simulation while running a convex centroidal controller. The robot with a reaction wheel is able to recover, but the robot without a reaction wheel is unable to stabilize itself and falls over.

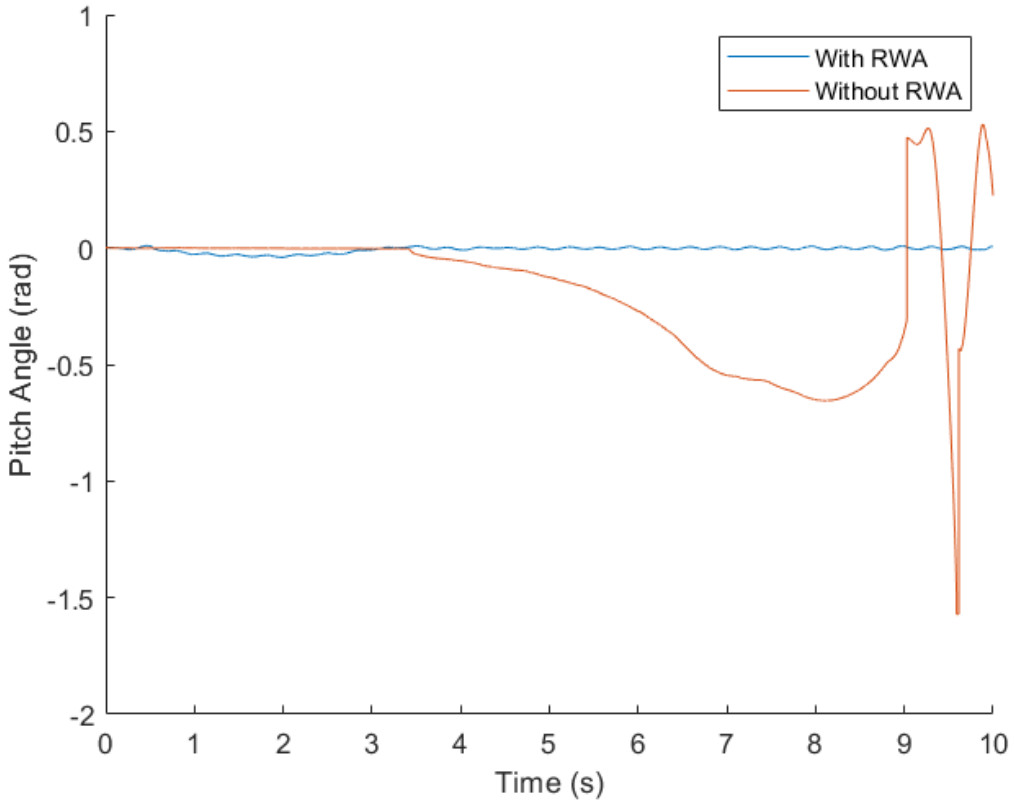

Figure 2.26: Controller with Perturbation

2.6.2 Hardware Results

We demonstrate successful balancing and stabilization with a PD controller for attitude control in Fig 2.29.

We also demonstrate a stabilizing centroidal controller, as shown in Fig. 2.30.

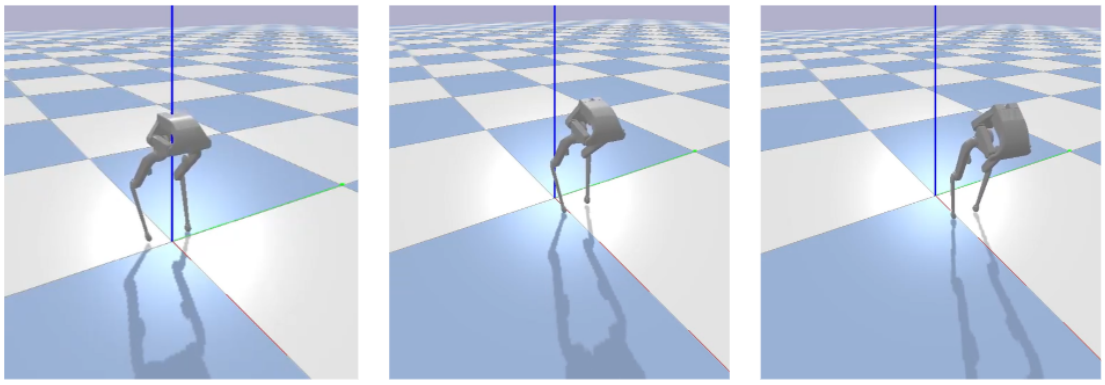

Figure 2.27: The original robot is unable to recover from a perturbation and falls over.

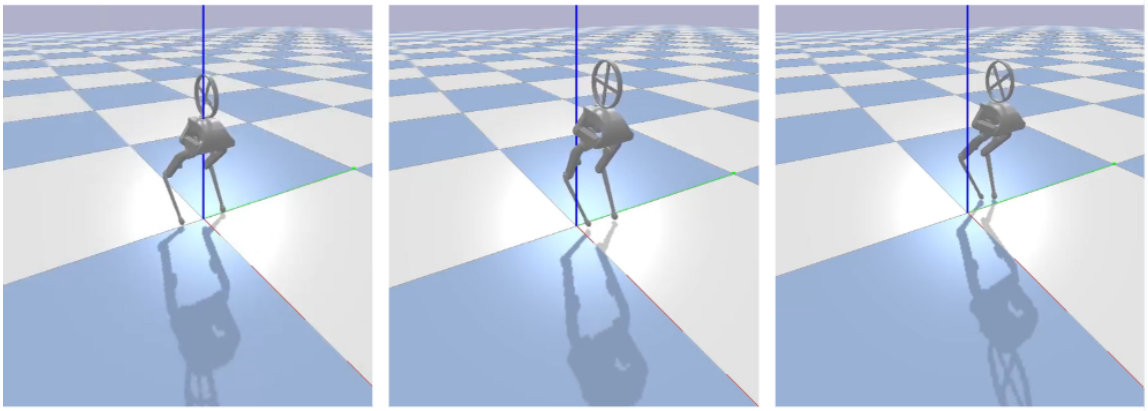

Figure 2.28: The robot with a reaction wheel is able to stabilize after a perturbation

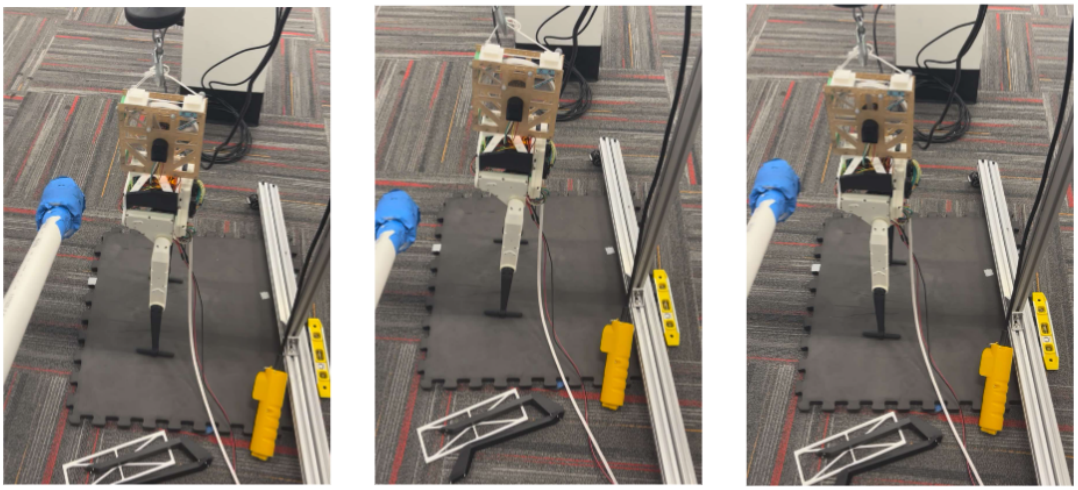

Figure 2.29: PD recovery from perturbation

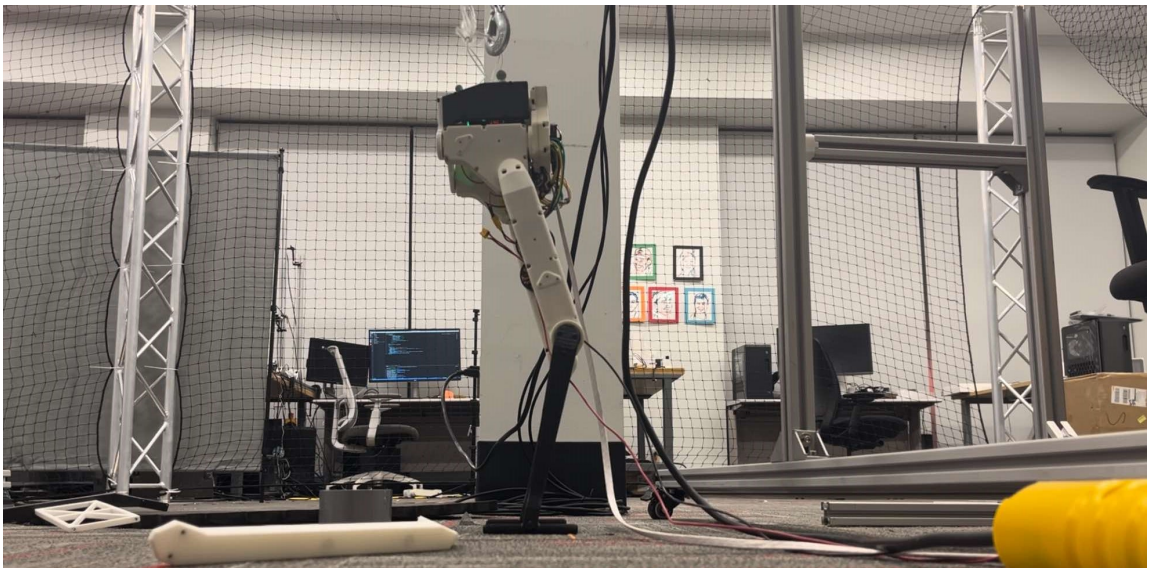

Figure 2.30: Centroidal controller base stabilization

Chapter 3

Jellyfish

3.1 Introduction

New control schema for navigating through vortices and turbulent environments show exciting developments in designing energy-efficient aquatic robots. However, testing new controllers often proves challenging. RL-based control policies require large quantities of data to train, making simulation fidelity difficult [23]. Model-based control policies also require an accurate representation of the robot and the interactions between the fluid and the linkages. In order to test and develop novel control algorithms on aquatic robots, we require accessible and simple hardware platforms.

Commercially sold underwater robots are frequently intended for underwater inspection or servicing, weighing many kilograms and costing thousands of dollars. These commercial products are often not feasible for researchers, as they require a large body of water to test in and foster a reliance on proprietary design and software knowledge. Previous research projects in aquatic robots are often under actuated and involve soft, deformable material [3]. This makes the robot incredibly difficult to accurately simulate, since modeling fluid interactions with rigid bodies often requires complicated governing physics equations. Additionally, many high-fidelity simulators only analyze a planar fluid field due to processing constraints [16]. This means that any robot which varies in the z-plane cannot be realistically modeled.

We introduce BLOOPER (Bio-inspired Locomotion Optimization and Oceanic

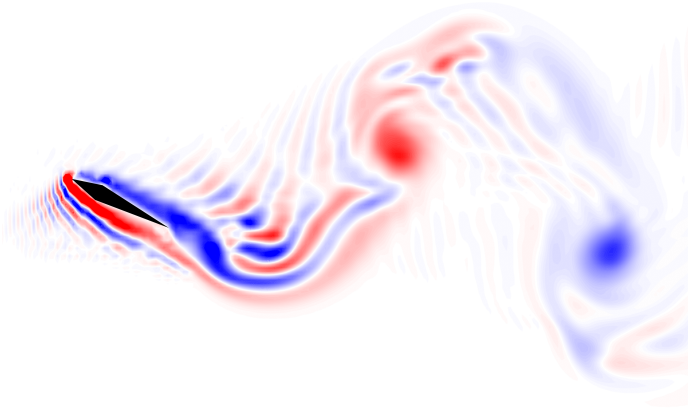

Figure 3.1: Fluid simulation of vorticity off of a rigid body [16]

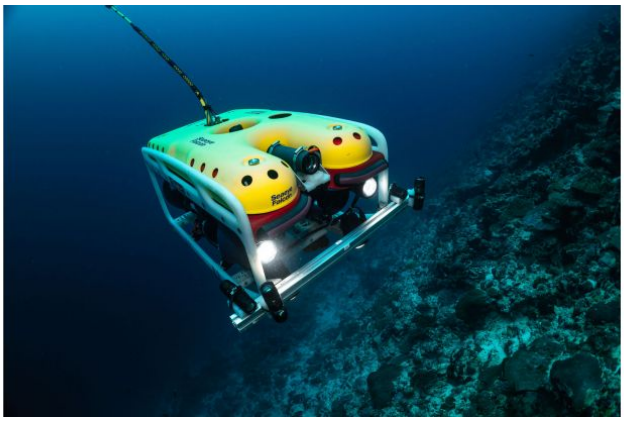

Figure 3.2: An underwater robot for ocean surveillance and mapping [2]

Planar Evaluation Robot), a cheap, open-source hardware platform for fluid testing. BLOOPER has very little variance in the z-plane, narrowing the sim-to-real gap. It comes with a controller for the five-bar linkage that makes up the fins and rudders, which is able to rapidly and accurately travel to the end-effector positions.

3.2 Related Work

3.2.1 Aquatic Robots

There are many open-source aquatic robots. However, many of these are designed for industrial applications or environmental monitoring, and are nontrivial to maintain and simulate. Open-source robots are also often expensive, costing thousands of dollars to build, and by necessity have complicated control systems [9].

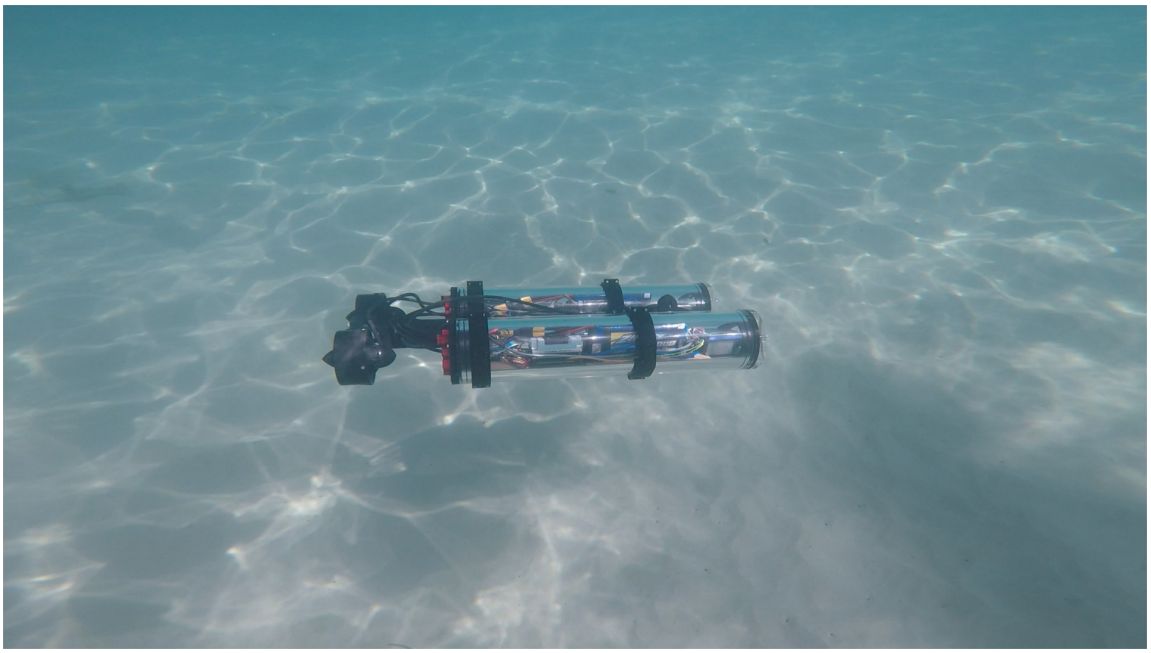

Figure 3.3: LoCO AUV, an open-source autonomous underwater vehicle [9]

Other aquatic robots are bio-inspired and mimic the actuation modes of sea creatures, but are not easily modeled due to their soft structure [25]. This makes sim-to-real transfer difficult or near-impossible, since deformable materials lead to out-of-plane motion. Unfortunately, since most aquatic animals use deformable or flexible materials for propulsion, there is a tradeoff between accuracy and complexity. Most research platforms compromise by using several actuators inside a deformable structure to generate thrust.

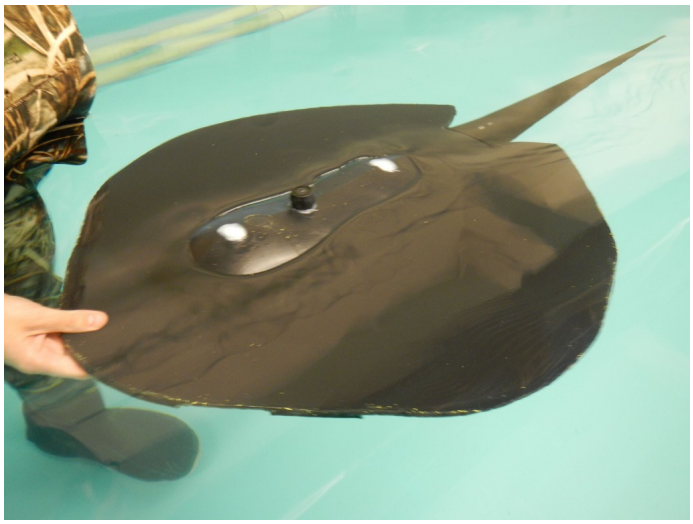

Figure 3.4: Robotic stingray from MIT Mechatronics Lab [25]

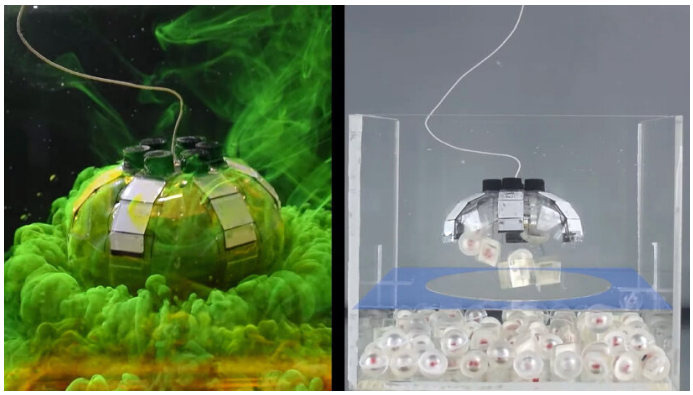

Figure 3.5: Robotic jellyfish from Max Planck Society [27]

3.2.2 Robotic Jellyfish

Previous work on robotic jellyfish has focused on the swimming mechanisms of jellyfish, leaving the robots under actuated [27]. While realistic to nature, having an inherently uncontrollable system is not advantageous for testing many new control schema. Additionally, many jellyfish robots are made of deformable material, adding an additional layer of complexity to modeling [26].

3.3 Background

3.3.1 Five-Bar Linkage

Five-bar linkages have been used in robotics for engraving, as legs for novel robot designs, and as mechanisms for haptic feedback. They are used over serial linkage mechanisms where precise control of an end-effector is desired, as the inherent mechanical stiffness of the system reduces the possible slop.

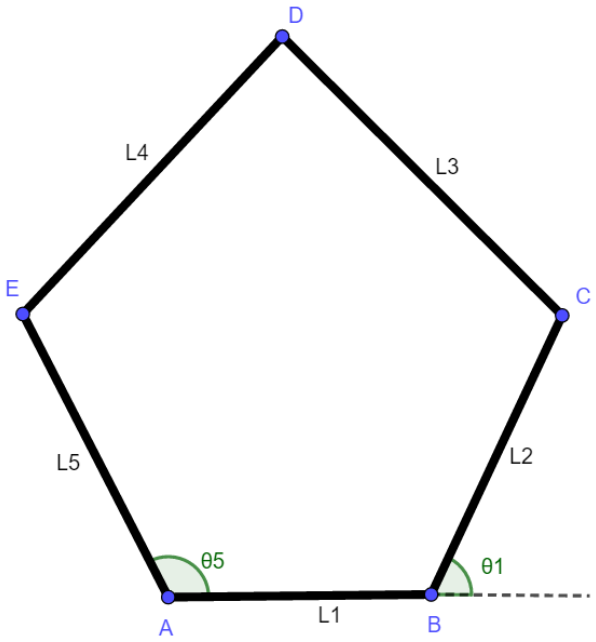

Figure 3.6: Five-bar linkage mechanism by I. Eglamal

We can further understand the mechanical advantages of a five-bar linkage mechanism by studying Fig. 3.6. When two actuators are placed at the base points A and B, we can control the angles θ_1 and θ_5, allowing us to control the position of the end effector. This gives the system two degrees of freedom, and lets the end effector cover the area within the reach space.

There are many resources online to help model and optimize the link lengths, actuator spacing, and other components of the linkage design, including a built-in Matlab object and Python package that can help to solve for the best design

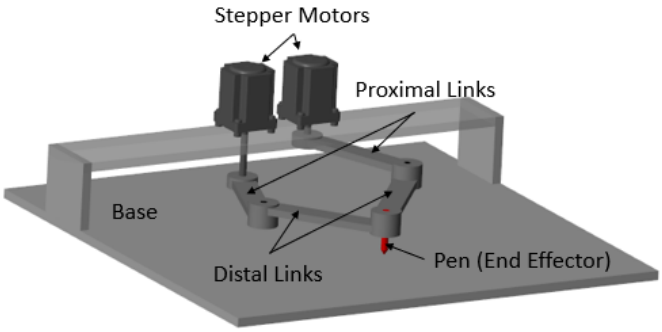

Figure 3.7: Matlab kinematics package for a five-bar robot [1]

parameters. It is important to avoid the situation where one of the arms fully extends, reducing the degrees of freedom of the end effector and leading to a kinematic singularity. Although our controller does not encounter mathematical problems when a singularity is reached, it does increase the slop in the system, and can lead to unnecessary strain on the actuator.

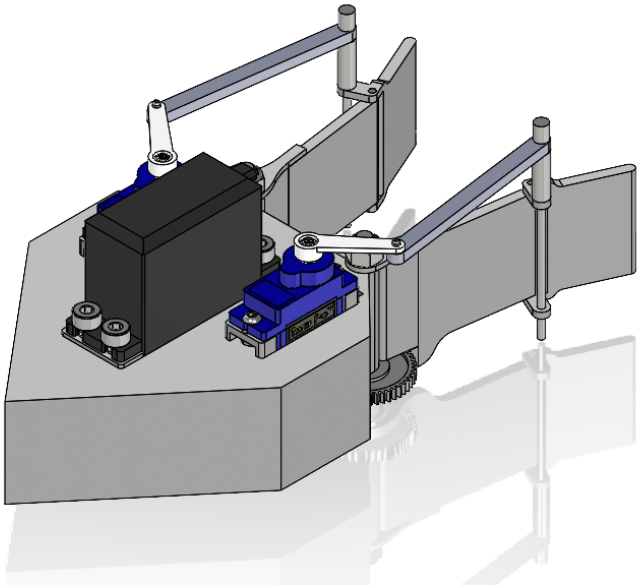

Figure 3.8: Jellyfish Assembly

3.4 Hardware

BLOOPER is designed to be cheap and easy to assemble. The system has less than 0.01 degrees of error when correctly toleranced, making it a good option for RL or model-based control policies.

3.4.1 Vision

In order to find the robot position, we use an Aruco marker on the jellyfish body coupled with a USB camera. Using publically available OpenCV packages to find the position and angle of the Aruco marker gives us the state of the robot.

3.4.2 Electronics

We use servos for this project, as they offer increased precision and do not require an external encoder. For the central servo, which requires more torque due to water resistance, we use a MG995 servo. The outer two servos experience relatively little torque and thus are smaller and cheaper SG90 servos. The MG995 requires more

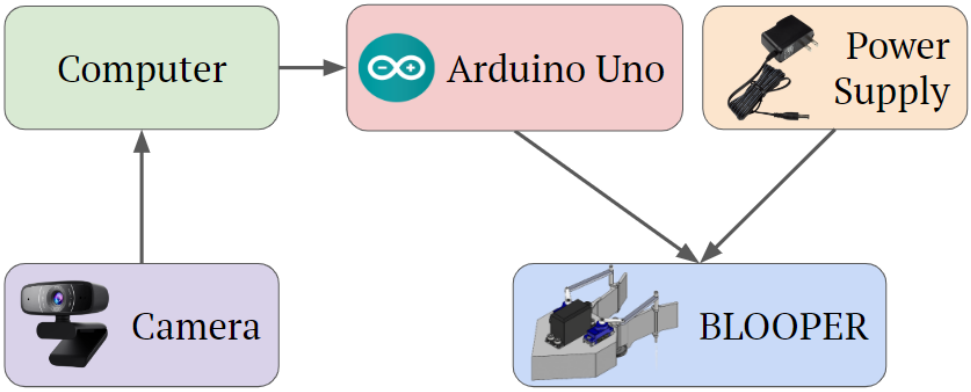

Figure 3.9: Electronic system for BLOOPER

voltage than the Arduino can supply, and thus requires an external power source. We use a 5V wall converter to power all three servos with a breadboard.

For controlling the robot, we use an Arduino Uno. This can be connected via serial port to a laptop, making it easy to simultaneously record the robot state and run a controller.

3.4.3 Waterproofing

BLOOPER is designed such that only the body, fins, and rudders are submerged, making modeling the structure much simpler. This has the additional benefit of considerably reducing the risk of shorting the motors.

In order to reduce potential risk to the electronics, we keep the Arduino and power supply offboard. Due to the high cost and relatively small selection of waterproof servos, we instead use relatively inexpensive servos and avoid submerging them. The MG995's sensitive electronics are far enough removed from the water level when mounted that it requires no waterproofing. The SG90s can be made water resistant by hot gluing the wire exit port shut, as shown in Fig. 3.10.

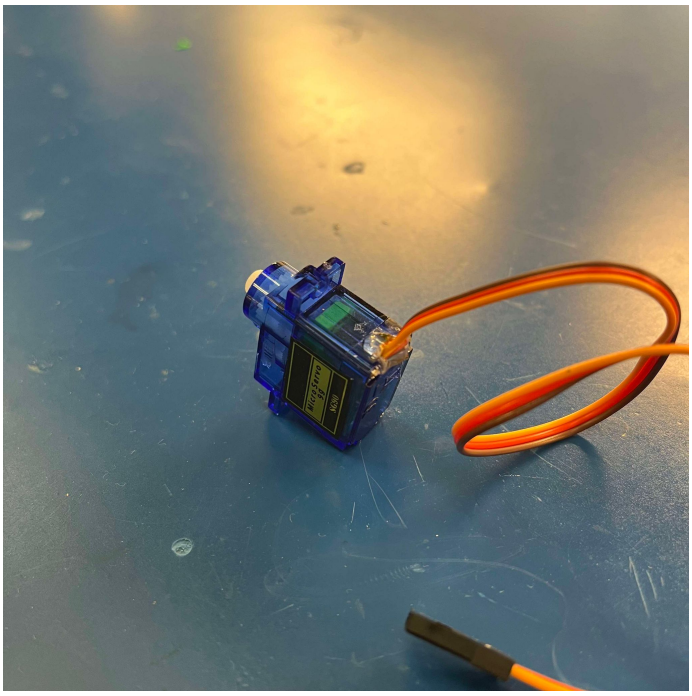

Figure 3.10: SG90 with hot-glued wire exit

3.4.4 3D Print Preparation

All 3D prints were printed with 30 percent infill on a Bambu Carbon X1 printer, using black Bambu PLA-CF. All prints were sliced using a 0.20mm standard preset. In order to ensure minimal play on pivoting parts, we deliberately under size the holes shown in Fig. 3.11. These holes are used for rotating joints and are drilled after printing to precise diameters.

Additionally, to ensure neutral buoyancy, we make the body of BLOOPER hollow. This can be done with either as little infill percentage as possible or with an internal cavity, depending on printer specifications. The internal cavity size can be adjusted depending on total weight of motors and other components that have non-trivial mass. We recommend making the body fully watertight with epoxy or a spray-on sealant, as PLA has a tendency to absorb water over time. We then hot-glue the baseplate and trim the excess glue, as shown in Fig. 3.12.

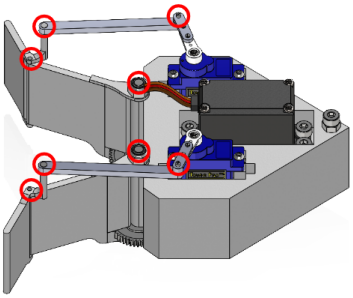

Figure 3.11: Pivot points that require precise tolerancing

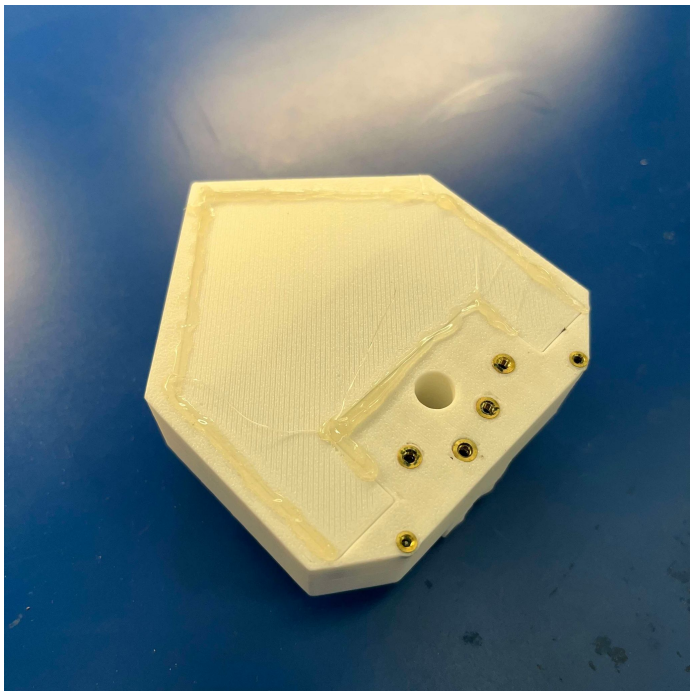

Figure 3.12: Hot-glued BLOOPER body

3.4.5 Bill of Materials

The total cost to build BLOOPER comes out to $119.52. This is considerably less expensive and time-consuming to assemble than other open-source aquatic robots.

Part	Quantity	Cost
Arduino Uno	1	27.60
MG995	1	3.82
SG90	2	3.60
5V converter	1	7.99
Dowel Pins	4	27.44
0.5 Modulus Plastic Gears - 20 T	5	5.70
0.5 Modulus Plastic Gears - 40 T	5	5.70
PLA	1 roll Bambu PLA Basic	19.99
M2.5 Screws		17.68

Table 3.1: Jellyfish Bill of Materials

3.4.6 Assembly

We use Loctite to ensure minimal axial play between gears while still preserving rotation. The M4 screws that serve as the gear shafts should be tightened until the gear can no longer shift vertically but is still free to spin. The servo power, ground, and signal wires should also be extended to ensure no electrical contact with the water. All parts should be friction or interference fit, and should not require a press or mechanical force beyond what the human hand can apply.

3.4.7 Testing

A simple test script is included in the repository that tests the serial connection between the computer and the Arduino. Once the robot is assembled, we recommend running a test script included in the Github repository that sweeps from a 0 to 90 degree fin angle with a 0 degree rudder angle. If the rudder angles look incorrect, check the offset of the side and central servos.

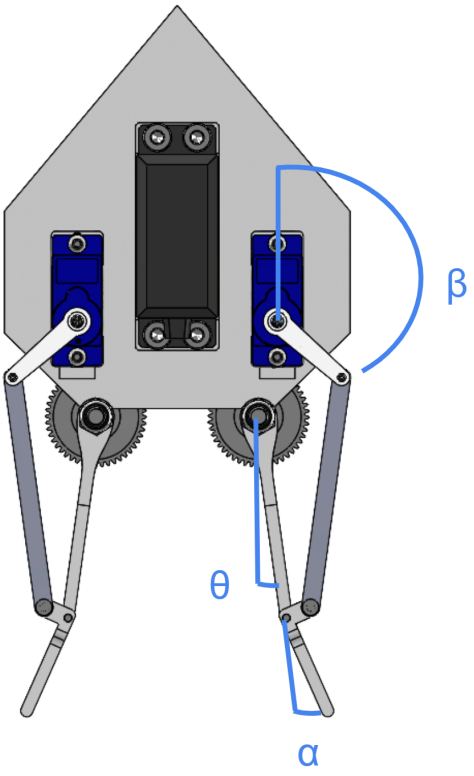

Figure 3.13: Jellyfish angle conventions

3.5 Controller

In order to understand the controller setup of the jellyfish, we can represent it as a five-bar linkage. For simplicity, we can model this as a planar linkage mechanism and assume minimal axial shift. In our case, our end effector is the rudder control point. We set this at a right angle to the rudder itself to avoid singularities during control phases, which results in a DOF loss.

There are many different approaches to solving for the control inputs and end effector position of a five-bar linkage. In our case, we know θ, the angle of the fins from the vertical, and α, the desired angle between the fins and the rudders. Knowing these two angles and the mechanical parameters of the linkage, we can solve for β, the angle that the outer servo should go to. We will calculate β for the right five-bar linkage. The calculations for β for the left five-bar linkage are near identical with

slightly different signs.

There are many different ways to solve for β. We will use trigonometry and our known distance constraints, assuming that all links are rigid and that there is zero stiffness at all joints. Assuming that our fin location is at the x-y origin, we can write the location of the fin as

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} f \sin\theta \\ -f \cos\theta \end{bmatrix} \tag{3.1}$$

where f is the length of the fin.

We can then solve for the location of the rudder control point (x_2, y_2) using our desired α:

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} r * \sin(\alpha + \theta + \pi/2) + x_1 \\ -r * \cos(\alpha + \theta + \pi/2) + y_1 \end{bmatrix} \tag{3.2}$$

where r is the distance from the rudder pivot point to the rudder control point.

Then, using the known offset (m_x, m_y) between our outer servo and fin pivot point, we can solve for the point at the end of the servo arm, E.

$$\begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} l \sin\beta + m_x \\ l \cos\beta + m_y \end{bmatrix} \tag{3.3}$$

where l is the length of the servo arm.

Finally, we can use a distance constraint to solve for β, since we know that the distance between the rudder pivot point and the servo arm pivot point must stay constant with length d.

$$d^2 = (x_2 - x_3)^2 + (y_2 - y_3)^2 \tag{3.4}$$

$$d^2 = (x_2 - s\sin\beta - m_x)^2 + (y_2 - s\cos\beta - m_y)^2 \tag{3.5}$$

Solving for β, we obtain the expression

$$a = 4lx_2 - 4lm_x \tag{3.6}$$

$$b = d^2 - l^2 + 2lm_y - 2ly_2 - m_x^2 + 2m_x x_2 - m_2^2 + 2m_y y_2 - x_2^2 - y_2^2 \tag{3.7}$$

$$c = d^2 - l^2 - 2lm_y + 2ly_2 - m_x^2 + 2m_x x_2 - m_2^2 + 2m_y y_2 - x_2^2 - y_2^2 \tag{3.8}$$

$$\beta = 2\arctan\left(\frac{0.5\sqrt{a - 4bc} + 2lm_x - 2lx_2}{b}\right) \tag{3.9}$$

The design parameters for the jellyfish can be modified, but care must be taken to ensure that β stays positive as θ sweeps from 0 to $\pi/2$. A verification script that plots β for various α and θ values is included in the repository under tangent_function.m, and the Arduino implementation is included in jellyfish.ino. An example plot for values of α from 0 to $\pi/4$ is shown in Fig. 3.14. For all values of θ from 0 to π, β must be positive for the end effector position to be feasible.

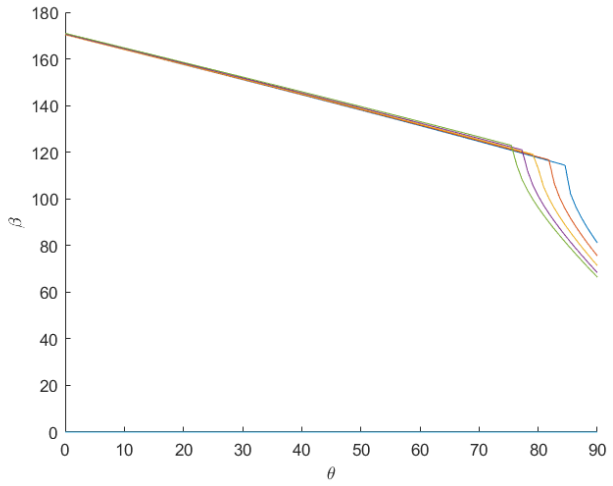

Figure 3.14: Values of β for θ with current design parameters

3.6 Results

We are able to successfully demonstrate BLOOPER deploying and balancing in an aquatic environment. BLOOPER is able to control its fins and rudders to within 0.1 degrees. Future work involves testing BLOOPER in a larger aquatic environment with different steering and control strategies.

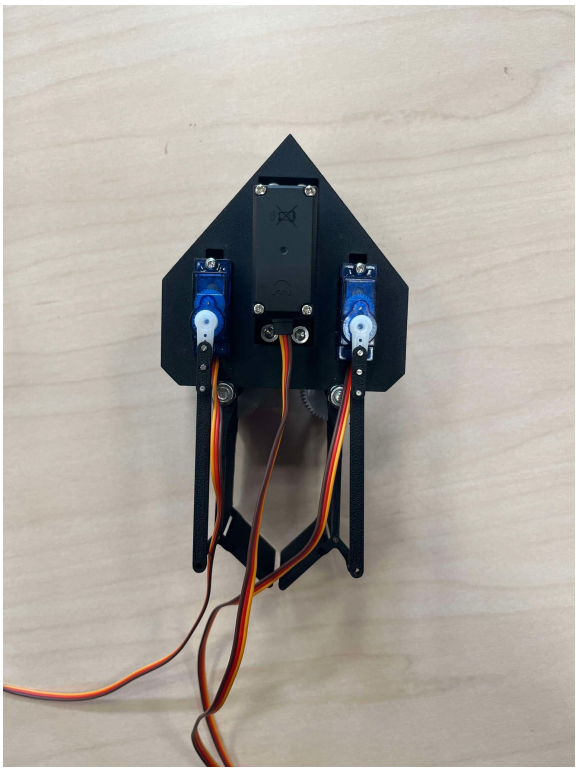

Figure 3.15: BLOOPER with fins at 0 degrees

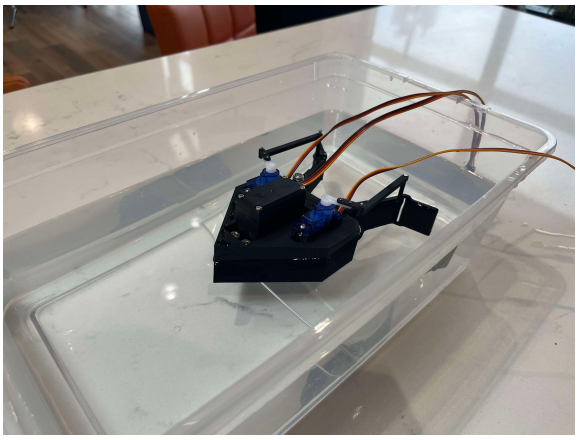

Figure 3.16: BLOOPER being tested for leaks and balance

Chapter 4

Conclusion

In this thesis, we introduce two low-cost robotic systems that fill gaps in robotics research hardware. We utilize the recent developments in FDM 3D printing to make systems that involve minimal amounts of precision machining and expensive, custom components. With this work, we hope to make testing novel control policies on hardware more accessible.

Research frequently creates more questions than answers. Both robotic systems can undoubtedly be improved and will likely go through many more design iterations as they are used to test different controllers. Future research directions for the biped system include implementing and testing dynamic locomotion, such as jumps or running, or testing contact-rich control methods. Future research directions for BLOOPER include extensive testing to ensure the design is water-resistant and tuning the mechanical parameters of the five-bar linkage system to make end-effector control more precise.

Bibliography

[1] Five-bar linkage kinematics. https://www.mathworks.com/help/sm/ug/five-bar-robot.html. Accessed 7/13/24. (document), 3.7

[2] Saab seaeye falcon. https://www.saabseaeye.com/solutions/underwater-vehicles/falcon. Accessed 7/13/24. (document), 3.2

[3] Simon R Anuszczyk and John O Dabiri. Electromechanical enhancement of live jellyfish for ocean exploration. *Bioinspiration & Biomimetics*, 19(2):026018, feb 2024. doi: 10.1088/1748-3190/ad277f. URL https://dx.doi.org/10.1088/1748-3190/ad277f. 3.1

[4] Gerardo Bledt, Matthew J. Powell, Benjamin Katz, Jared Di Carlo, Patrick M. Wensing, and Sangbae Kim. Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2245–2252, 2018. doi: 10.1109/IROS.2018.8593885. 2.1

[5] Travis L. Brown and James P. Schmiedeler. Energetic effects of reaction wheel actuation on underactuated biped robot walking. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2576–2581, 2014. doi: 10.1109/ICRA.2014.6907228. 2.2.2

[6] Travis L. Brown and James P. Schmiedeler. Reaction wheel actuation for improving planar biped walking efficiency. *IEEE Transactions on Robotics*, 32 (5):1290–1297, 2016. doi: 10.1109/TRO.2016.2593484. 2.2.2

[7] Justin Carpentier and Nicolas Mansard. Multicontact locomotion of legged robots. *IEEE Transactions on Robotics*, 34(6):1441–1460, 2018. doi: 10.1109/TRO.2018.2862902. 2.1

[8] Elham Daneshmand, Majid Khadiv, Felix Grimminger, and Ludovic Righetti. Variable horizon mpc with swing foot dynamics for bipedal walking control. *IEEE Robotics and Automation Letters*, 6(2):2349–2356, April 2021. ISSN 2377-3766. doi: 10.1109/LRA.2021.3061381. Publisher Copyright: © 2016 IEEE. 2.1

[9] Chelsey Edge, Sadman Sakib Enan, Michael Fulton, Jungseok Hong, Jiawei Mo, Kimberly Barthelemy, Hunter Bashaw, Berik Kallevig, Corey Knutson, Kevin

Orpen, and Junaed Sattar. Design and experiments with loco auv: A low cost open-source autonomous underwater vehicle. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1761–1768, 2020. doi: 10.1109/IROS45743.2020.9341007. (document), 3.2.1, 3.3

[10] Mohanarajah Gajamohan, Michael Merz, Igor Thommen, and Raffaello D'Andrea. The cubli: A cube that can jump up and balance. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3722–3727, 2012. doi: 10.1109/IROS.2012.6385896. (document), 2.2, 2.1

[11] Felix Grimminger, Avadesh Meduri, Majid Khadiv, Julian Viereck, Manuel Wüthrich, Maximilien Naveau, Vincent Berenz, Steve Heim, Felix Widmaier, Thomas Flayols, Jonathan Fiene, Alexander Badri-Spröwitz, and Ludovic Righetti. An open torque-controlled modular robot architecture for legged locomotion research. *IEEE Robotics and Automation Letters*, 5(2):3650–3657, 2020. doi: 10.1109/LRA.2020.2976639. (document), 2.1, 2.1

[12] Aaron M. Johnson, Thomas Libby, Evan Chang-Siu, Masayoshi Tomizuka, Robert J. Full, and Daniel E. Koditschek. Tail assisted dynamic self righting. *Adaptive Mobile Robotics*, pages 611–620, 2012. doi: https://doi.org/10.1142/9789814415958_0079. 2.2.1

[13] Nathan Kau, Aaron Schultz, Natalie Ferrante, and Patrick Slade. Stanford doggo: An open-source, quasi-direct-drive quadruped. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6309–6315, 2019. doi: 10.1109/ICRA.2019.8794436. 2.1

[14] Hendrik Kolvenbach, Elias Hampp, Patrick Barton, Radek Zenkl, and Marco Hutter. Towards jumping locomotion for quadruped robots on the moon. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5459–5466, 2019. doi: 10.1109/IROS40897.2019.8967552. 2.2.2

[15] Chi-Yen Lee, Shuo Yang, Benjamin Bokser, and Zachary Manchester. Enhanced balance for legged robots using reaction wheels. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9980–9987, 2023. doi: 10.1109/ICRA48891.2023.10160833. (document), 2.5, 2.2.2

[16] Jeong Hun Lee, Mike Y. Michelis, Robert Katzschmann, and Zachary Manchester. Aquarium: A fully differentiable fluid-structure interaction solver for robotics applications. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11272–11279, 2023. doi: 10.1109/ICRA48891.2023.10161494. (document), 3.1

[17] Thomas Libby, Aaron M. Johnson, Evan Chang-Siu, Robert J. Full, and Daniel E. Koditschek. Comparative design, scaling, and control of appendages for inertial reorientation. *IEEE Transactions on Robotics*, 32(6):1380–1398, 2016. doi:

10.1109/TRO.2016.2597316. 2.2.1

[18] Richard Montgomery. Gauge theory of the falling cat. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume Fields Inst. Commun., pages 5506–5511, 1993. doi: https://doi.org/10.1090/fic/001/09. 2.1

[19] Jongwon Park, Jinyi Lee, Jinwoo Lee, Kyung-Soo Kim, and Soohyun Kim. Raptor: Fast bipedal running and active tail stabilization. In *2014 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 215–215, 2014. doi: 10.1109/URAI.2014.7057424. 2.2.1

[20] Amir Patel and M. Braae. Rapid turning at high-speed: Inspirations from the cheetah's tail. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5506–5511, 2013. doi: 10.1109/IROS.2013.6697154. 2.1

[21] M. Popovic, A. Hofmann, and H. Herr. Angular momentum regulation during human walking: biomechanics and control. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 3, pages 2405–2411 Vol.3, 2004. doi: 10.1109/ROBOT.2004.1307421. 2.1

[22] Marc H. Raibert and Ernest R. Tello. Legged robots that balance. *IEEE Expert*, 1(4):89–89, 1986. doi: 10.1109/MEX.1986.4307016. 2.1

[23] Colin Rodwell and Phanindra Tallapragada. Physics-informed reinforcement learning for motion control of a fish-like swimming robot. *Scientific Reports*, 13, 07 2023. doi: 10.1038/s41598-023-36399-4. 3.1

[24] Samuel E. Schoedel, Alexander J. Fuge, Bhaben Kalita, and Alexander Leonessa. Development of an Affordable and Modular 3D Printed Quadruped Robot. In *ASME International Mechanical Engineering Congress and Exposition*, volume Volume 4: Biomedical and Biotechnology; Design, Systems, and Complexity, page V004T06A017, 10 2022. doi: 10.1115/IMECE2022-95700. 2.1

[25] Pablo Valdivia y Alvarado, Stephanie Chin, Winston Larson, Anirban Mazumdar, and Kamal Youcef-Toumi. A soft body under-actuated approach to multi degree of freedom biomimetic robots: A stingray example. pages 473 – 478, 10 2010. doi: 10.1109/BIOROB.2010.5627803. (document), 3.2.1, 3.4

[26] Alex A Villanueva, Kenneth J Marut, Tyler Michael, and Shashank Priya. Biomimetic autonomous robot inspired by the cyanea capillata (cyro). *Bioinspiration & Biomimetics*, 8(4):046005, Oct 2013. doi: 10.1088/1748-3182/8/4/046005. URL https://pubmed.ncbi.nlm.nih.gov/24166747/. 3.2.2

[27] Tianlu Wang, Hyeong-Joon Joo, Shanyuan Song, Wenqi Hu, Christoph Keplinger, and Metin Sitti. A versatile jellyfish-like robotic platform for effective underwater propulsion and manipulation. *Science Advances*, 9(15):eadg0292, 2023. doi: 10.1126/sciadv.adg0292. URL https://www.science.org/doi/abs/10.1126/sciadv.adg0292. (document), 3.5, 3.2.2