ARTIFICIAL NEURAL NETWORK-BASED ROBOTICS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Justin Ng

June 2018

COMMITTEE MEMBERSHIP

TITLE:	Artificial Neural Network-Based Robotics
AUTHOR:	Justin Ng
DATE SUBMITTED:	June 2018
COMMITTEE CHAIR:	Andrew Danowitz, Ph.D. Assistant Professor of Electrical Engineering
COMMITTEE MEMBER:	Xiao-Hua Yu, Ph.D. Professor of Electrical Engineering
COMMITTEE MEMBER:	Fred W. DePiero, Ph.D. Professor of Electrical Engineering

ABSTRACT

Artificial Neural Network-Based Robotics

Justin Ng

Artificial neural networks (ANNs) are highly-capable alternatives to traditional problem solving schemes due to their ability to solve non-linear systems with a non-algorithmic approach. The applications of ANNs range from process control to pattern recognition and, with increasing importance, robotics. This paper demonstrates continuous control of a robot using an actor-critic algorithm based on deep deterministic policy gradients (DDPG) originally conceived by Google DeepMind. The robot performs tasks such as locomotion within an enclosed area and object transportation. The paper also details the robot design process and explores the challenges of implementation in a real-time system.

ACKNOWLEDGMENTS

Thanks to:

- Everyone for everything.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Note: All project files can be found at `https://www.github.com/okayjustin/roborodentia2017`

1.1 Cal Poly Roborodentia

The robot is designed to compete in the 2018 Cal Poly Roborodentia, the university's annual intramural robotics competition, and thus conforms to its particular specifications and requirements. Briefly, competitors must produce autonomous robots to collect and fire Nerf Rival Balls into nets to win points. A drawing of the field is shown below in Figure 1.1. Two robots compete separately in each half so the effective field is a 4' wide by 8' long area enclosed by 4" walls. 1 inch PVC tubes along the 4' walls hold the balls which the robots fire into rectangular nets located along the 8' wall. The rules provide additional restrictions on robot dimensions, capabilities, and other aspects, to be covered specifically in the following chapters.

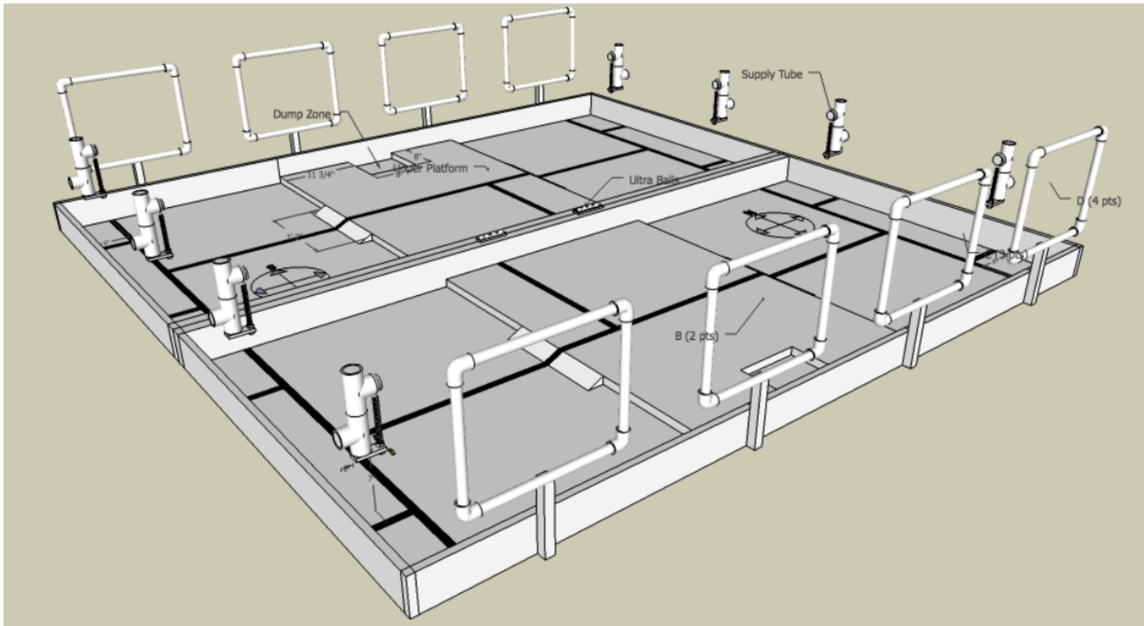

Figure 1.1: Roborodentia Field [3]

Chapter 2

MECHANICAL DESIGN

2.1 Introduction

The robot meets the design specification shown in Table 2.1. It consists of four subassemblies: the base platform, shooting mechanism, ball hopper, and control unit. Each section was first modeled in SolidWorks, an industry-standard solid modeling CAD program. The designed parts were then fabricated using a laser cutter or 3D printer and assembled with metric hardware. Figures 2.1 through 2.4 show standard view renders of the robot. Note that the robot uses mecanum wheels (a type of omni-directional wheel) which are modeled here as regular wheels for simplicity.

Table 2.1: Roborodentia 2018 Mechanical Requirements

	Requirement
1	Maximum footprint of 12” x 14” or smaller at start of match but may expand up to 14” x 17” during match.
2	Maximum height of 14” at start of match but no restriction during match.
3	Robot may not disassemble into multiple parts.
4	Robot may not be airborne.
5	Shooting mechanisms may not accelerate balls past 50 feet per second.

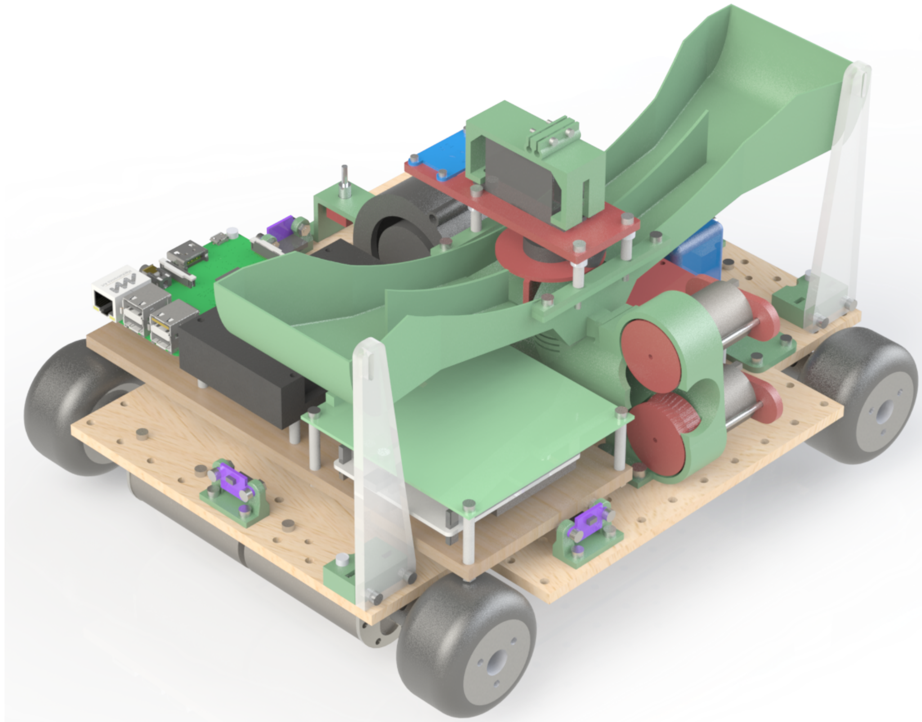

Figure 2.1: Full Robot Render – Isometric View

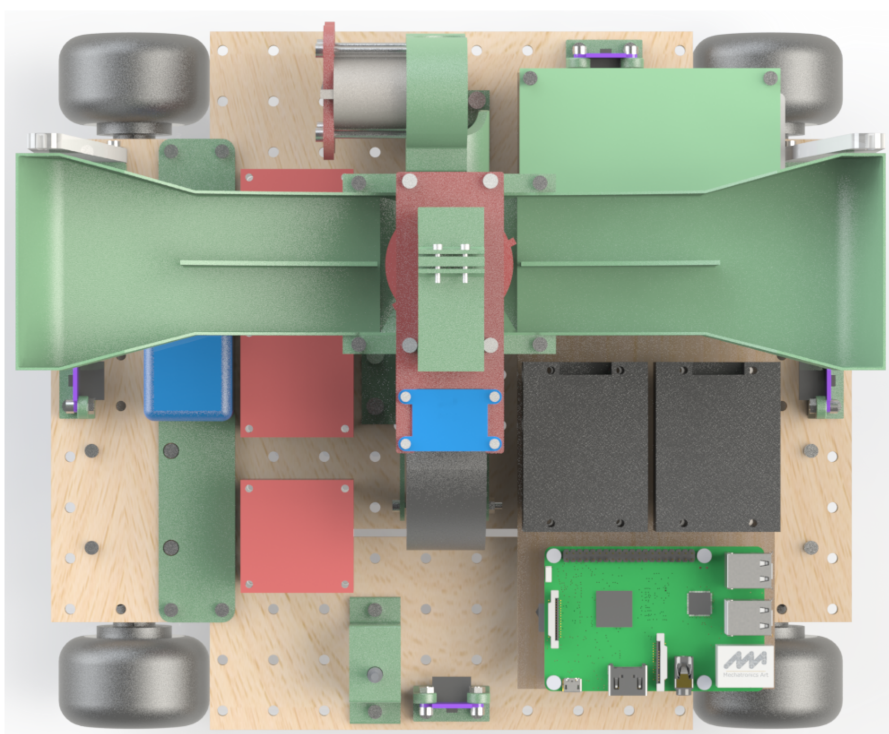

Figure 2.2: Full Robot Render – Top View

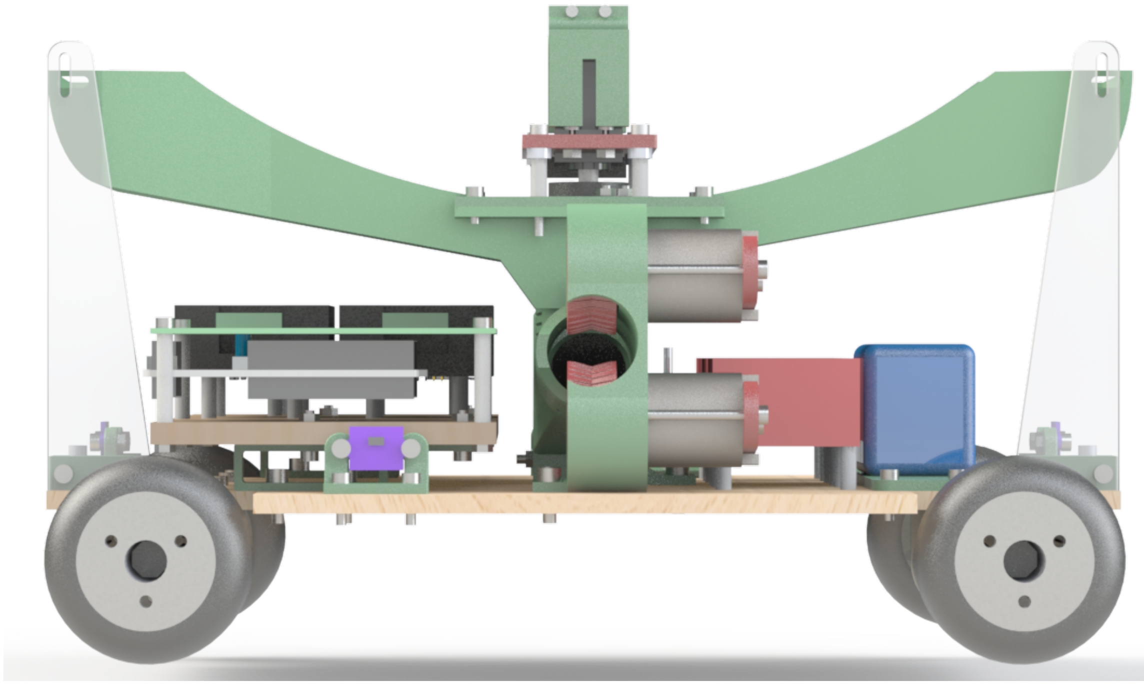

Figure 2.3: Full Robot Render – Front View

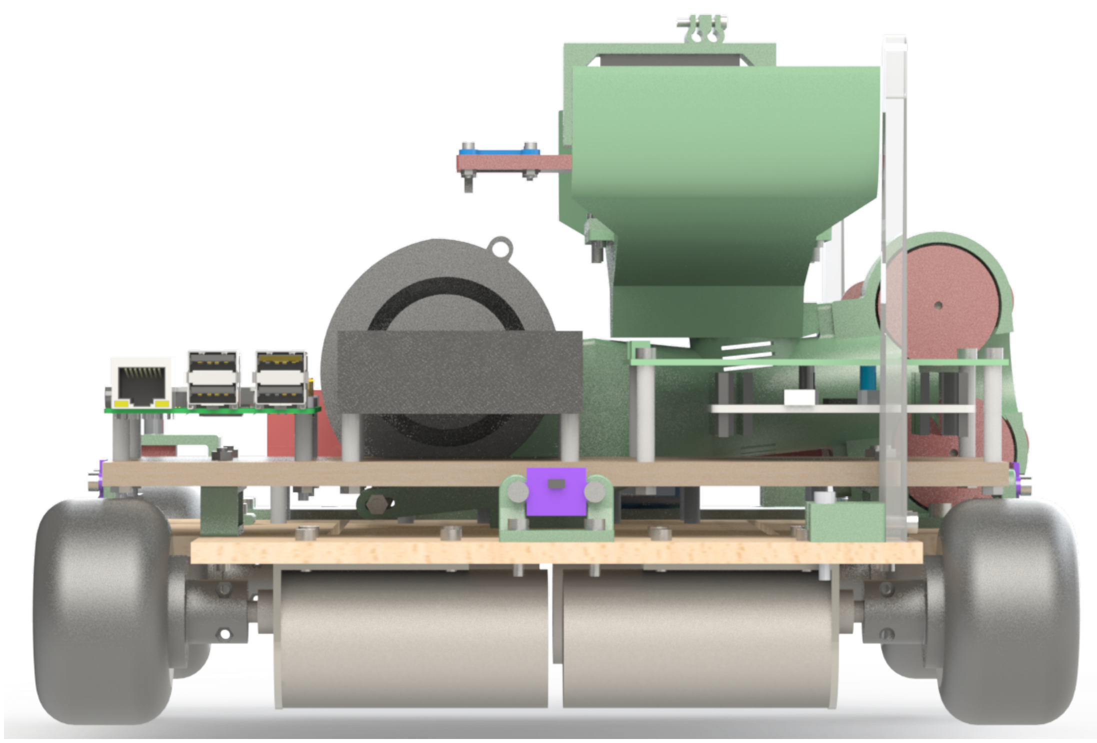

Figure 2.4: Full Robot Render – Right View

2.2 Base Platform

The base platform of the robot, made from 1/4" medium density fiberboard (MDF), serves as the primary structural component and a mounting point for the motors, electronics, shooting mechanism, and hopper. The wood is laser cut with a 20 mm grid of 4.5 mm holes to allow modular placement of components and the corners are removed to allow clearance for the wheels. Figures 2.5 shows the assembled view of the subassembly.

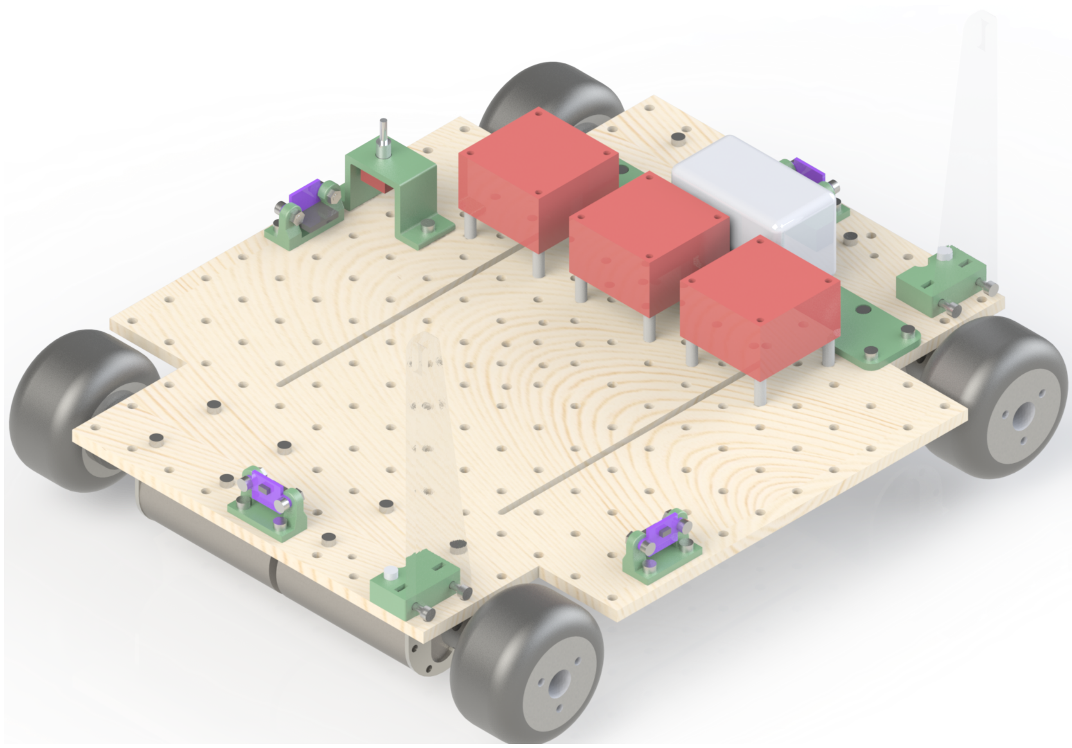

Figure 2.5: Base Platform

Four 12V Pololu 37D motors geared at a 70:1 ratio drive each of the 60 mm mecanum wheels. Each wheel contains eight angled rollers so unlike regular wheels which only produce a force vector perpendicular to the axis, mecanum wheels also produce a vector parallel to the axis. With the appropriate combination of speed and direction of each wheel, the robot can achieve simultaneous translation and rotation in any direction.

Several electronic components are mounted directly on the base platform. Along the four edges of the platform, four ST Microelectronics VL53L0X laser rangefinders mounted on 3D printed brackets sense distance. These sensors cost between $6 to $20 mounted on a small PCB with supporting circuitry and can sense distances between 30 mm and 2000 mm at a rate of 30 Hz and less than 10% error in most test conditions [9]. A 4S, 1200 mAH LiPo battery powers the system through a on/off toggle switch.

2.3 Shooting Mechanism

The shooting mechanism naturally takes inspiration from the official Nerf Rival Blaster toys since the manufacturer specifically optimized them to fire Nerf Rival balls in a way similar to baseball pitching machines. Figure 2.6 shows an exploded view of the subassembly while Figure 2.7 displays the top view. The mechanism consists of two sections: the **barrel** (left green part in Figure 2.6) and the **wheel housing** (right green part in Figure 2.6). Both parts were fabricated using a fused deposition modeling (FDM) 3D printer as the geometries are highly complex. Therefore, the shooting mechanism consists of two separate components versus a unibody design to allow each half to be fabricated with optimal print direction, strength, and finish quality. The barrel is angled 6° above horizontal, targeting the vertical center of the nets.

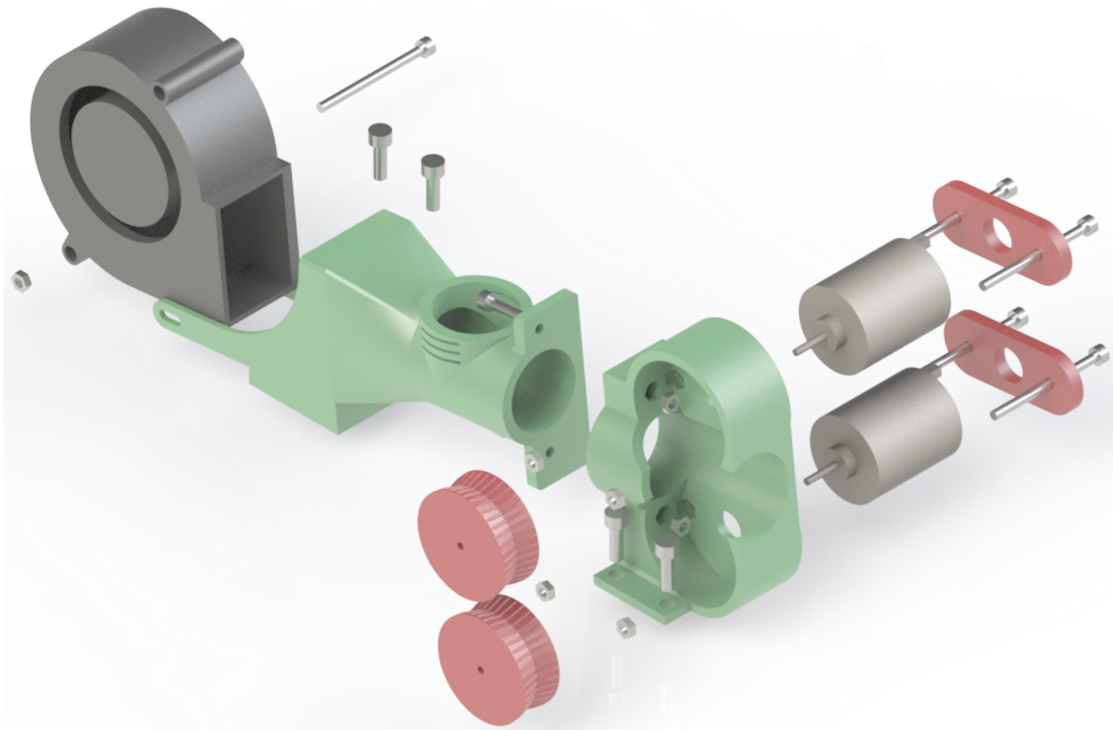

Figure 2.6: Shooting Mechanism – Exploded View

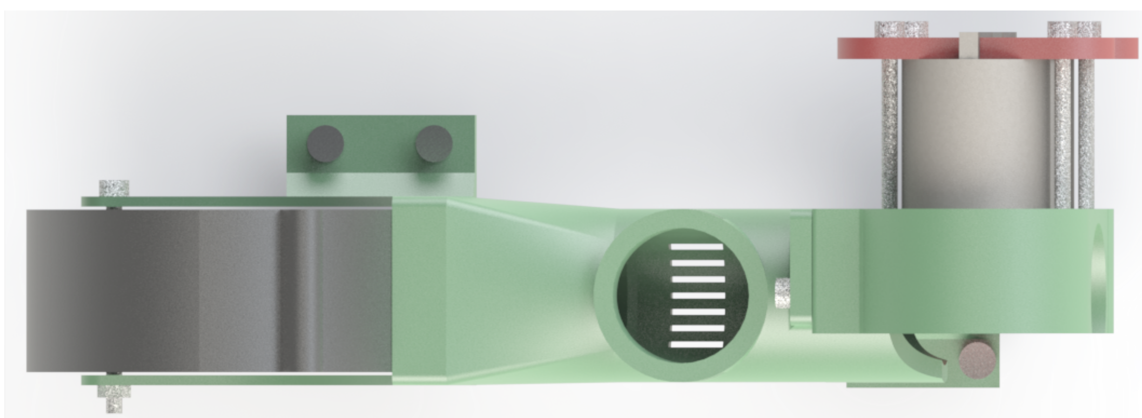

Figure 2.7: Shooting Mechanism – Top View

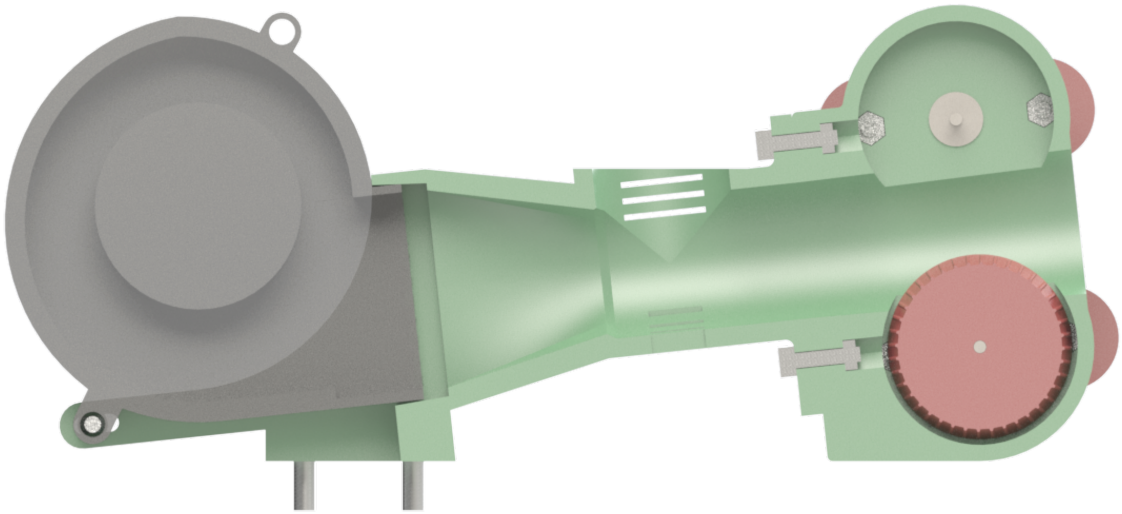

Figure 2.8: Shooting Mechanism – Cross Section View

The **barrel** directs balls from the **ball hopper** to the **wheel housing**. First, the ball enters the barrel through a vertical chute by force of gravity. As the ball falls into the barrel, a high-pressure centrifugal (or blower fan) attached at the back of the barrel pushes it into the wheel housing inlet. As seen in Figure 2.8, the barrel slightly narrows in the area behind the top chute to prevent the ball from rolling backwards towards the blower fan. A loft feature creates a smooth transition between the rectangular fan connection and the circular barrel. The foam balls, nominally 23 mm in diameter, would occasionally jam in a 24 mm barrel so all pathways are 25 mm. In the initial design, the pressure created by the blower fan was so high that it prevented the ball from falling down the vertical chute so strategically placed vents reduce the barrel pressure as the ball falls through the chute. As the ball travels down the chute into the barrel, it blocks the vents, increasing the pressure and forcing the ball into the wheel housing.

Inside the **wheel housing**, two counter-rotating 34mm wheels press fitted to two high-speed 12 V motors rapidly accelerate the foam ball up to 50 feet per second. The 14 mm gap between wheels compresses the ball to increase grip, thereby improving energy transfer. The motors lightly press fit into the wheel housing and are secured with 3D printed braces. The perimeter of each 3D printed wheel, detailed in Figure 2.9, consists of a ribbed V-groove to increase the contact patch and grip with the compressed foam ball. Two ”feet” with bolt holes at the bottom of the barrel and wheel housing secure the shooting mechanism to the base platform.

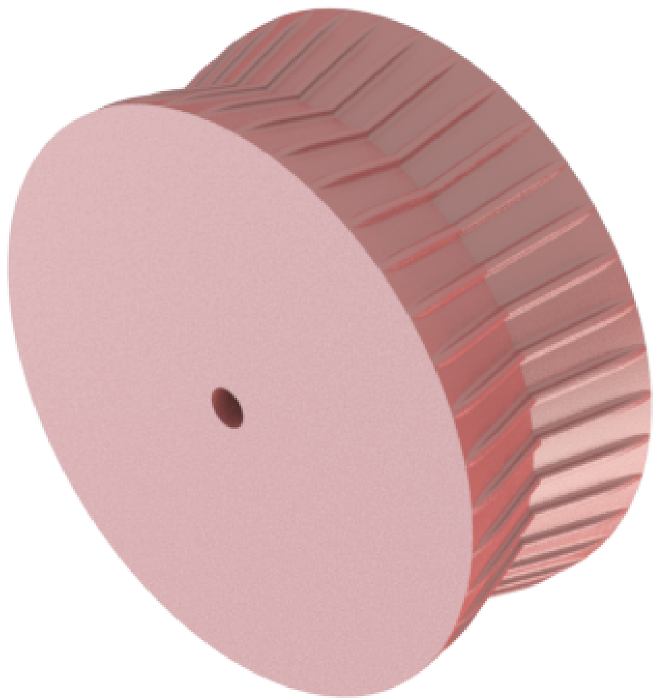

Figure 2.9: Shooting Mechanism – Shooter Wheel

2.4 Ball Hopper

The robot must obtain the foam balls from supply tubes mounted on two sides of the side. The bottoms of the supply tubes are positioned seven inches above the floor and a swinging flap holds the balls in. The ball hopper, shown in Figure 2.10 is a large 3D printed component designed to push the swinging flap away, collect the balls, store them, and dispense them into the shooting mechanism. Figure 2.11 shows an exploded view of the subassembly.

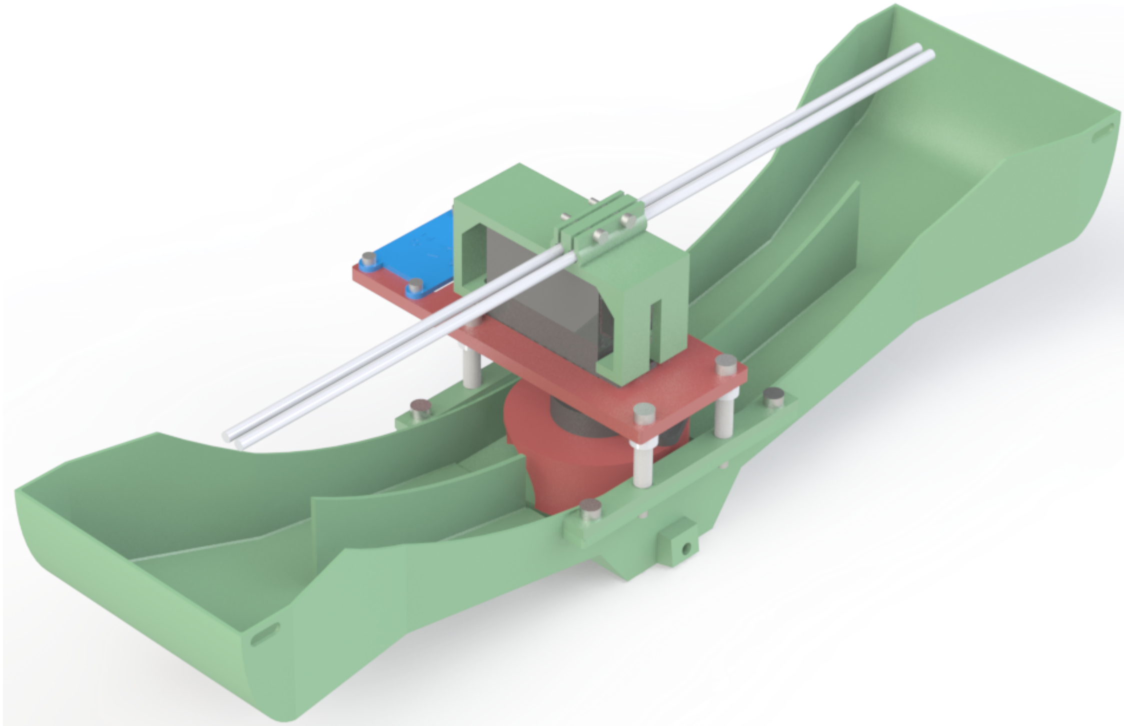

Figure 2.10: Ball Hopper

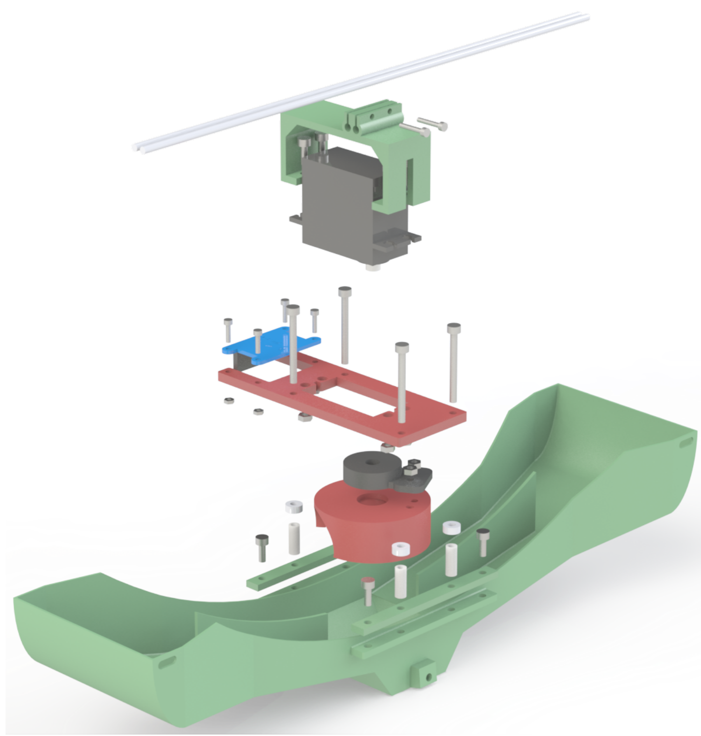

Figure 2.11: Ball Hopper – Exploded View

First, the robot moves the hopper underneath the supply tube. As the flap swings open, the balls rolls down the steep sloped portion of the hopper. Visible in the cross section view of Figure 2.12, The slope rapidly becomes less steep in order to transition the balls downward momentum into sideways momentum, keeping balls from jamming against each other. The balls then roll into one of two channels before stopping at the dispensing gate.

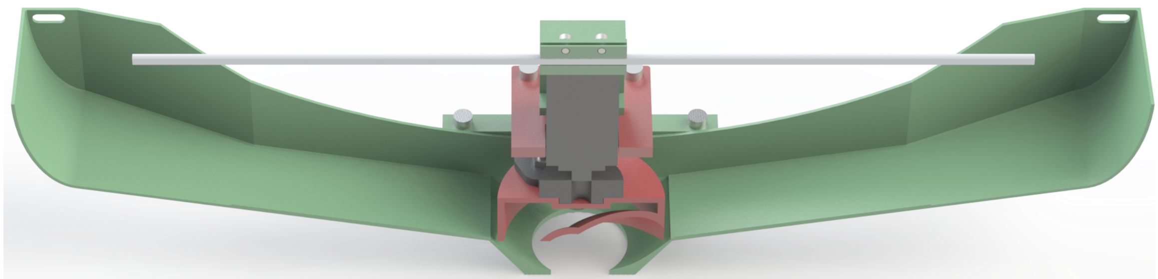

Figure 2.12: Ball Hopper – Cross Section View

The dispensing gate, shown in Figure 2.13, controls the movement of balls between hopper channel and shooting mechanism entrance. Its complex shape directs balls into the center of the ball hopper from one channel at a time to prevent jamming. A common 180° movement servo, mounted in a 3D printed bracket above the center of the hopper, controls the dispensing gate. Fastened to the same bracket, an inertial measurement unit (IMU) measures magnetic compass heading and acceleration in three dimensions.

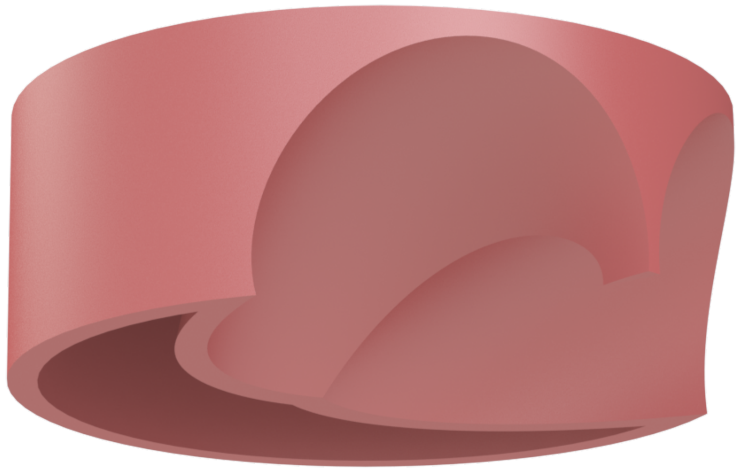

Figure 2.13: Ball Hopper – Dispensing Gate

The ball hopper is mounted at three points: the top of the shooting mechanism and the left and right edges of the robot using 3D printed and acrylic braces shown in Figure 2.14.

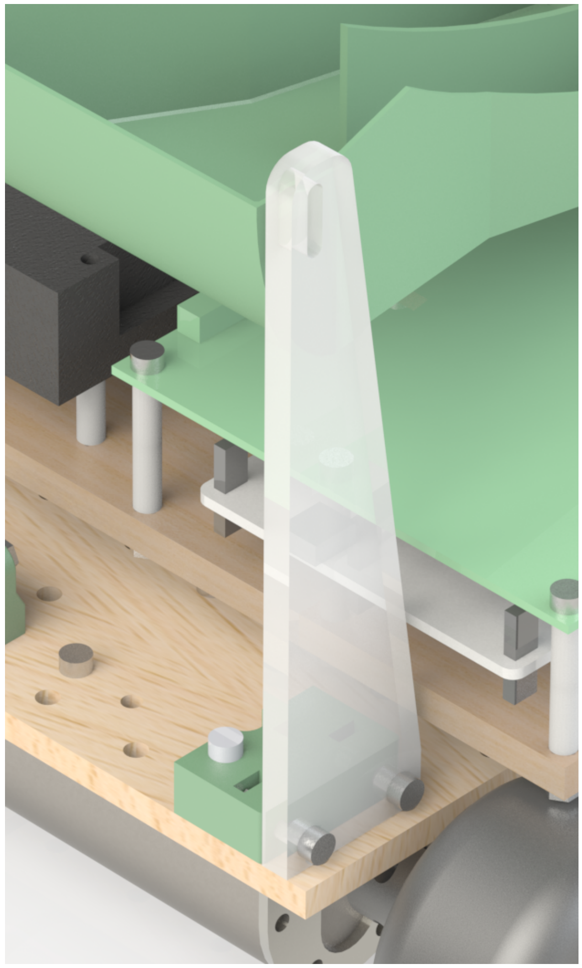

Figure 2.14: Ball Hopper – Braces

2.5 Control Unit

The control unit, shown in Figure 2.15, consists of a 1/4” MDF board with various electronic components mounted: two off-the-shelf DC-DC switching converters, a custom interconnect printed circuit board (PCB), an off-the-shelf STM32 Nucleo-64 development board, and the Raspberry Pi computer. Two 3D printed standoffs, the green parts shown in Figure 2.16, connect the control unit to the platform and raise it slightly to avoid colliding with the robot’s wheels.

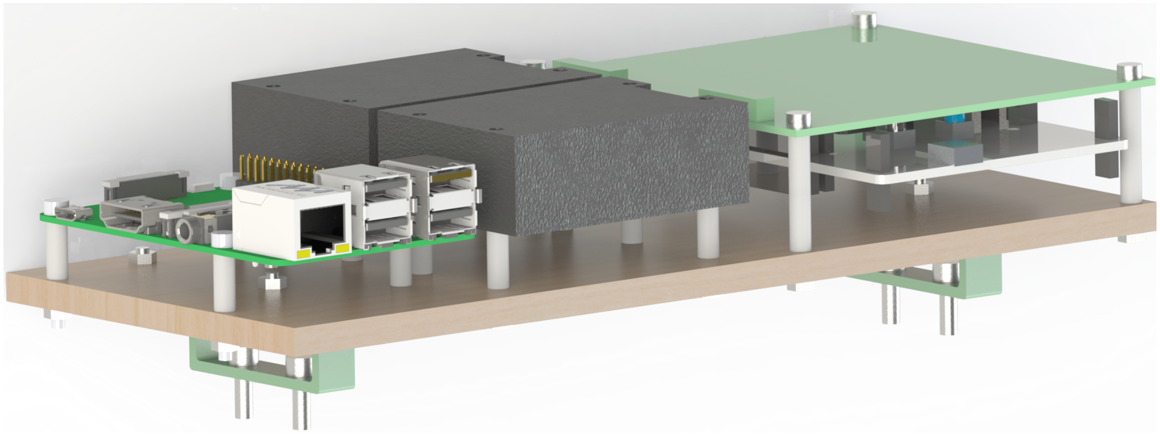

Figure 2.15: Control Unit

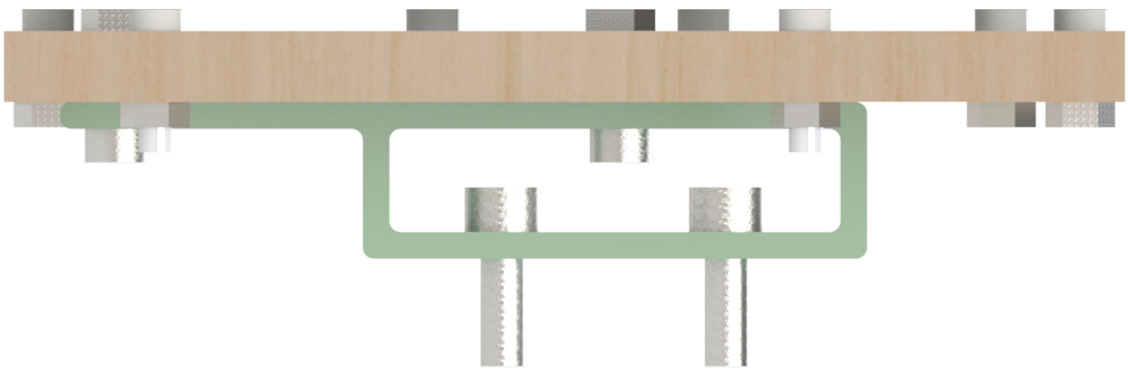

Figure 2.16: Control Unit – Standoffs

Chapter 3

ELECTRICAL DESIGN

3.1 Introduction

The electronics of the robot use a combination of off-the-shelf parts and custom designed circuits.

3.2 Power

A four cell, 1800 mAH lithium polymer (LiPo) battery powers the entire system. The battery is connected using polarized XT60 connectors to prevent reverse connection. The battery voltage varies between 16.8 V when fully charged and 14.8 V when depleted so two off-the-shelf DC-DC buck regulators buck battery voltage down to 12 V and 7 V supplies. The switching regulators accept a 7 – 40 V supply and can output 1.2 – 35 V at 8 A each. The 12 V bus powers the three motor drivers boards while the 7 V bus powers the STM32 Nucleo-64 development board and two AZ1085CD low-dropout linear regulators (LDO). One LDO produces a 5 V bus while the other provides 3.3 V, each at 3 A.

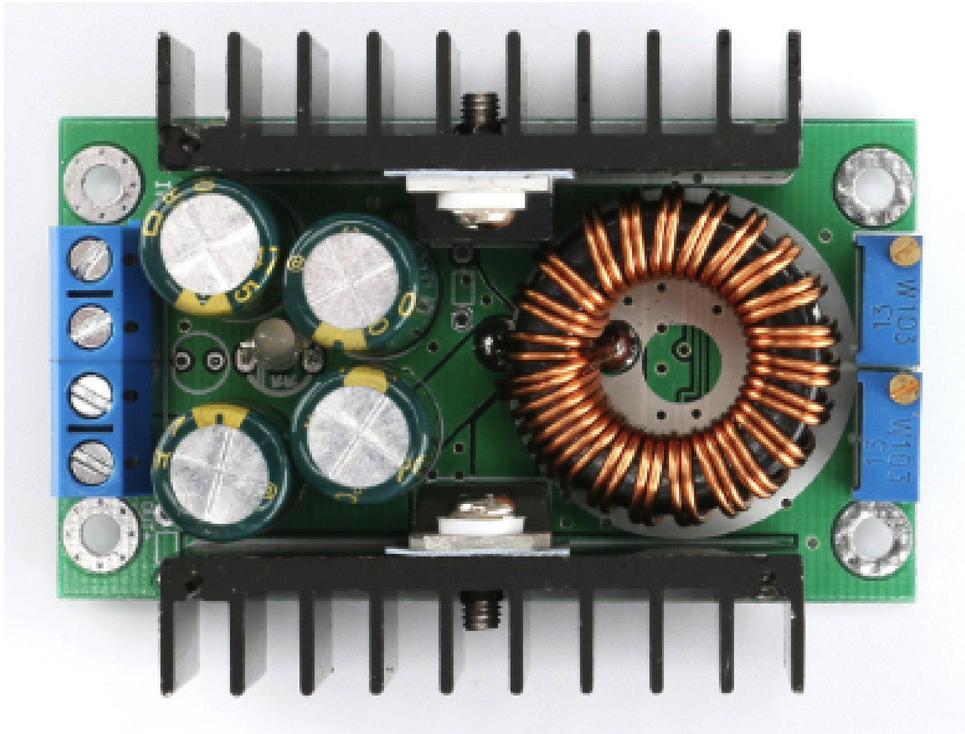

Figure 3.1: DC-DC Buck Regulator [2]

3.3 Sensors

3.4 Motor Drivers

The system uses three off-the-shelf L298N motor driver boards since they are easily obtainable for less than $6 each and incorporate features such as heat-sinking, flyback voltage protection, supply filtering, and screw terminal connections. Implementing comparable motor drivers with a similar feature set would undoubtedly cost more. Each L298N is a dual H-bridge driver with 2 A maximum output per bridge using a 5 – 35 V supply. Two motor drivers handle the four robot drive motors while the third powers the blower fan and shooting mechanism motors.

Figure 3.2 shows a wiring diagram for each motor driver. The board uses four digital control inputs, each controlling the state of one half-bridge. Each motor uses a pair of inputs: IN1 and IN2 control one motor while IN3 and IN4 control the other. To achieve direction and speed control, IN1 and IN3 are pulse width modulated (PWM) while IN2 and IN4 are digitally set. Table 3.1 is a truth table of the motor state versus inputs.

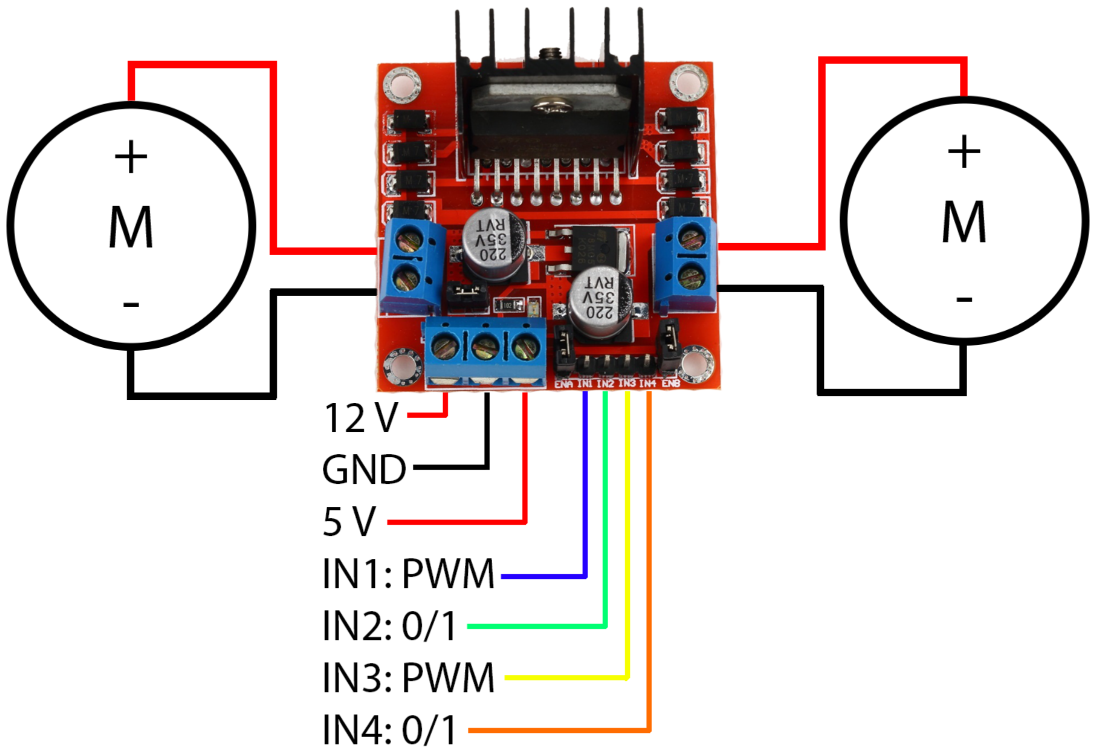

Figure 3.2: L298N Motor Driver Wiring Diagram [6]

Table 3.1: Motor Control Truth Table

IN1/IN3 Duty Cycle	IN2/IN4 State	Motor State
0%	0	Stopped
>0%	0	Forward, speed increases with duty cycle
<100%	1	Reverse, speed decreases with duty cycle
100%	1	Stopped

3.5 Interconnect PCB

3.5.1 Schematic Capture

3.5.2 Board Layout

3.5.3 Assembly

3.5.4 Reworks

Chapter 4

FIRMWARE DESIGN

Chapter 5

SOFTWARE

5.1 Definitions and Assumptions

The robot is designed to only travel within a rectangular closed area of eight feet in the x direction and five feet in the y direction. The coordinate system is chosen as Cartesian with the origin placed at the bottom-left corner of the field. The position of the robot is always in the xy-plane since it cannot move vertically (z = 0). Therefore, x refers to the robot position along the x-axis and ranges from 0 to 8 feet, and y refers to position along the y-axis and ranges from 0 to 5 feet. Additionally, the robot can only rotate around the z-axis so θ refers to the angle of the robot in the xy-plane. Maintaining standard Cartesian coordinates, $\theta = 0°$ is along the positive x direction while $\theta = 90°$ is along the positive y direction.

5.2 Kalman Filters

The system relies on several different sensors to determine where it is within the environment, a problem commonly referred to as robot localization. The *Kalman Filter* (KF), an optimal state estimator, performs noise filtering and sensor fusion, the process of combining measurements from multiple sensors. The filter operates on the principles of Bayesian inference and uses statistically noisy measurements over time and knowledge of the system to produce a more accurate estimate of an unknown variable than with measurement alone.

5.2.1 Algorithm

The Kalman Filter algorithm and equations are reproduced here from Roger Labbe's excellent interactive online book [5]. The algorithm consists of two stages (not including initialization): prediction and update. During the first stage, the filter uses the current state and a process model (typically a function of time) to estimate the state in the next time step along with its uncertainty. The second stage uses sensor measurements to update the estimation by taking a weighted average based on the ratio of uncertainty between the prediction and measurement.

Initialization

Before the first run of the filter, initialize the estimated state ($\mathbf{x}$, also called the posterior) and estimated state covariance matrix ($\mathbf{P}$).

Predict

During the predict phase, the process model is used to predict the future state (known as the prior) ($\bar{\mathbf{x}}$) after one time step by summing the posterior ($\mathbf{x}$) multiplied by the *state transition function* ($\mathbf{F}$) with the control input model ($\mathbf{B}$) multiplied by the control input ($\mathbf{u}$). The covariance of prior ($\bar{\mathbf{P}}$) is larger than the posterior covariance ($\mathbf{P}$) due to uncertainty in the process model ($\mathbf{Q}$).

$$\bar{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\bar{\mathbf{P}} = \mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{Q}$$

Update

Make measurements ($\mathbf{z}$, measurement mean) and determine their accuracy ($\mathbf{R}$, measurement noise covariance). Calculate the residual (or difference) ($\mathbf{y}$) between the measurement and the product of the measurement function ($\mathbf{H}$) and the prior from the previous phase. $\mathbf{H}$ converts the prior from the state space to the measurement space. Calculate the weighting factor ($\mathbf{K}$, Kalman gain), valued between 0 and 1, based on the whether the measurement or prior is more accurate. Set the new posterior, $\mathbf{x}$, to an average of the measurement and prior, weighted by $\mathbf{K}$. Finally, update the posterior's covariance, $\mathbf{P}$, based on the measurement certainty. The algorithm then loops back to the predict phase using the newly-calculated posterior.

$$\mathbf{y} = \mathbf{z} - \mathbf{H}\bar{\mathbf{x}}$$

$$\mathbf{K} = \bar{\mathbf{P}}\mathbf{H}^T(\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^T + \mathbf{R})^{-1}$$

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{K}\mathbf{y}$$

$$\mathbf{P} = (\mathbf{I} - \mathbf{K}\mathbf{H})\bar{\mathbf{P}}$$

5.2.2 Design

The desired state variable $\mathbf{x}$ is chosen as the linear position, velocity, and acceleration in the x and y directions as well as the angular position, velocity, and acceleration about the z-axis:

$$\mathbf{x} = [x\ y\ \theta]^T$$

$$\dot{\mathbf{x}} = [\dot{x}\ \dot{y}\ \dot{\theta}]^T$$

$$\ddot{\mathbf{x}} = [\ddot{x}\ \ddot{y}\ \ddot{\theta}]^T$$

The process model for position and velocity:

$$\begin{cases} \bar{x} = x + \dot{x}\Delta t + 0.5\ddot{x}(\Delta t)^2 \\ \bar{\dot{x}} = \dot{x} + \ddot{x}\Delta t \\ \bar{\ddot{x}} = \ddot{x} \end{cases}$$

Which can be written in the form:

$$\begin{bmatrix} \bar{x} \\ \bar{\dot{x}} \\ \bar{\ddot{x}} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0.5(\Delta t)^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}$$

$$\bar{\mathbf{x}} = \mathbf{F}\mathbf{x}$$

The measurement vector is chosen as:

$$\mathbf{z} = \begin{bmatrix} z_x & z_{\ddot{x}} & z_y & z_{\ddot{y}} & z_\theta & z_{\dot{\theta}} \end{bmatrix}^T$$

The measurement noise matrix is shown below. The off-diagonals are 0 because the noise between sensors is assumed to be uncorrelated.

$$\mathbf{R} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\ddot{x}}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\ddot{y}}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\dot{\theta}}^2 \end{bmatrix}$$

Chapter 6

NEURAL NETWORK DESIGN

Chapter 7

CONCLUSION

BIBLIOGRAPHY

[1] . . , 2018. [Online; accessed May 11, 2018].

[2] AliExpress. DC-DC CC CV Buck Converter. `https://www.aliexpress.com/item/DC-DC-CC-CV-Buck-Converter-Volt-Step-Down-12V-19V-24V-Car-Laptop-Power-Supply/32822603345.html?spm=2114.search0104.3.135.191f73dfqonWDU&ws_ab_test=searchweb0_0,searchweb201602_2_10152_10151_10065_10344_10130_10068_10324_10547_10342_10325_10546_10343_10340_10548_10341_10545_10696_10084_10083_10618_10307_10059_308_100031_10103_10624_10623_10622_10621_10620,searchweb201603_32,ppcSwitch_5&algo_expid=30a33cc9-3acf-4021-b514-6ba550085dd9-19&algo_pvid=30a33cc9-3acf-4021-b514-6ba550085dd9&priceBeautifyAB=0`, 2018. [Online; accessed May 11, 2018].

[3] Cal Poly. Roborodentia - Cal Poly, San Luis Obispo. `http://www.myurl.com`, 2018. [Online; accessed May 11, 2018].

[4] J. Doe. *The Book without Title*. Dummy Publisher, 2100.

[5] R. R. Labbe. Kalman and Bayesian Filters in Python, Sep 2017.

[6] MechaMan. Working with L298N DC Motor Driver. `http://fritzing.org/projects/working-with-l298n-dc-motor-driver`, 2018. [Online; accessed May 13, 2018].

[7] R. Negenborn. *Robot Localization and Kalman Filters*. PhD thesis, Sep 2003.

[8] D. H. Nguyen and B. Widrow. Neural Networks for Self-Learning Control Systems. *IEEE Control Systems Magazine*, Apr 1990.

[9] STMicroelectronics. VL53L0X Datasheet. http://www.st.com/content/ccc/resource/technical/document/datasheet/group3/b2/1e/33/77/c6/92/47/6b/DM00279086/files/DM00279086.pdf/jcr:content/translations/en.DM00279086.pdf, 2018. [Online; accessed May 12, 2018].