Final Report

Team F3: Optimus Dine

Nicholas Cheong
Vraj Shah
Pete Spofforth
Jack Sweet

Engineering 1282.01H
Ben Grier 3:00 PM
04/25/2022

Executive Summary

In this project, team F3 was tasked with creating an autonomous robot that performed simple restaurant tasks for Carmen's Diner. This robot had to fit within a 9'' x 9'' x 12'' box and be constructed without exceeding a budget of $160.00. The required tasks included reading and distinguishing between light colors, depositing a tray, sliding an order ticket, flipping a burger, switching an ice-cream lever, and pressing different buttons.

The team began the project by brainstorming and debating designs for the robot, a course strategy, and the different mechanisms the robot would include. There were three initial designs that the team created. The first design consisted of a triangular base driven by three Omni-wheels with one multipurpose arm. The second design included a square base, two powered wheels, one loose Omni-wheel, and multiple arms. The third design also included a square base and two wheels. However, this model also included two front skids. The team decided to combine concepts of both the second and third designs to create Optimus Dine.

With plenty of space on the robot's base, the robot included a separate arm for almost every task. The robot flipped the burger using a pronged servo motor attachment. The tray and ice-cream tasks were completed with a large lever mechanism that had a basket on top of it. On the back of the robot, a servo was used to lift a lever to slide the ticket. With these mechanisms, the robot was able to complete the course efficiently.

As the constructors built the robot, the coding team made initial framework functions for each task for an easy software-based approach. The robot was programmed to navigate the course accurately, correcting itself when needed. Each week of the project, further modifications were made to the robot build and programming to prepare for the performance tests and final competitions.

Table of Contents

List of Figures

List of Tables

1. Introduction

Carmen's diner can serve a delicious brunch, and with its new online ordering system, it is increasing in popularity by the day. Due to their increased demand, they have launched a new strategy, implementing automated vehicles to perform simple restaurant tasks. They want to remove the burden of simple tasks from the human staff, so that the human staff can interact more with customers and encourage their "Friends Eat Here" atmosphere.

As a result, they contracted the Ohio State Research and Development (OSURED) team to help them select a robot prototype from those produced by several companies, including this team's company. OSURED has developed a simulation for choosing an efficient prototype, creating a scenario using a scale model of Carmen's Diner, and implementing various tasks required of the robot [1].

1.1. Objectives

The purpose of the company's project was to build a self-propelled and robotic vehicle prototype that would be picked to work within Carmen's Diner. The robot was required to travel the scale model of Carmen's Diner, perform tasks that simulate tasks of the real-world diner, and return to its start position without any human assistance in an efficient manner.

Section 2 discusses the team's initial strategy for the course as well as the course layout, preliminary robot designs, and what the company initially decided after brainstorming. Section 3 describes the testing process and different performance tests that guided the team in making code and physical refinements for their final robot design. Section 4 explores the outline of the Individual Robot Competition and the improvements that were made after each competition. Section 5 highlights the team's final robot design and discusses both code and physical

refinements since the initial design. Section 6 outlines the Final Robot Competition, presents the team's final strategy, and analyzes the performance of the final robot design. Finally, Section 7 summarizes the overall decisions made for the team's final robot as well as a summary of its performance, concluding with potential improvements for future work.

2. Preliminary Concepts

Before beginning the building process, the team evaluated the requirements and constraints of the robot project to help the team generate ideas regarding course strategy and initial robot designs. These evaluations are explained thoroughly in the following sections. The optimal strategy for the course was to choose the path that saved the most time and eliminated risks. These risks included getting stuck on course obstacles, missing tasks, or not completing the course within the required time. The key components of the robot were the chassis, drivetrain, and the different mechanisms that were used to complete a combination of tasks. The tasks included sliding the ticket, depositing the tray, flipping the ice cream lever, clicking the correct jukebox button, and flipping the burger hot plate. Each of these components required brainstorming and discussion and was decided using decision matrices.

2.1. Requirements and Constraints

The requirements taken into consideration for the robot project included the tasks the the robot had to complete, the time the robot had to complete these tasks, and the size of the robot. To compete in the competitions at the end of the project, the robot could be no larger than 9’’x 9’’ and no taller than 12’’. This meant the robot’s chassis had to fit within a glass box and provide enough room to incorporate all the desired mechanisms. The robot also had to be self-propelled without interference from the team or wireless signals. The only signal the robot had to receive was the desired ice cream flavor and it had to do this through the RPS system. To do this, a QR code had to be mounted to the robot. The design of the course presented another requirement for the robot. The robot course consisted of two levels, a kitchen, and a lobby. Between these floors was a ramp, so to complete half the tasks, the robot had to be able to get up this ramp. This requirement was taken into deep consideration when evaluating the drivetrain for

the robot and is further explained in Section 3. Along with getting up the ramp, the robot would be required to complete the specific tasks previously mentioned. These requirements were taken into consideration when evaluating the mechanisms that the robot needed to include. Another constraint set in place included a three-wheel limit on the use of Omni-wheels and a 4-motor limit on drivetrain motors. Along with the physical requirements of the robot, there were conceptual requirements and constraints set in place for the team. The team was given a $160 limit for the entire robot project. This budget included everything purchased from the FEH store, other online stores, and materials used from the FEH workshop [3]. To prevent robots from taking too long on the course, each robot had only two minutes to complete the course. Therefore, the robot not only had to complete the tasks, but it had to complete them in a fast manner. These requirements and constraints were considered when brainstorming the course strategies, robot designs, and mechanisms described in the following sections.

2.2. Course Strategy

The course layout consisted of a start/end position and four tasks that were required of the robot prototype as shown in Figure 1. The optimal strategy for the course was to choose the path that saved the most time and eliminated risk.

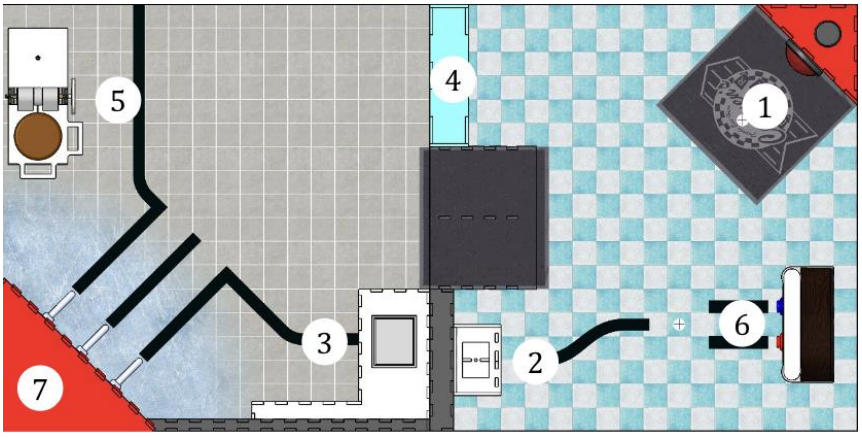

Figure 1: Course Layout

At its start position, on the lower level, the robot's proteus was initiated and after reading the red light, the robot began to complete the required tasks. Carrying a tray with it, the robot had two options, to return the tray to the trash can or deposit the tray in the sink on the second level. Also on the first level, the robot was required to select the correct song on the jukebox per the customer's request and slide a ticket across the rack to indicate to other workers that an order was in progress. Along with these first-floor tasks, the robot had to complete tasks in the kitchen as well. The robot had to flip a burger to ensure even cooking and flip an ice-cream lever based on the received flavor choice.

The team decided between two strategies for moving about the course. Both involved the robot initially going up the ramp, performing all the tasks before coming back down, and finishing with the jukebox. The team prioritized the kitchen tasks because that was where the most points were. If the robot failed on the first floor, most tasks would be missed if the robot could not correct itself and get to the kitchen. There was one key difference between the two-course strategies. As shown in Figure 2, the first strategy involved going directly to the sink before moving onto the ice cream lever.

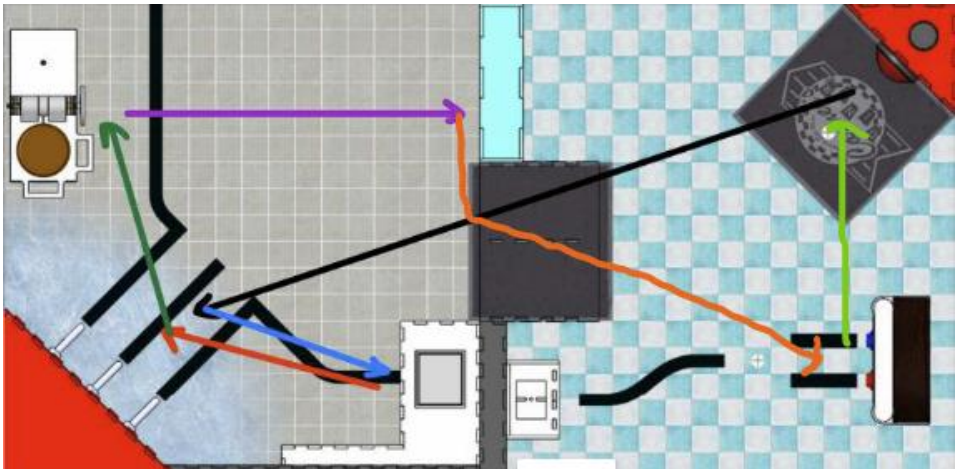

Figure 2: First Course Strategy

This strategy allowed the tray to be deposited first before any other task, minimizing the risk of the tray falling out.

The strategy shown below in Figure 3 was slightly riskier. This strategy involved hitting the ice cream lever first and then going to do the sink task while waiting seven seconds for the ice cream to finish.

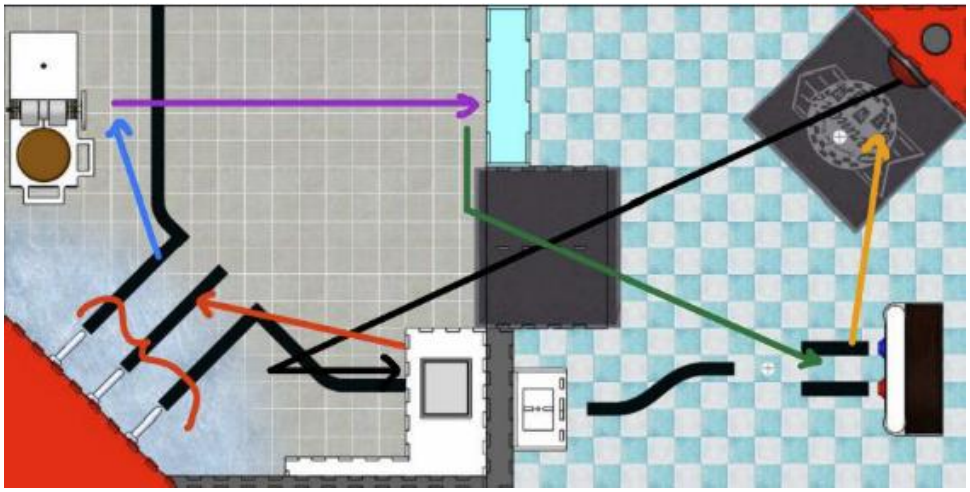

Figure 3: Second Course Strategy

In the end, the team evaluated the pros and cons of each strategy and determined that minimizing the risk of losing the tray was worth taking a few extra seconds. Overall, the completion of every task was the absolute priority.

2.3. Initial Designs

To guarantee the robot incorporated the best and most efficient mechanisms, chassis, and drivetrain, decision matrices were created to judge each idea on certain criteria. Selection criteria for the initial designs included cost, required space, weight, efficiency, multi-functionality, stability, accuracy, and more. These criteria and their weight were used to decide which mechanisms would work best. An example of a decision matrix can be seen on the next page in Table 1.

Table 1: Drive Train Decision Matrix

Drivetrain:

	Turning ability	Ramp	Speed	Coding Difficulty	Economic	total
Weight (1-3)	3	3	1	3	1	
3-wheel shopping cart	4	4	3	5	5	47
2 wheel with skids	3	2	1	5	5	36
Three Omni Wheels	5	4	4	1	2	36
4 wheel - shopping cart	2	3	2	4	3	32

In this decision matrix, the drivetrain ideas were being judged based on the criteria of turning ability, ramp capability, speed, coding difficulty, and affordability. Each of these was weighted on a scale of 1-3. The team believed the most important to be ramp-accessibility, turning ability, and coding difficulty. The robot's ramp and turning abilities were important as they were basic actions needed to complete the other tasks. Coding difficulty was also weighted highly so that a drivetrain with a simple coding structure would score better. The other options were scored lower because they did not impact the robot's ability to complete tasks. After creating the criteria, the team went through each design idea and discussed how it was scored. This process was used to judge each mechanism and the chassis. These decision matrices can be found in Appendix B. After creating these decision matrices, three initial designs were created that included features from each decision matrix. These designs are explained below.

2.3.1. First Design Concept

The first design concept was an Omni wheel-style drive train with a triangular base. The ability to move in any direction meant that only one arm was necessary to complete each task. This arm would have a fork shape that could be used to flip the burger and hold the tray. It would also be used to press the jukebox button. The only complication with this fork would be pressing the ice cream lever. During the initial brainstorming process, it was thought that the fork

could be rotated to hit the ice cream lever and then rotated back to flip it back. The overall design can be seen below in Figure 4.

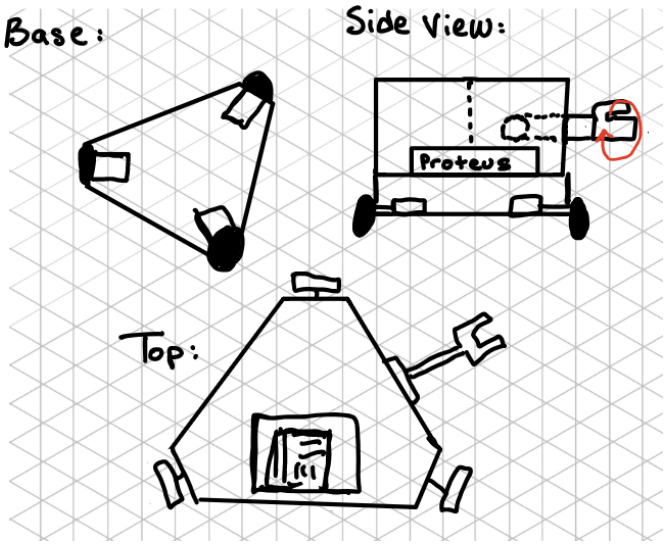

Figure 4: First Design Concept Sketch

Although this design seemed simple and easy, there were a few drawbacks. The triangular base reduced mobility due to the decreased space. Additionally, the three motors required to power the wheel would put a strain on the budget, and the complex Omni wheels meant that more effort would be required to program it. The awkward shape would also make it difficult to use bump switches. Without bump switches, the robot would be unable to use the course walls to straighten its heading.

2.3.2. Second Design Concept

The second design concept was a two-wheel drive train with a single Omni wheel in the front as shown in Figure 5.

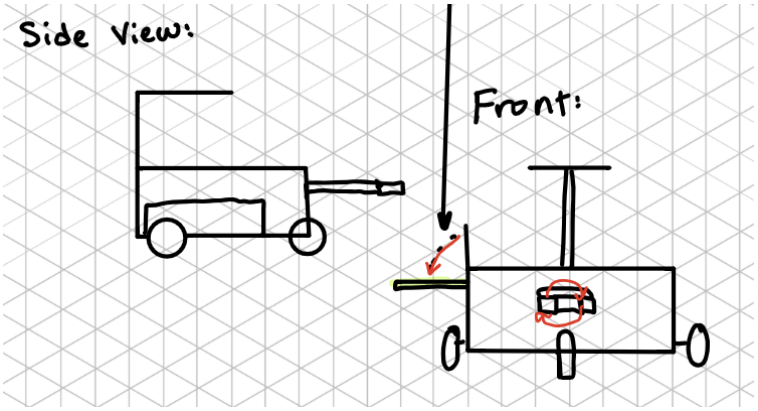

Figure 5: Second Design Concept Sketch

This design featured a square chassis with a cut-out in the front for an Omni wheel. While this design improved overall mobility by providing a box structure compared to the triangular structure of the first design concept with a box structure, the issue became the third wheel in the front. This wheel obstructed any front-facing robot arms and blocked potential bump switches on the front. For this design, the robot would require two forks, one in the front like the one described above in 2.3.1, and one on the side for the ticket mechanism. The front arm would flip the burger and drop it off the tray. The sidearm would be lowered when sliding the ticket and retracted at other times. It was also thought that the sidearm could be used to flip the ice-cream levers if the robot positioned itself correctly. This eliminated the possibility of the rotating fork being too low to do it.

2.3.3. Third Design Concept

The third design concept was a two-wheel drive train with skids in the front, a rotating front robot arm, and a ticket-sliding mechanism on the side, as shown in Figure 6.

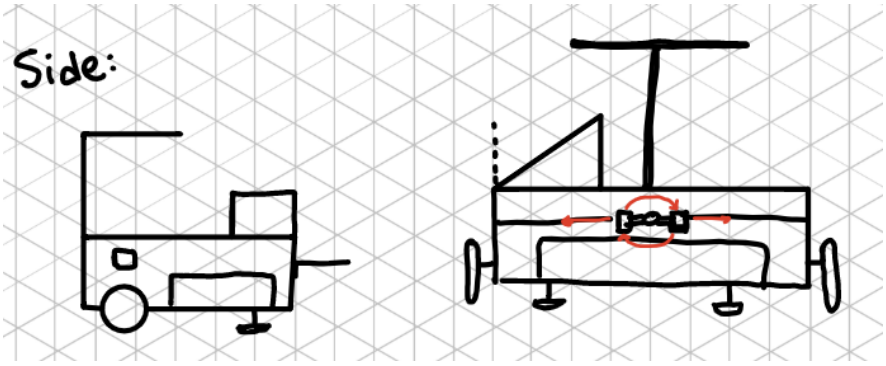

Figure 6: Third Design Concept

This design was effective through its simplicity. Without a front wheel and its large box design, there was plenty of room for bump switches and multiple moving arms to perform each task. Additionally, navigating and turning with two wheels was straightforward. Unfortunately, the speed and ability to drive up the ramp were a concern with this design because of the increased friction from the skids. In the figure above, there was a slanted board which was used to drop the tray off in its correct location. When in the right position, the wall was lowered, dropping the tray into the slot. The ticket-sliding mechanism would behave very similarly to the previous design where it would lower to slide the ticket and be raised at all other times. It would also be used to flip the ice cream lever.

2.4. Initial Design

After narrowing down the designs into three main builds, the team went with the highest-scoring design from the decision matrix: the triangular base with the Omni wheels. The prototype of this design can be seen on the following page in Figure 7.

Figure 7: Initial cardboard model

In this design, the chassis had a pentagon-like shape with two slanted wheels in the front and a perpendicular wheel in the back. In this model, there were two floors to the robot. The lower floor housed all the motors and their motor mounts. The second layer was where the proteus sat. A fork extended from the front of the robot to complete all the tasks. The only tasks that needed further attention were the ice-cream lever and ticket slider. To combat this, a lever was added to the robot that was lowered and raised by the winding and unwinding of the fork. As the front fork rotated, the rope wound around the fork's shaft pulling the lever down. This had the capabilities of lowering and raising the ice cream lever. On the side of the robot, there was a drop-down ticket fork. Controlled by a servo motor, the bar lowered to catch the ticket and raise to release it. Along with creating a physical prototype of the robot, a virtual model was created on SolidWorks. This model is shown on the following page in Figure 8.

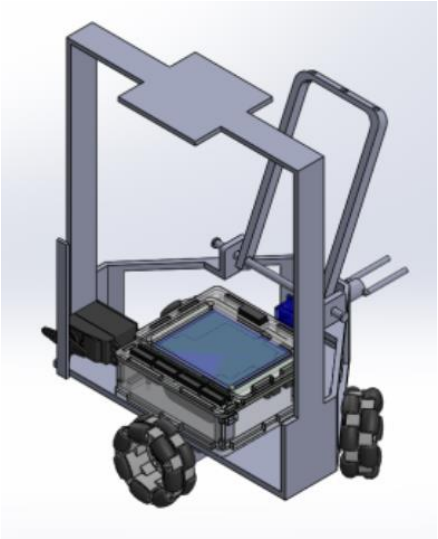

Figure 8: Solid works Mockup

This model is very similar to the cardboard model. However, it did not have two layers. In this model, the motors were mounted to the bottom of the base to save vertical space. The frame built around the proteus was supposed to allow for easy machine mounting. Originally, this robot model was believed to be the solution to the diner's growing popularity. However, its flaws quickly became prevalent.

2.5. Initial Coding Ideas

The team had a good idea from the beginning on how to organize the code on Visual Studio. The team used GitHub servers to save the robot code without getting lost. The team's initial coding ideas consisted of building frameworks for each task and adding different functions to move the robot using shaft encoding, RPS, bump switches, and alternative methods of navigation. The coding team made frameworks for each performance test from the very beginning to allow consistency and ease of access throughout the code. The frameworks also allowed the team to easily adjust numbers between each test to increase the testing rate. Before the framework was added for each task, the team made prototypes, with different variables passing in for each movement. This allowed the team to easily change distances for shaft encoding or RPS between

each test. To learn how the team's code changed between the initial ideas to the final code, see Section 5.4.

3. Analysis, Testing, and Refinements

The team made significant changes to the robot in terms of design and strategy throughout the building and testing process. Originally, the team planned to use a three-wheel, triangular robot but discovered flaws after constructing a prototype and mockup on SolidWorks. The lack of space on the chassis heavily restricted the ability to include key sensors such as the Omni wheels which blocked bump switches on the sides, and the string mechanism planned for the front lever was too much of a risk. When the front fork rotated, the string wound up but caught itself on the fork's prongs and tangle up. With this new information, the team adjusted the decision matrix by increasing the weight of the space category and decreasing the functionality score of the triangular chassis and the single-arm design. This decision matrix can be seen below in Table 2.

Table 2: Overall Design Decision Matrix

Overall Design:

	Completion of Tasks	Expense	Efficiency	Programmable	Accessible Parts	total
Weight	3	1	2	3	2	
-Triangular body -Omni Wheels -Static Fork, rotating arms, robot moves	4	3	5	3	4	42
Three-wheel robot with two motors, static fork with rotating arms, drop down claw for ticket slider	4	4	3	5	4	44
Two-wheel robot with skids and two motors, cube body, rotating and sliding claw, inclined tray surface with barrier that drops	4	4	2	4	2	36

In this matrix, the design models were judged based on task completion, expense, efficiency, programmability, and accessible parts. The completion of task criteria was a broad requirement. It entailed the robot's ability to complete each task with the mechanisms included in the design. The "efficiency" criteria entailed the robot's ability to complete the tasks without wasting much time. For example, to do the burger flip, a fork that used the turning wheel was optimal because it did not require the robot to reposition after flipping the burger. The "accessible parts" criteria were based on the openness of the robot's chassis and how accessible each feature of the robot would be for the constructors and coders. After readjusting the decision matrix, the highest-scoring design became the two-wheel design. However, the team replaced the skids with a ball caster to also improve the speed score of the two-wheel design. The caster ball removed the drag that the skids caused and did not crowd the robot's base. The two-wheel design was the initial design the team decided to pursue.

3.1. Calculations

To determine the proper motor to meet the performance requirements of the robot, the team conducted a drivetrain analysis. This analysis involved calculating the minimum speed required to travel the course within the time limit and involved finding enough torque to make it up the ramp. First, the total distance the robot had to travel was calculated, with the path and distances represented in Figure 9.

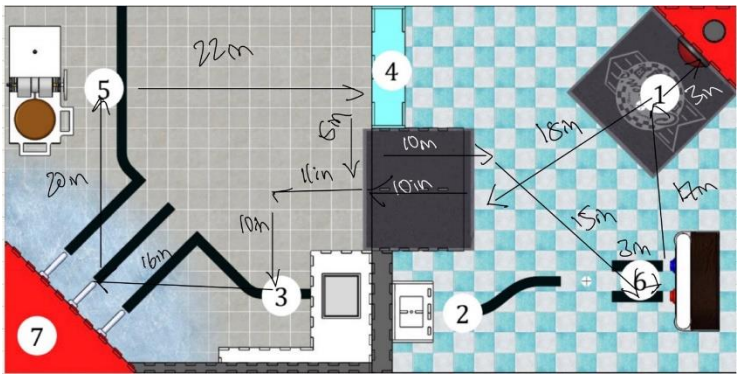

Figure 9: Distance of route through the course

The total distance the robot must travel was determined to be approximately 170 inches. After estimating and subtracting out the time it took for each task, the minimum speed for the robot to travel the 170 inches in the remaining time was determined to be 2.7 inches per second. Due to the 1.175-wheel radius, this measurement was equal to 21.95 rotations per minute, which gave the minimum speed requirement for the motor.

Next, the team estimated the weight of the robot and analyzed the ramp to determine the torque requirement for the motors. Using the torque equation found in Appendix D (1), the force was translated to torque. This value was divided between the two motors, and a minimum torque requirement for the motor was found.

The results were plotted on a graph containing the speed to torque relationship of each available motor as shown in Figure 10.

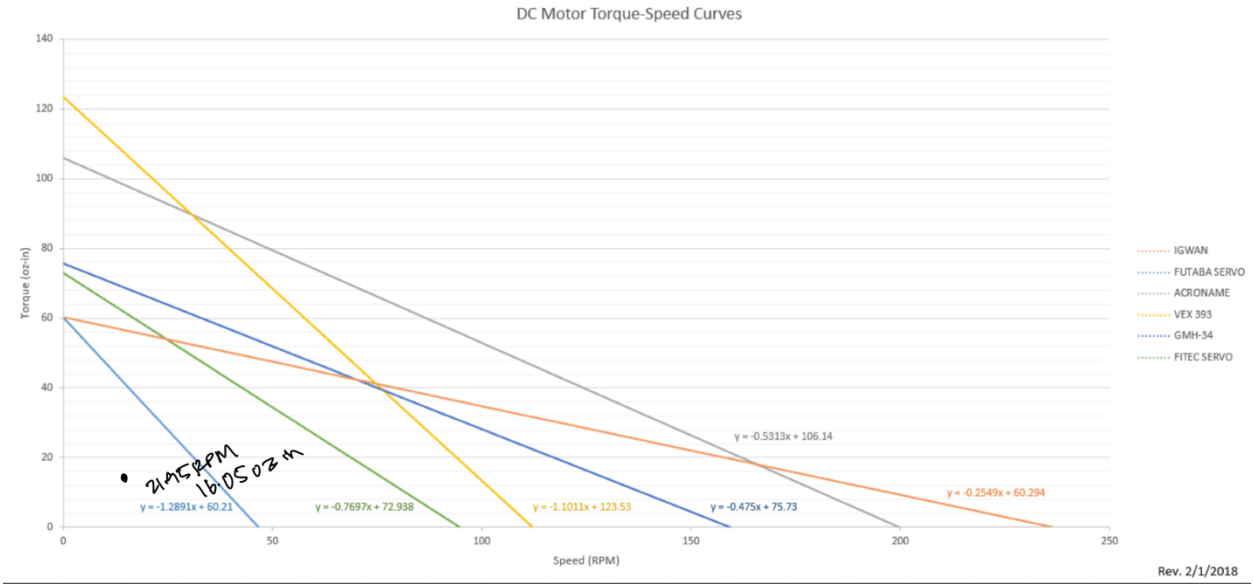

Figure 10: Speed/torque quantities for each motor

The minimum speed and torque needed for the robot are represented by the black dot in the lower-left corner of the figure. As each motor had superior speed and torque compared to what was required, every motor option was available. By determining additional criteria such as compatibility, functionality, size, and cost, the team decided on the Igwan motors due to their small size and built-in shaft encoding.

3.2. Testing Process

The team planned to test the robot by Tuesday or Wednesday of every week with each Performance test on Friday. This plan meant finalizing the construction by Monday/Tuesday morning and the code by Wednesday.

Having ample time to test the robot allowed the team to make the necessary changes for each run to be consistent and provided time to earn all the bonuses, including early success. To

evaluate the robot more efficiently and effectively, the team implemented a note-taking system and the use of testing logs.

During each trial, a team member took notes on what went well and what required improvement. An example of the testing notes can be found in Appendix A, Figure A1.

Furthermore, the team used testing logs for each trial, discussing what was being assessed and what the outcomes were. The team then evaluated these logs and discussed sources of error to make the necessary corrections.

3.2.1. Performance Test 1

In terms of hardware, the robot's chassis, drivetrain, CDS cell, and a jukebox button-presser had to be constructed and wired to complete the first performance test. MDF was used for the base because of its workability. The DUBRO wheels, IGWAN motors, and caster ball were mounted to complete the drivetrain, and the CDS cell reader was configured to read the start light.

A framework for the software was made before coding began. The team created flowcharts to understand each function's purpose before writing the code. The flowcharts for each function can be seen in Appendix C.

For the first performance test, the team coded the robot to begin moving after reading the start light, first driving to the jukebox. The robot was then coded to read the correct light color using the CDS cell and was provided the correct movements to press the button, ending with moving up the ramp. Exploration one, which explored the use of CdS cells, helped the team understand how to program the robot's CdS cell and correctly read the jukebox light. Depending on the light read by the CdS cell, different voltage values were sent to the Proteus. Using a red filter to cover the CdS funnel allowed the CdS cell to better differentiate between light colors [2].

The code then evaluated the voltages and performed the following task accordingly. All the maneuverings required for this test were done with IGWAN's shaft encoders. The team learned how to use the encoders in exploration two, so coding them was not an issue [3].

After original testing began, many changes were made to the robot's design as well as the code. First, the erector piece that was supposed to press the jukebox button was too small, always missing the buttons. Because of this, a small MDF square was glued to the front of it as a temporary fix. Another big issue was with the DUBRO wheel adaptors. They were constantly breaking, causing wobbly wheels, and unwanted directions. These issues made going up the ramp difficult. A solution was to wrap rubber bands around the DUBRO wheels. They increased the friction, improved the robot's driving accuracy, and guaranteed success by making it up the ramp.

The original build's strongest features were the wiring, caster ball, and CDS cell. All the soldering was done correctly and did not have to be redone, the caster ball allowed 360-degree movement and had no issues with the ramp, and the CDS cell with the filter distinguished between lights correctly. The robot used to complete Performance Test 1 can be seen below in Figure 11.

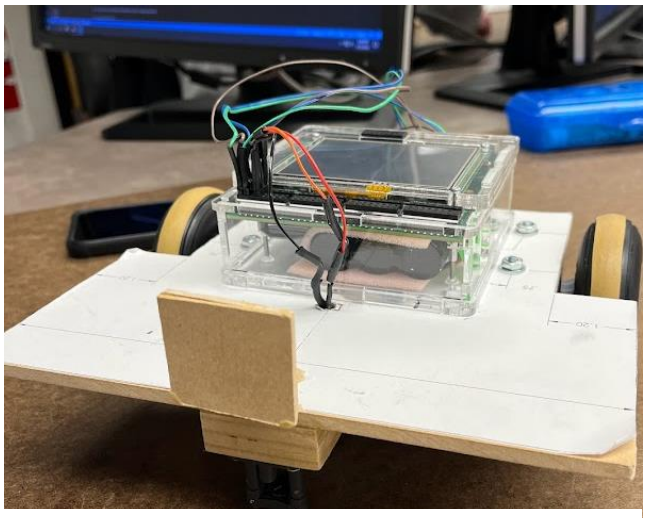

Figure 11: Performance Test 1 Robot

Software issues appeared as well. The team was first unable to program the CdS cell to read the correct light value. After debugging the code, the CdS cell values were printed to the Proteus screen, and the team was able to create a code for the robot to use the CdS cell values and perform functions associated with the specific light colors. As a result, the team was able to program the robot to start moving after reading the start light and clicking the correct jukebox button depending on whether it was red or blue.

3.2.2. Performance Test 2

Performance Test 2 included the completion of the ticket slider and the tray drop-off. To complete these, the team added two servo motors to the robot. To complete the ticket slider task, a handle-shaped bracket was added to provide an attachment location for the ticket mechanism. The ticket servo motor lifted a beam that grabbed the ticket and allowed the robot to pull it across the slider. To complete the tray task, or sink drop-off, a lever was attached to a servo motor with a tray-holder at the top. The servo motor controlled the lever allowing it to drop the tray when desired. The tray lever used a FITEC high-torque servo motor as the lever held a lot of weight. Without the high-torque servo motor, the motor could have broken or not have been able to move the lever with ease. Along with these specific task changes, the temporary button presser was removed and microswitches were attached to the back of the robot. The switches were not used for this test.

These changes were complete by Tuesday giving the programmers time to test. The robot was programmed to drive up the ramp, drop off the tray, back up to the ticket mechanism, slide the ticket, and then drive to the burger station. The team completed the performance test on

Wednesday and all extra credit was received. The final robot's build for performance test two can be seen below in Figure 12.

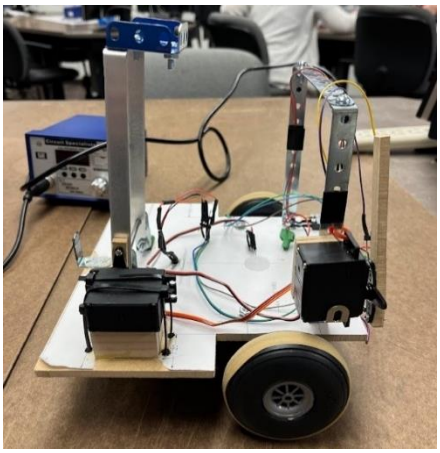

Figure 12: Performance test two robot

After testing began, changes to the original plan had to be made. Originally, the tray drop-off was supposed to occur when the robot was facing the sink head-on. However, there was not enough room for the robot in that location, so the programmers altered the programming to make the robot drop off the tray from the side of the sink. This ended up working better because after dropping off the tray, the robot could get to its position to slide the ticket in a couple of seconds.

Another issue that had to be accounted for was the weight distribution. Because both servo motors were added to the same side of the robot, the robot was having a challenging time making it up the ramp. To correct this issue, the coders updated the motor speeds that counteracted the weight distribution. One thing that worked well for this test was the ticket mechanism. Its simplicity made it easy to construct and easy to code. It got the ticket all the way across and guaranteed the bonus for leaving it in the final position. The servo motor also takes up no space on the robot's base which was helpful in future tests.

3.2.3. Performance Test 3

For Performance Test 3, the robot had to flip the burger. To accomplish this, another FUTABA servo motor was added to the front of the robot on top of the MDF spacers. Two prongs (screws wrapped with tape) were attached to the servo motor and the mechanism was complete. This was the only physical refinement made for this test and can be seen below in Figure 13.

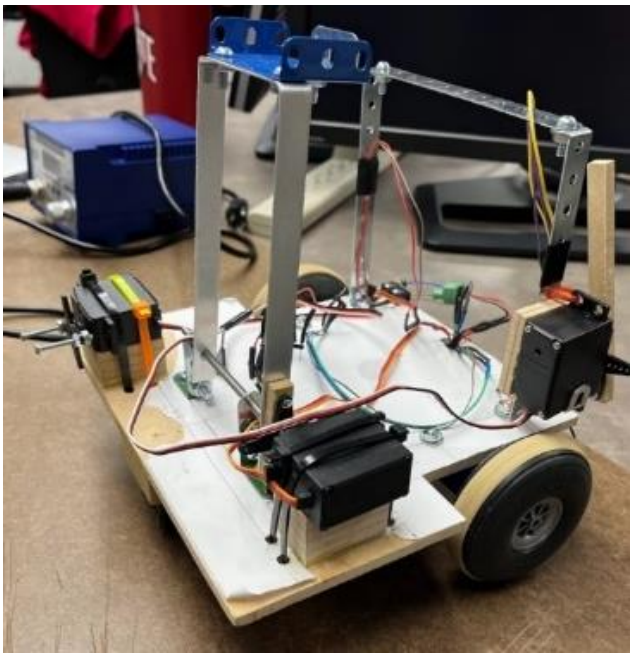

Figure 13: Performance test three robot

Overall, this test was easier for the team. The motor was added on Monday, and testing was complete by Tuesday. The burger flipping fork worked well and had no problems flipping the burger and returning the plate. The tray lever was also used to press the ice cream lever and had no issues. Adding the burger-flipping servo motor to the opposite side of the robot helped counteract the weight difference and made getting up the ramp easier. Another positive aspect of this design was the location of the tray lever and burger fork. They were positioned so that each of them could press a jukebox button. This eliminated the risk of inaccurate shaft encoding when

pressing the buttons. The microswitches came into use for this step because they allowed the robot to square up and align with the rotating burger wheel accurately. Once the robot was near the burger station, the robot backed into the wall, pressed its microswitches, and straightened out. For this test, the software aspects of performance test three consisted of the robot heading up the ramp, bumping the wall, flipping the burger, and then switching an ice-cream lever. It was completed on Wednesday and all bonuses were received.

3.2.4. Performance Test 4

For performance test four, the robot had to flip the correct ice cream lever. To do this, the team had to attach a QR code. To attach the QR code, erector brackets were mounted to the cross beam already present on the robot and the QR code was glued to the erector brackets. This gave the QR code the optimal height of around 9'' and kept it out of the way of the tray lever. With a height of 9'', the QR could be better picked up and read by the RPS system. The team's performance test 4 iteration of the robot can be seen below in Figure 14.

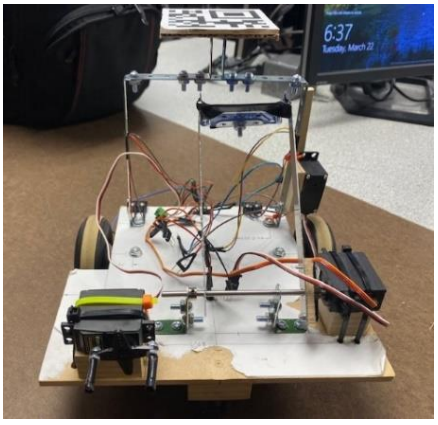

Figure 14: Performance Test 4 Robot

All other changes needed for this test were made within the code. A program was created that received the desired ice-cream lever and repositioned the robot accordingly all using RPS. Exploration three helped prepare the team to use RPS to move the robot to correct locations [4]. The ice-cream lever was then pressed with the robot's tray lever. During testing, the robot was

not performing consistently. This was due to the low battery of the proteus. The team had everything ready by Wednesday, and the test was completed earning all the bonuses.

4. Individual Competition

The individual competition took place on April 1st and was an in-class event in which each team had three chances to complete a full course run. For the first run, the course and conditions were randomly selected, in the second run, the course and conditions were selected by the instructors, and in the third run, the course and conditions were selected by the team. To be successful in the individual competition, the robot had to complete the necessary primary tasks in under two minutes and without any interference from the team. The primary tasks and their associated points are displayed in Appendix B, Table B1.

Completing these tasks was the priority for the team because they encompassed the most points for the robot. However, there were also secondary points available that determined placement for the final competition. These are displayed in Appendix B, Table B2.

The individual competition was used to seed the robot for the final competition. The seed was based on the score of our robot, both primary and secondary points, compared to the scores of the other robots. If a tie occurred, the overall time to complete the course was used as the tiebreaker.

The goal of the Optimus Dine robot was to complete all primary and secondary tasks, without much consideration of time. The following sections will further explain the individual competition including our strategy, the robot's final design, and its performance.

4.1. Competition Outline

The main priority for the individual competition was completing each primary task as these tasks made up most of the robot's score. To get this done, the team's programmers combined parts of the code from each performance test to create code that allowed the robot to complete the entire course. It was programmed to first go up the ramp, drop off the tray, slide the

ticket, flip the burger, flip the ice cream lever, press the jukebox bottom, and then finish the course. This course strategy can be seen below in Figure 15.

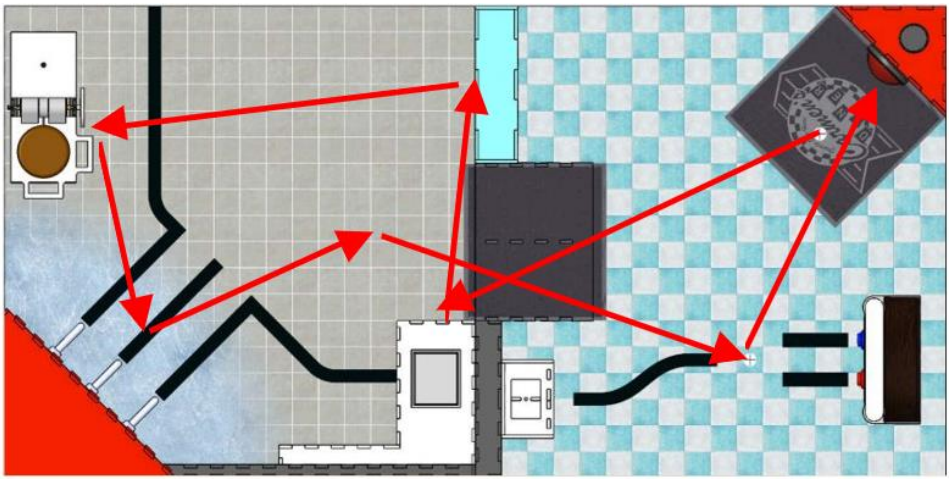

Figure 15: Final Course Strategy

The RPS system was implemented into the code to allow the robot to check its position and readjust. This was used to improve the consistency of the robot's runs.

4.2. Robot performance

Unfortunately, the robot did not perform well in the individual competition. For the individual competition, the robot performed three trials and failed to complete one or more tasks in each one. In the first trial, the team randomly selected course D. In this run, the robot's CdS funnel got caught on the ramp's lip and ended the run with a total score of 8/75. Between trials 1 and 2, no changes were made to the code.

For the second trial, the faculty selected Course F for our robot. After a strong start and making it up the ramp, the robot dropped the tray off, initialized the ticket, initialized the burger flipper, correctly got the ice-cream lever, but got caught going back down the ramp. The robot's final score on this trial was 57/100 with 45/75 of the primary points.

For trial three, the team selected Course E. In this trial, the robot started strong again and dropped off the tray but got stuck sliding the ticket. This ended the run and gave the team a score of 30/100. Therefore, the final score for the robot was a top score of 57/100.

4.3. Analysis

In each run, the robot faced a novel issue. For trial one, the CdS cell's funnel scraped the ramp's lip and stopped the robot which was an issue that only existed on Course D. This issue was fixed and will be explained in Section 4.4.

In trial two, the robot made it up the ramp and dropped the tray into the sink correctly. The robot missed its first points while sliding the ticket. It initiated the slide but could not get the ticket to its final position. After lifting the ticket lever, the robot angled away from the ticket structure losing its hold on the ticket. It also initiated the burger flip but did not spin the wheel entirely. This was because of the long length of the burger mechanism's prongs. They stuck too far through the revolving wheel and got caught on the hot plate. The robot then correctly switched the ice cream lever. While going down the ramp, the robot's back left wheel got caught, ending the run.

Lastly, in trial three, the robot made it up the ramp and dropped the tray into the sink correctly again. The robot initialized and moved the ticket, but the front left side of the robot got caught on the ticket slider structure, which ended the run. After the second trial's failed attempt of sliding the ticket, an adjustment was made in the code to position the robot closer to the ticket structure. This change caused the robot to get stuck in trial three.

4.4. Further Modifications

After a disappointing day on the robot course, the team had a week to improve the robot's consistency before the final competition. This required repeated testing to improve the robot's score and consistency in the final competition.

The team's biggest change to the robot involved the ticket lever's movement. After sliding the ticket, the ticket lever originally went back down, hovering just over the course floor. This caused the lever to scrape the lip of the ramp and changed the trajectory of the robot traveling down the ramp. To fix this problem, the team recalibrated the ticket servo motor and programmed the lever to rotate upward after sliding the ticket. Before the individual competition, the team also shaved the front two corners of the robot's chassis to reduce the risk of the robot getting caught on the edges of the course. The team tried to correct all the problems that occurred during the individual competition, both hardware and software. Hardware-wise, the shaved corners as aforementioned were shaved even more following the individual competition. Furthermore, the team shortened the CdS cell's funnel to prevent the robot from getting caught on Course D's ramp. To make sure the robot flipped the burger completely, the prongs of the burger mechanism were shortened, twisting the screws a little more into the servo's attachment. Software-wise, the team added time-outs to functions to assure that the robot avoided getting stuck on the course. If the robot was ever in one position for too long, the time-out function caused the robot to move on to the next task.

5. Final Design

The robot kept the same design following the fourth performance. The one final change that was made was cutting off the two front corners of the chassis. The final design, with these last changes, can be seen below in Figure 16.

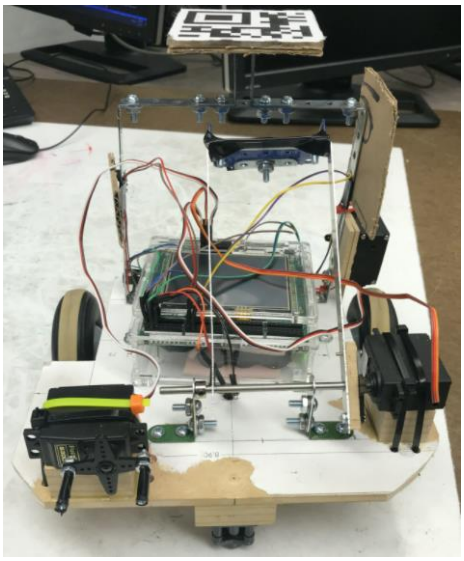

Figure 16: Final Robot Design

The idea behind this change was to decrease the likelihood of the robot bumping into objects on the course as it turned. This change helped prevent it from getting stuck or going off course.

5.1. Features

The final design included many features that allowed the robot to navigate the course and complete tasks efficiently. Between each performance test, new mechanisms were added to the robot, and its ability to complete more of the course was strengthened. The final chassis, drivetrain, mechanisms, and electrical systems are further explained in the following 5.1 sections.

5.1.1. Chassis

The final robot design was built on top of a square shaped MDF chassis. The chassis was a 7.75'' x 8.90'' square with a 1/4'' thickness. In each of the back corners of the chassis, there were 1.2'' x 3.75'' cutouts for the back wheels. The front corners of the chassis were cut off to prevent the robot from getting caught on the edges of the course. On the back of the chassis, there is a large, rectangular hole which was used to allow the motor wires through. The large circle hole in the center was used for the CdS cell. All other holes were attachment locations. The final chassis can be seen below in Figure 17.

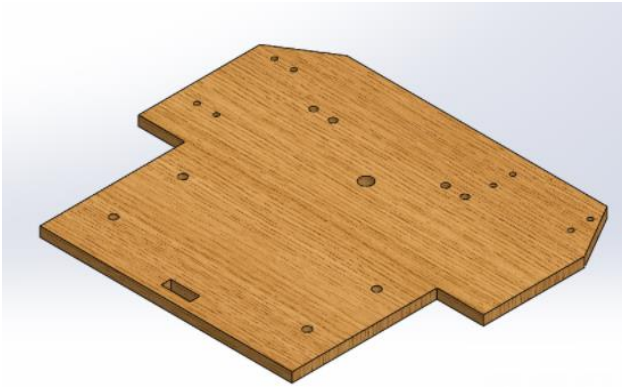

Figure 17: MDF Chassis

5.1.2. Drivetrain

The Optimus Dine robot drove with a drivetrain consisting of two 3.5'' DUBRO wheels with a caster ball in the front. The two DUBRO wheels were powered by IGWAN motors mounted to the bottom of the robot. The drivetrain of the final robot design can be seen on the following page in Figure 18.

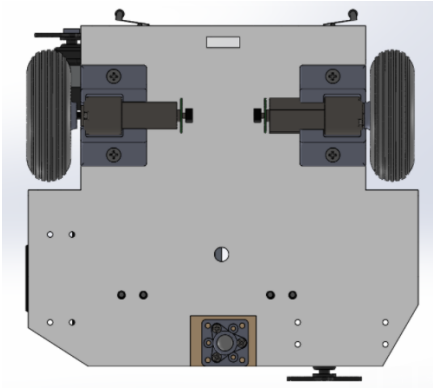

Figure 18: Final Design Drivetrain

This drivetrain performed as required. It successfully went up the ramp consistently. A benefit of the IGWAN motors was their built-in shaft encoders. The shaft encoders combined with the rubber band-wrapped wheels allowed the robot to drive in straight lines and at accurate lengths. This made navigating the course much easier.

5.1.3. Mechanisms/Arms

The final robot design consisted of three main mechanisms/arms. The primary arm, a large lever, was programmed to flip the ice cream levers and drop off the tray. The mechanism consisted of an aluminum lever, FITEC high torque servo motor, MDF spacers, erector pieces, and an axel. It can be seen below in Figure 19.

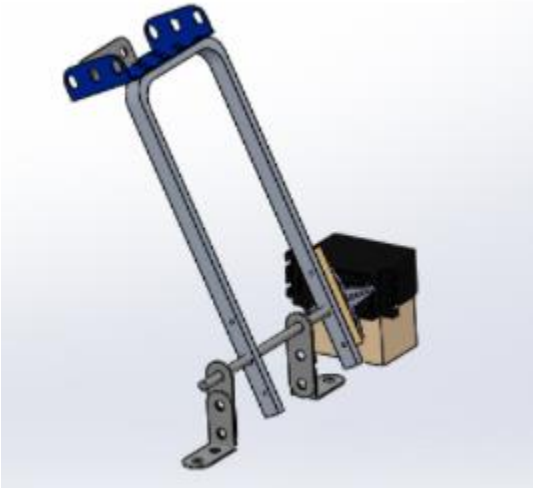

Figure 19: Ice-Cream Lever

Not shown in the figure above are the zip ties that were used to secure the servo motor to the MDF chassis of the robot. Because of the heavy tray added on the lever, the team decided to use a high torque servo motor as it could better handle the increased weight. This lever was first used to drop off the tray in the sink. It did this task by carrying the tray in the erector basket on top of the lever. When in the correct location, the lever turned and dropped the tray into the sink. The mechanism was also used to flip the ice cream levers. The robot positioned in front of the desired flavor and lowered the arm, flipping the correct ice-cream lever. The robot then backed up, waited seven seconds, drove forward, and raised the mechanism lever, flipping the ice-cream machine off.

The robot also consisted of an arm for sliding the ticket. This mechanism can be seen below in Figure 20.

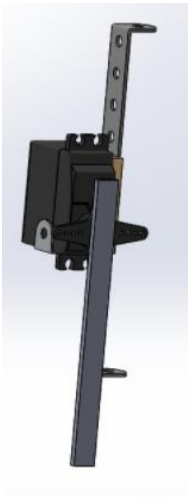

Figure 20: Ticket-Sliding Mechanism

This mechanism consisted of a FUTABA servo motor, an MDF lever and spacer, and erector pieces. Not included in the figure above are the zip ties used to secure the motor to the erector piece. When in front of the ticket slider, the lever was raised. The robot then drove

forward pulling the ticket with it. When the ticket was in its final position, the lever lowered, and the robot continued with its other tasks.

The last mechanism on the Optimus Dine robot was the burger-flipping fork. This mechanism consisted of a FUTABA servo motor with screws through the propeller-shaped attachment and an MDF spacer. The servo motor was secured to the robot with zip ties. To use this arm, the robot positioned itself, so the two screws were within the spaces of the revolving wheel of the burger station. The servo motor turned the hot plate, flipping the burger, and then turned back, returning the hot plate. A visual can be seen below in Figure 21.

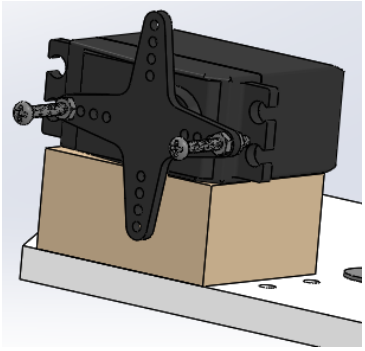

Figure 21: Burger Mechanism

Overall, the arms and mechanisms on the Optimus Dine robot were effective. If the robot positioned correctly, they completed their specific tasks and were efficient. The rotating claw and the lever on the front of the robot were also manipulated with code to press the jukebox buttons. The CdS cell read the correct color, and either lowered the lever to press the blue button or turned the burger fork to press the red button. This method allowed the robot to drive forward to hit the button instead of having to reposition and risk missing the button. This feature allowed the robot to finish this task in the same spot every time, leading to greater consistency when navigating from this task to the next spot.

5.1.4. Electrical Systems and Sensors

The Optimus Dine robot included two types of electrical sensors, a CdS cell, and two bump switches. In Exploration One, the team learned how to use a CdS to distinguish between different light colors. Powered by the proteus on the robot, a CdS cell was used during the robot course twice. To start the course, the CdS cell read when the start light turned on. The CdS cell sent voltage values, between 0.0 and 3.0, to the proteus, and if they were within a specific range, the proteus would perform the desired task. This ability was also used to press the correct jukebox button. The robot had to use the CdS cell to distinguish between the red and blue light shown in front of the jukebox. After reading the light and sending the voltage values to the proteus, the robot could press the correct button [2].

The team also learned from Exploration One about the importance of bump switches. When pressed, these sensors sent a value to the Proteus signaling it to perform the next action. These bump switches helped realign the robot to be straight and improved its task completion consistency. Before flipping an ice cream lever, the robot backed into the plastic wall to straighten its path. This guaranteed that the lever was consistently flipping the correct ice cream lever.

Along with these electrical systems, the robot included servo motors and IGWAN motors that received their power from the Proteus. The Proteus was programmed to power the motors at the desired times to either complete a task or move the robot. Each of the robot's mechanisms consisted of a servo motor. The ticket-sliding lever and burger-flipping fork used FUTABA servo motors. The ice cream and tray-drop off lever used a FITEC high torque servo motor. The robot's drivetrain consisted of two IGWAN motors. These motors provided the robot with accurate shaft encoding and made it easier for the robot to travel the course. Figure B9 displays a connection guide of all the electrical systems and can be found as Figure B8 in Appendix B.

5.1.5. Discussion of Final Code

The team's final code was about 1100 lines and consisted of a class named 'robotFunctions'. This class included all the functions and global variables to make the robot do the tasks. Figure C20, found in Appendix C, shows the team's class function.

The team's functions were set up and organized to know their purpose, including distinct functions that allowed the robot to move forward, backward, right, and left. The final code also had functions for RPS coordinates and functions for debugging various motors and sensors, such as the servo motors and CdS sensors. For instance, the team implemented simple functions like moving the robot forward and combined them with complex RPS functions that helped the robot relocate itself. Calling these functions together allowed the robot to move forward quickly and then slow down to check for the correct RPS coordinates. The robot's quick and efficient movement was accounted by the many combinations of the team's different functions. The team also had many constants that allowed quick changes to the robot's shaft encoding. Because of the organization of the functions and variables, the team only had one line of code in the main function to run all tasks for the robot. High-level flowcharts of each function are shown in Appendix C.

Lastly, the final code consisted of many comments that explained the purpose of different functions. The comments helped the team, both the constructors and programmers, debug issues withing the code. Overall, having a framework from the initial and final code allowed the coders to easily implement necessary changes between each test, and the combination of simple functions allowed the robot to complete any required task.

5.1.6. Strengths and Weaknesses of the Final Design

The Optimus Dine robot's final design had many strengths as well as some weaknesses. The overall strength of this robot was its simplicity. It was composed of many basic mechanisms

that were easy to test and debug. Having a specific arm for almost every task was both a positive and a negative. It clarified which arm was malfunctioning and also allowed the robot to travel the course without making as many turns, reducing the chance of missing tasks. However, having more arms also increased the likelihood of error or breakage. They also took up space on the robot and increased the robot's overall weight. To account for this increased weight, the programmers had to constantly adjust the code to account for the different weight distributions.

Another small weakness of the final robot design was the CdS cell's narrow funnel. After having to shorten the length of the funnel to safely make it up Course D's ramp, the remaining funnel was short and narrow. This caused the robot to occasionally misread the jukebox light color. However, it happened rarely enough for it not to be of main concern.

A unique strength of this robot was its method for completing the jukebox station. After positioning the CdS funnel on top of the light, the robot reacted in one of two ways. If the light was blue, the tray mechanism lowered to the jukebox button's height and the robot drove forward, pressing the button. If the robot read a red-light voltage, the burger mechanism would revolve, and the robot would drive forward pressing the red button. This efficient way of completing the task reduced any risk that repositioning the robot make cause and saved a lot of time.

5.2. Final Budget

The team had $160 at the beginning of the project and was expected to purchase all needed materials with this money. After fully constructing the robot, the team was left with $29.92, spending $130.08. The distribution of the spent money can be seen below page in Figure 22.

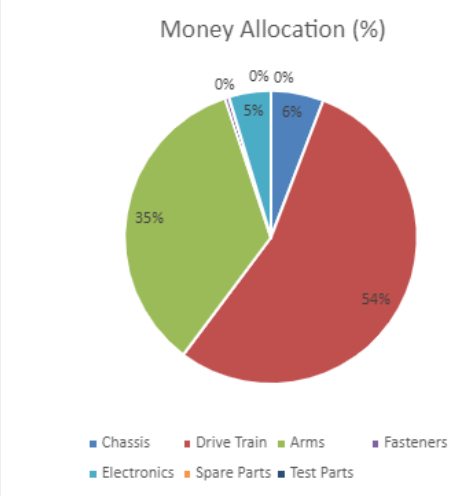

Figure 22: Money Distribution

As shown in the figure above, the team spent most of its budget on the robot's drivetrain. This makes sense as the IGWAN motors used were expensive. This section also included the wheels and motor mounts. Another substantial portion of the budget was spent on the robot's arms. Each arm required a different servo motor, and the motors were expensive. However, the team was impressed with the $29.00 remaining. It gave the team a substantial amount of money to be used in case of an emergency. It also meant that Carmen's Diner would be able to mass-produce these robots at a cheaper rate than other robots.

5.3. Final Schedule

The team spent over 400 hours on this design project. The distribution of the number of hours spent on each section of the project can be seen in Table 3 on the following page.

Table 3: Time Tracking Totals

Team Totals		
Category	Hours	Percent
Documentation	103.5	24.8%
Project Management	42.5	10.2%
CAD	39	9.3%
Coding	85.5	20.5%
Construction	52	12.5%
Testing	95	22.8%
Other	0	0.0%
Total Time	417.5	

Overall, the team spent most of its time on documentation and testing. These sections combined were approximately 198.5 hours. At the beginning of the project, the team created a project schedule and estimated the total number of hours for the entire project to be 140.9 hours. However, the team did not expect any of the sections of the robot project to take as long as they did. The long hours of testing were necessary to ensure consistency in every course and updating the documentation was integral to keeping the team updated with the robot's changes. Nevertheless, the team's policy of completing deadlines ahead of time resulted with ease in submitting on time.

6. Final Competition

The final competition took place at the RPAC on Saturday, April 9. Each robot team got the chance to compete against the other teams in round-robin and bracket-style elimination matches. The Round Robin included three separate runs where robots were judged on their task completion, consistency, and innovation. The final competition was a 57-team competition where each robot competed against three teams per round with only one progressing to the next round. The robot with the highest score and fastest time was the winner. This event featured multiple scholarship prizes for various categories. These included top finishers in both round-robin and elimination, outstanding engineering, outstanding aesthetics, and outstanding innovation. The course and point system were identical to the individual competition.

6.1. Strategy

The strategy for the final competition differed slightly from the individual. Time was now a factor in the robot's ability to make it far in the competition. With an extra week to prepare and refine the code, the team spent the rest of the time perfecting the consistency and figuring out ways to shorten the run. The average time for a perfect run before the individual competition was around 1 minute and 45 seconds. After a week of removing unnecessary steps in the code, the average time for a perfect run was around 1 minute and 15 seconds. This new time was 30 seconds faster than the week prior and the robot was significantly more consistent.

6.2. Results

The competition was a mix of success and failure. The initial practice run was perfect with a fast time. However, the round robins were not as successful. In the first run of the round-robin, the robot started strong. It completed the tray drop-off, initialized the ticket, and flipped an ice cream lever. However, on its way down the ramp, the robot's left wheel got caught on the

ticket mechanism. This resulted in a score of 33/100. In the second and third rounds of this stage, the robot got caught on the sink station when turning to complete the tray drop-off. This error ended each of the runs with a score of 8/100. With only one more run left, in the bracket elimination competition, the team decided to alter the code in hopes of fixing the ramp issue. This adjustment ended up being exactly what was needed, and the robot finished with a score of 90/100, just a few points off moving on to the next round.

6.3. Analysis

Overall, the team was not satisfied with the final performance of the robot. After consistent runs in testing, the team was ready for a more successful round robin than what occurred. The robot was having issues that it did not face in the FEH classrooms. It never had an issue making it down the ramp or dropping the tray off. These new occurrences were most likely due to the new location of the course and the effects it had on the RPS system. Unfortunately, the team did not realize this early enough. If the code had been changed after the second-round robin run, the third round may have been more successful. This could have positively affected the robot's performance in the final competition as well.

Although the round-robin went poorly, the team's morale was boosted by the robot's performance in the bracket round. Its completion of almost all the tasks and strong battle with the other teams was the correct representation of the work that had been put into this project. It performed more similarly to its testing runs and proved to be the optimal robot for Carmen's Diner.

7. Summary and Conclusions

Overall, the robot project was a large success and a testament to the engineering design process. As outlined in Figure A2, the steps of this process included developing specifications, creating design concepts, designing a solution, building a prototype, and testing the design. This was not a linear process but a cycle that involved circling back to previous steps and working back up again. The team worked through this process and cycled back many times to create the robot.

7.1. Overview

Before brainstorming design concepts, the requirements and specifications of the course objectives had to be measured to create compatible designs. This included performing a drivetrain analysis to find a motor with the proper specifications to complete the course. After finding these requirements, design concepts were created around these specifications. Each team member combined their individual brainstorming ideas into multiple full robot design concepts. The design concept with the most potential was the triangular chassis Omni wheel design. After designing additional solutions and building a cardboard prototype, the main lever and small chassis seemed to be incompatible with the requirements of this project. The team circled back to the second stage of the process, re-evaluated the specifications of the course, and created a new design concept. The prototype of the new design proved to be much more successful. It consisted of a square MDF chassis, IGWAN-powered Dubro wheels, a caster ball, and three servo-powered mechanisms. One servo motor had prongs attached to the servo's fan and was used to flip the burger. Another servo had a large u-shaped lever attached to it that delivered the tray and flipped the ice-cream switches. The third servo lifted and lowered a lever to slide an order ticket.

With these components on the robot, the team's programmers designed a code that positioned the robot in all the right places.

Each week, the team went back and forth between building the prototype and testing the design to meet the specifications of each performance test. These tests included multiple course tasks and challenged the robot and the team's work ethic. However, they were a success and the Optimus Dine robot proved to be one of the best. After adding all the mechanisms and attaching the QR code bracket, the robot was completed within the team's budget of $160.

With this design and the finalized code created by the programmers, the robot was put to test in both the individual and final competitions. In the individual competition, a test that required all the course tasks, the robot performed decently with a top score of 57/100. In the week before the final competition, the robot was tested for hours to improve its consistency and speed. There were also some last-minute changes made including shortening the CdS cell funnel and shortening the burger mechanism's prongs. The first rounds of the final competition did not go as planned. The robot was having first-time errors including running into the sink and ramp corners. It finished the round-robin rounds with scores of 33/100 and two 8/100's. To fix the robot for the bracket rounds, the programmers made a small change in the code that ended up paying off. The robot performed well in the tournament earning a score of 90/100 and almost progressing to the second round. This round represented the potential of Optimus Dine-in Carmen's Diner.

7.2. Future Improvements

If there was more time available to perfect the Optimus Dine robot, improvements could be made to increase the efficiency and consistency of the robot.

Physically, the team would find a more permanent way of securing the servo motors. Although the zip ties secured the servo motors well for this project, they were not the most

professional method. A robot with zip ties may appear somewhat “sketchy” in a diner. Another physical modification that the team would make involves the tray method drop-off. Although the lever drop-off got the job done, each run included the risk that the tray may miss the sink. Along with updating the tray drop-off, the team would reconstruct the CdS cell’s funnel. On the Optimus Dine prototype, the funnel is constructed from printer paper wrapped with electrical tape. This design would not hold up well in the busy setting of a diner. To correct this, the team would create a funnel part on Solid works, and then have the piece 3d printed. This would be a more permanent way of improving the CdS cell’s functionality.

Along with these physical modifications, the team would make software modifications as well. To improve the driving of the robot, the programmers would add a PID system to the code. This system would allow the Proteus to find differences within each motor's shaft encoding, and proceed to alter each motor’s speed to correct the difference and result in a straighter path.

With these changes implemented, the Optimus Dine robot would be more consistent in completing each of the assigned tasks. With guaranteed task completion, this robot would allow the human employees to connect with customers and make the Carmen Diner experience one of a kind.

8. References

[1] FEH Program, “Design Problem Statement & Specifications” Ohio State University Department of Engineering Education, Columbus OH, United States. 2022. Accessed March 2022 [Online]

[2] Exploration_01_Writeup. 2022, April 4.

[3] Exploration_02_Writeup. 2022, April 20.

[4] Exploration_03_Writeup. 2022, April 21

[5] FEH Robot Store, 2022, April 4.

https://store.apps.feh.osu.edu/

APPENDIX A
Project Tracking

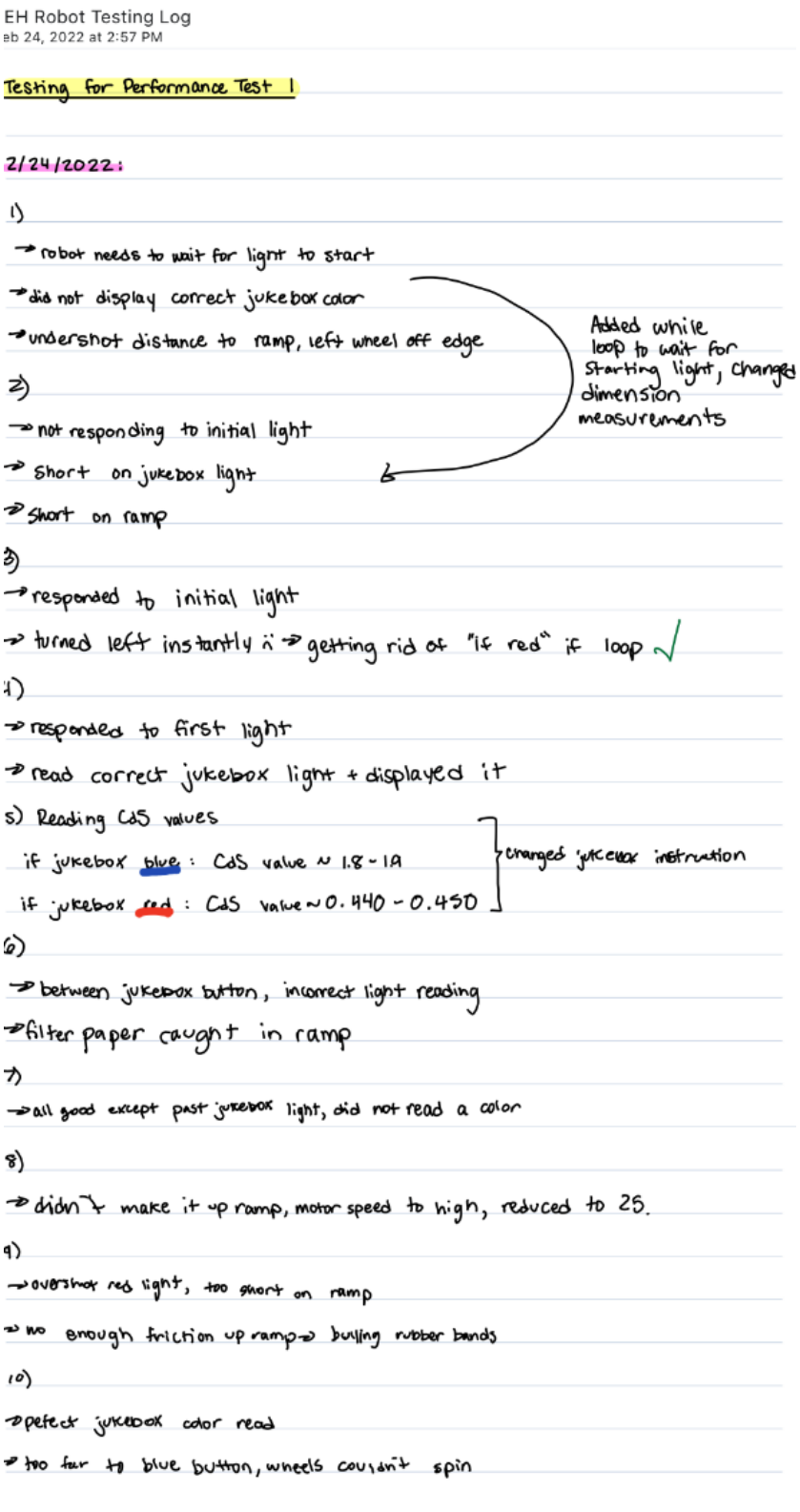

Figure A1: Testing Notes

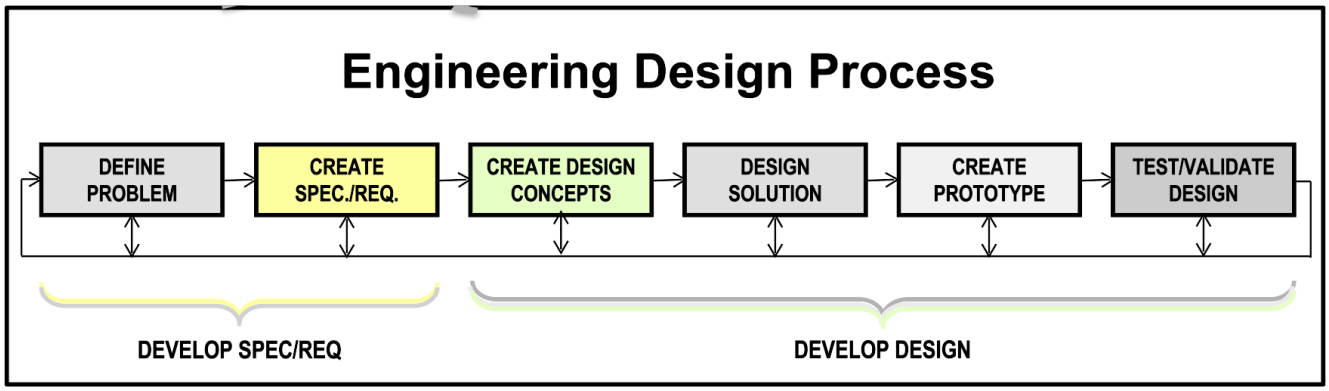

Figure A2: Engineering Design Process Diagram

APPENDIX B
Additional Robot Figures

Chassis:

	Build Space	Stability	Economic	Weight	Functionality	total
Weight (1-3)	2	3	1	2	3	
Tank	4	4	1	1	4	35
Triangular Body, with Omni wheels On corners	4	4	2	3	5	43
Cube shape with four wheels – static claw	4	5	1	3	4	42
Robot with no body, just arms – static claw	3	1	4	4	1	24

Figure B1: Chassis Decision Matrix

Sliding Order Ticket:

	Duality	Required Space	Buildability	Economic	Total
Weight	2	2	3	1	
Sliding Fork	3	2	4	4	26
Static Fork – robot moves	5 with the omni wheels	5	5	4	39
Separate fork (off-side) drop-in	2	3	5	4	29
Sticky Arm	1	5	3	5	30

Figure B2: Sliding Order Ticket Mechanism Decision Matrix

Pressing Jukebox Button:

	Duality	Force/Stability	Buildability	Economic	Consistency	total
Weight	2	3	3	1	3	
Existing claw	5	5	5	1	3	50
Robot runs into button	1	4	5	5	4	46
Launcher	1	4	2	2	1	25
New ext. claw	1	4	4	3	3	38

Figure B3: Pressing Jukebox Button Mechanism

Flipping Ice-Cream Lever:

	Duality	Functionality	Strength	Buildability	Economic	Consistency	Total
Weight	2	3	2	3	1	3	
Rotating claw	5	4	4	3	1	4	49
New lifter claw with a flat surface	2	4	5	3	1	4	48
Bar on a pulley system	1	4	4	3	2	4	45
Cylinder claw that lever inserts into	1	3	5	2	3	2	36

Figure B4: Flipping Ice Cream Lever Mechanism

Conveying/Depositing Tray

	Accuracy/ Consistency	Buildability	Stability	Duality	Economic	Total
Weight	3	3	2	2	1	
Rotating the tray into the sink – drops in	3	5	2	5	4	42
Claw that has arms that extend outward	3	3	4	3	1	33
Inclined surface with a barrier on top of the robot. Barrier drops to deposit	4	3	3	1	1	30
Surface that holds tray and inclines to drop	2	4	4	1	1	29

Figure B5: Conveying/Depositing Tray Decision Matrix

Flipping the Burger:

	Duality	Functionality	Strength	Buildability	Expense	Consistency	Total
Weight	3	3	2	3	1	3	
Rotating Fork	5	5	3	3	1	4	58
Claw that extends upwards and to the side	3	4	4	1	1	4	45
Spring like structure that catapults the burger and then falls back down	1	2	5	4	2	2	39
Inclined, wedge arm that forces burger over	1	3	4	2	2	4	40

Figure B6: Flipping the Burger Mechanism Decision Matrix

Pressing Final Button:

	Duality	Force/Stability	Buildability	Economic	Total
Weight	2	3	3	1	
Existing claw	5	4	4	5	39
Robot runs into button	1	4	5	5	34
Launcher	1	3	2	2	19
New arm specific for final button	1	4	4	1	27

Figure B7: Pressing the Final Button Mechanism Decision Matrix

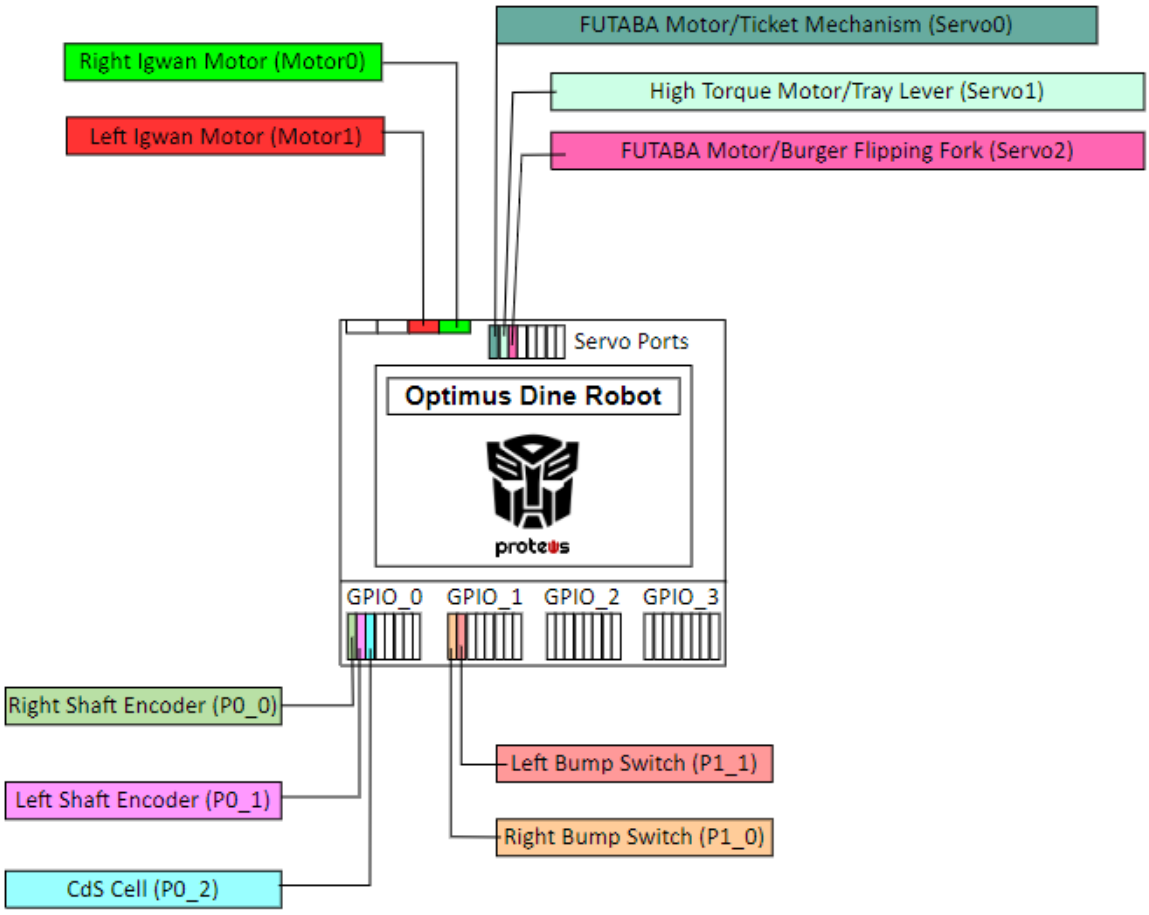

Figure B8: Electrical Systems Diagram

Table B1: Competition Primary Points

Primary Tasks	**Points**
Initiate on start light (Notes 1 & 2)	8
Press the final charging button (Note 17)	8
Return tray to either trash bin or sink (Notes 3 & 4)	7
Move ticket from starting position (Note 5)	8
Any ice cream lever is flipped down (Note 8)	8
Any ice cream lever is flipped back up (Note 9)	7
Press any juke box button (Note 12)	8
Press the correct juke box button (Note 13)	7
Burger flip is initiated (Note 14)	7
Burger flip is completed (Note 15)	7
Possible Primary Task Points	**75**

Table B2: Competition Secondary Points

Secondary Tasks	
Move ticket to final position (Note 6)	6
Ticket remains in final position (Note 7)	4
Only correct ice cream lever is flipped down (Note 10)	7
Ice cream lever is left in down position for at least 7 seconds (Note 11)	5
Hot plate is returned to initial position (Note 16)	3
Possible Secondary Task Points	**25**
Total Possible Task Points	**100**
Penalties	
Interfering with a competitor's robot (Note 18)	DQ

APPENDIX C
Code Diagrams

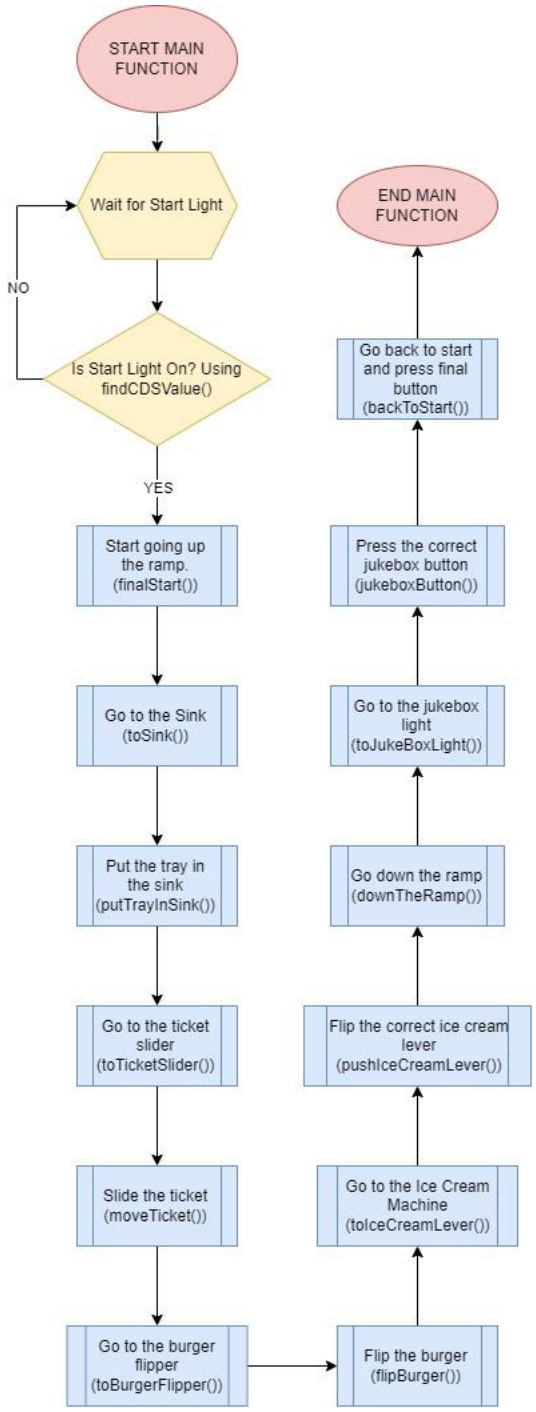

Figure C1: High Level Flowchart

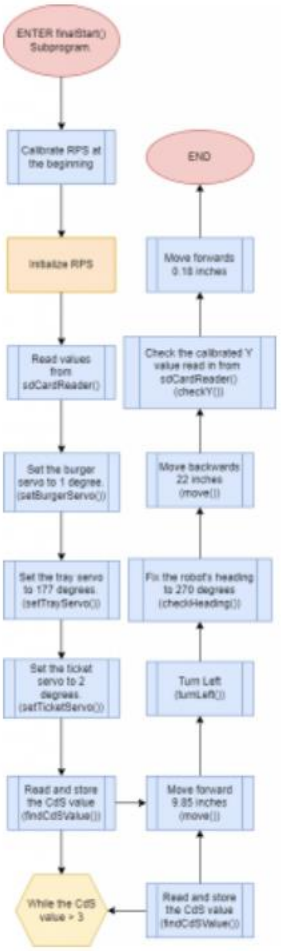

Figure C2: Final/Start Function Flowchart

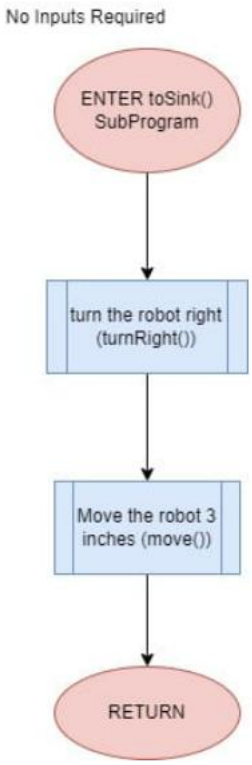

Figure C3: To Sink Function Flowchart

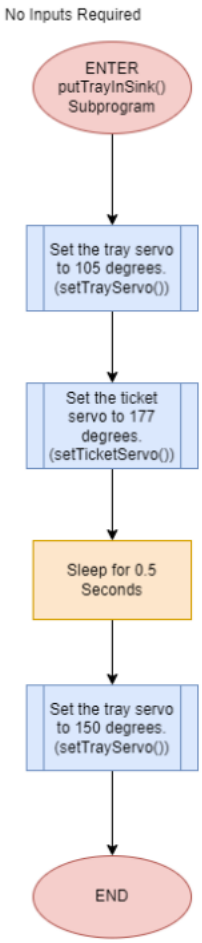

Figure C4: Putting Tray in Sink Function Flowchart

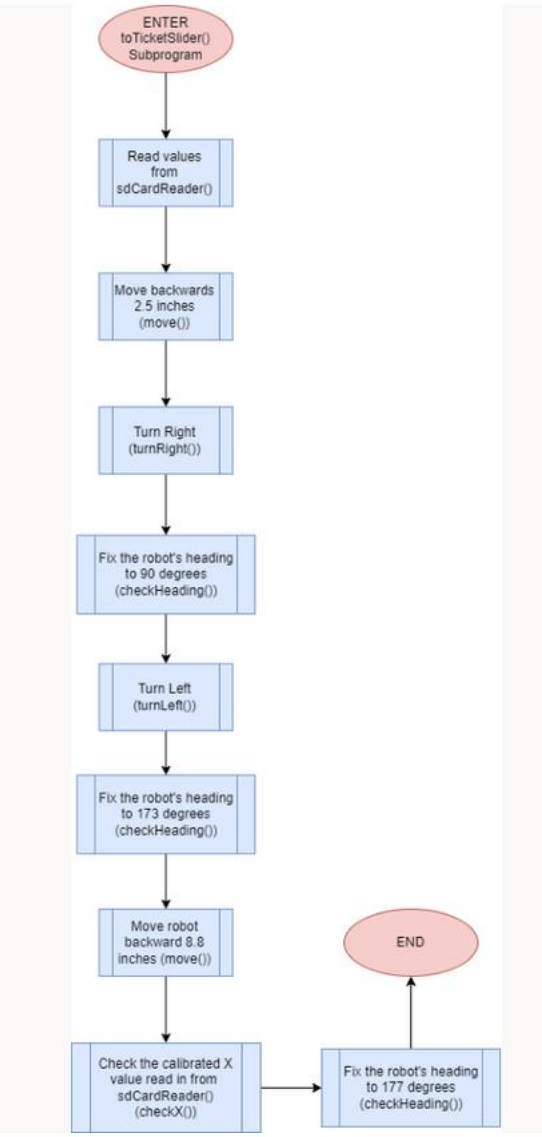

Figure C5: To Ticket Slider Function Flowchart

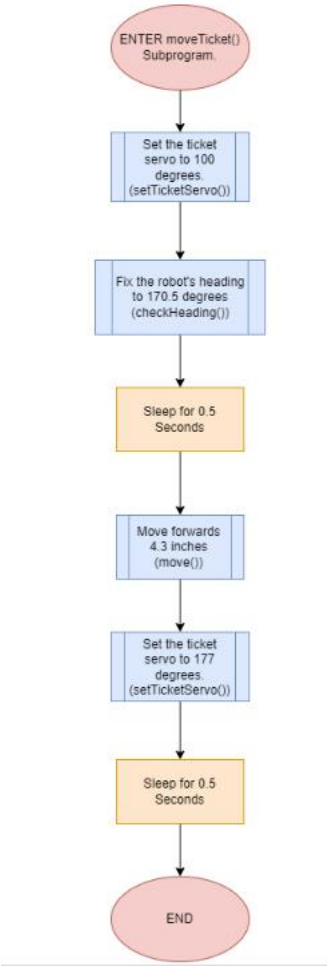

Figure C6: Move Ticket Function Flowchart

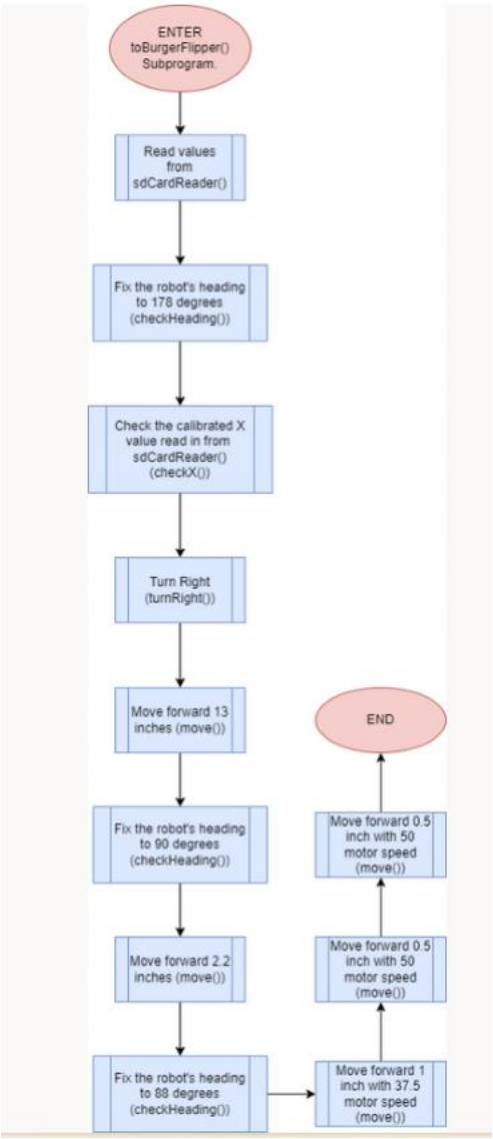

Figure C7: To Burger Flipper Function Flowchart

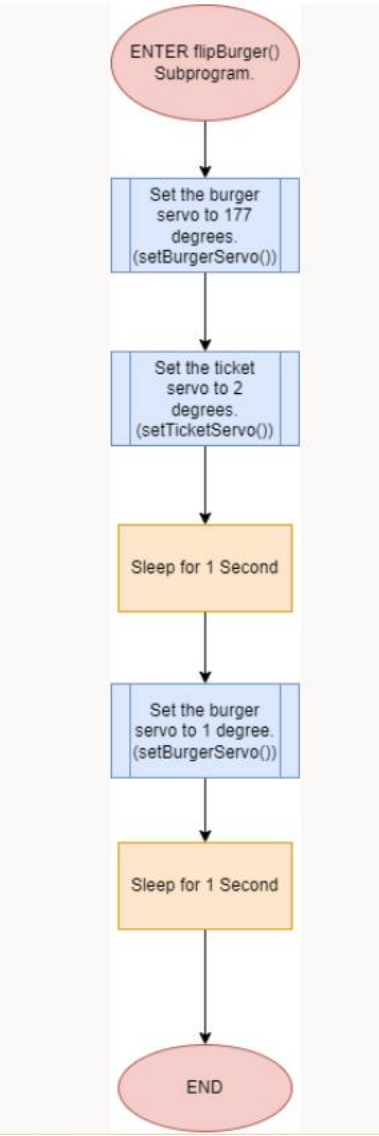

Figure C8: Flipping Burger Function Flowchart

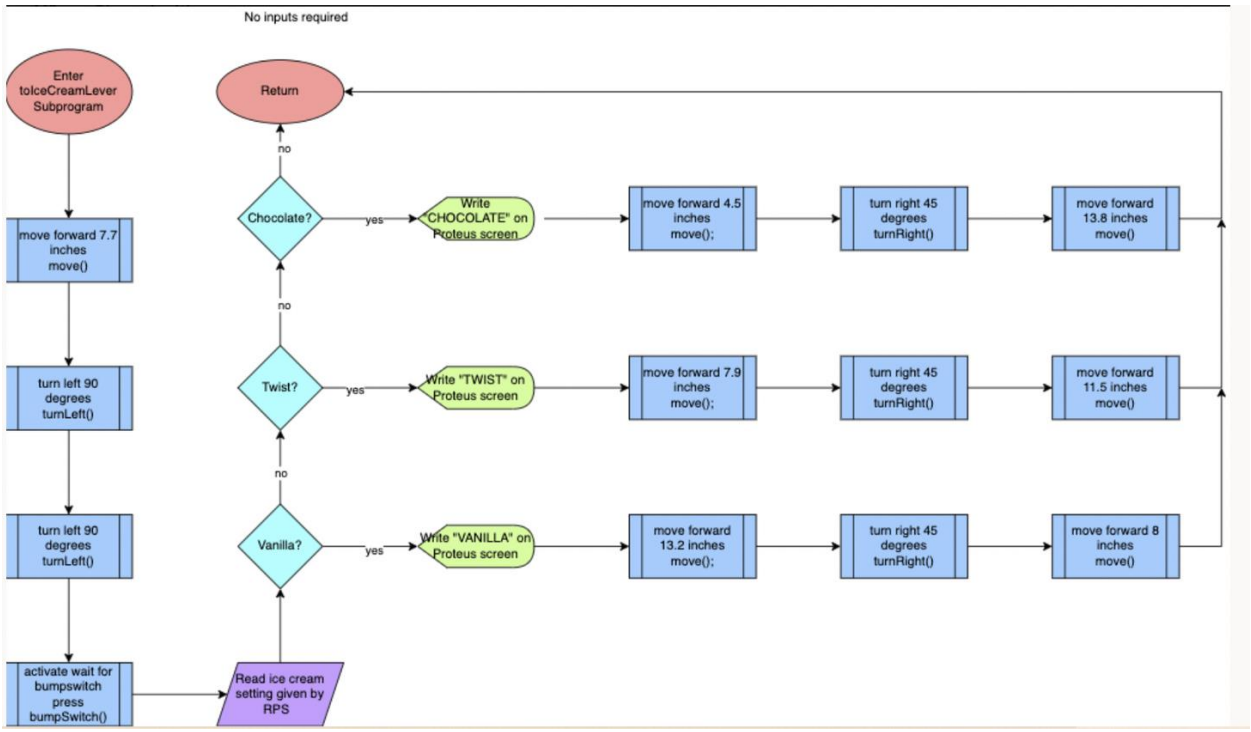

Figure C9: To Ice-Cream Lever Function Flowchart

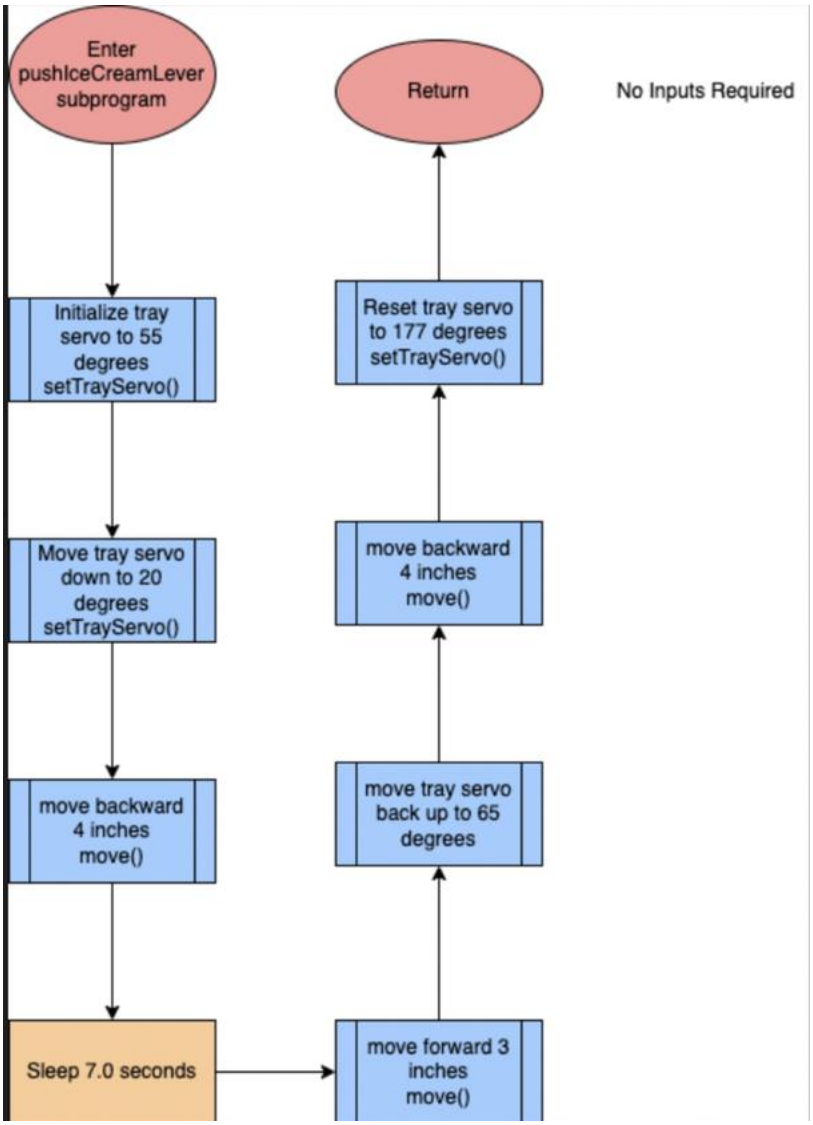

Figure C10: Push Ice-Cream Lever Function Flowchart

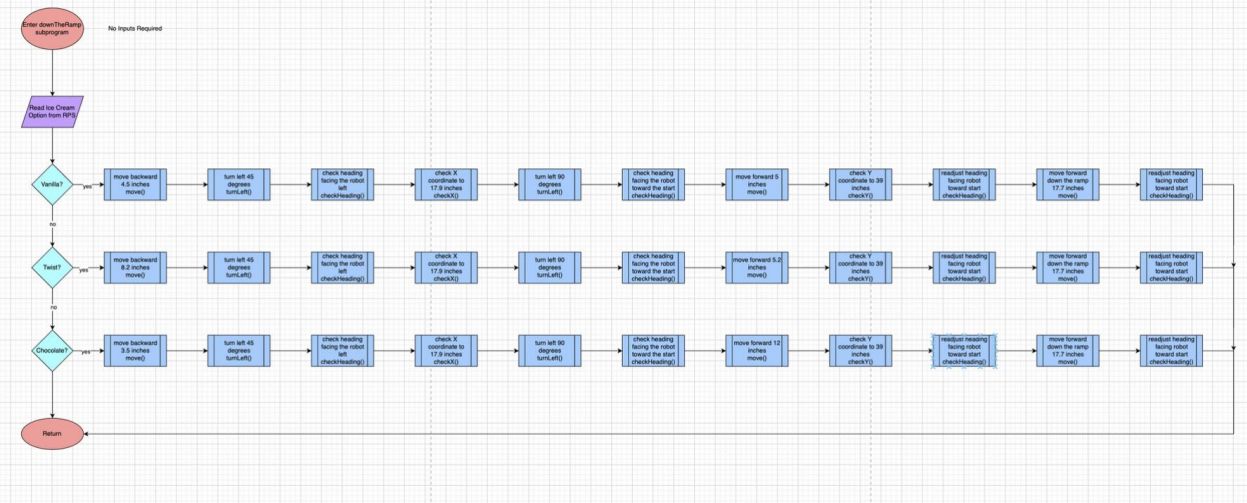

Figure C11: Down the Ramp Function Flowchart

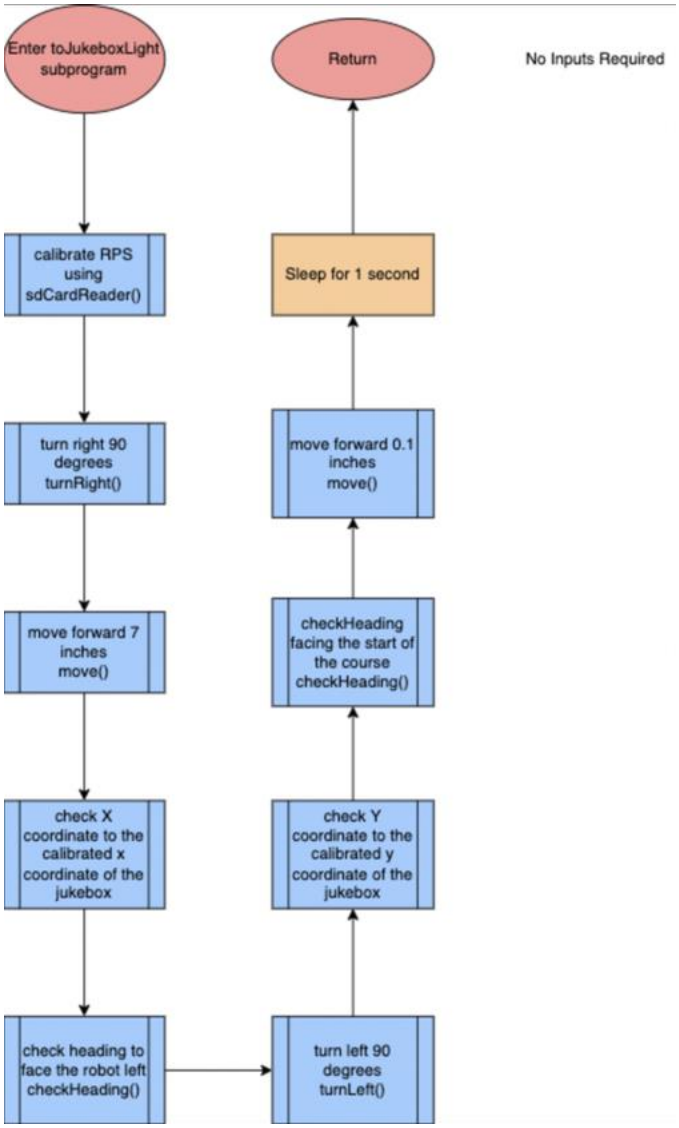

Figure C12: To Jukebox Light Function Flowchart

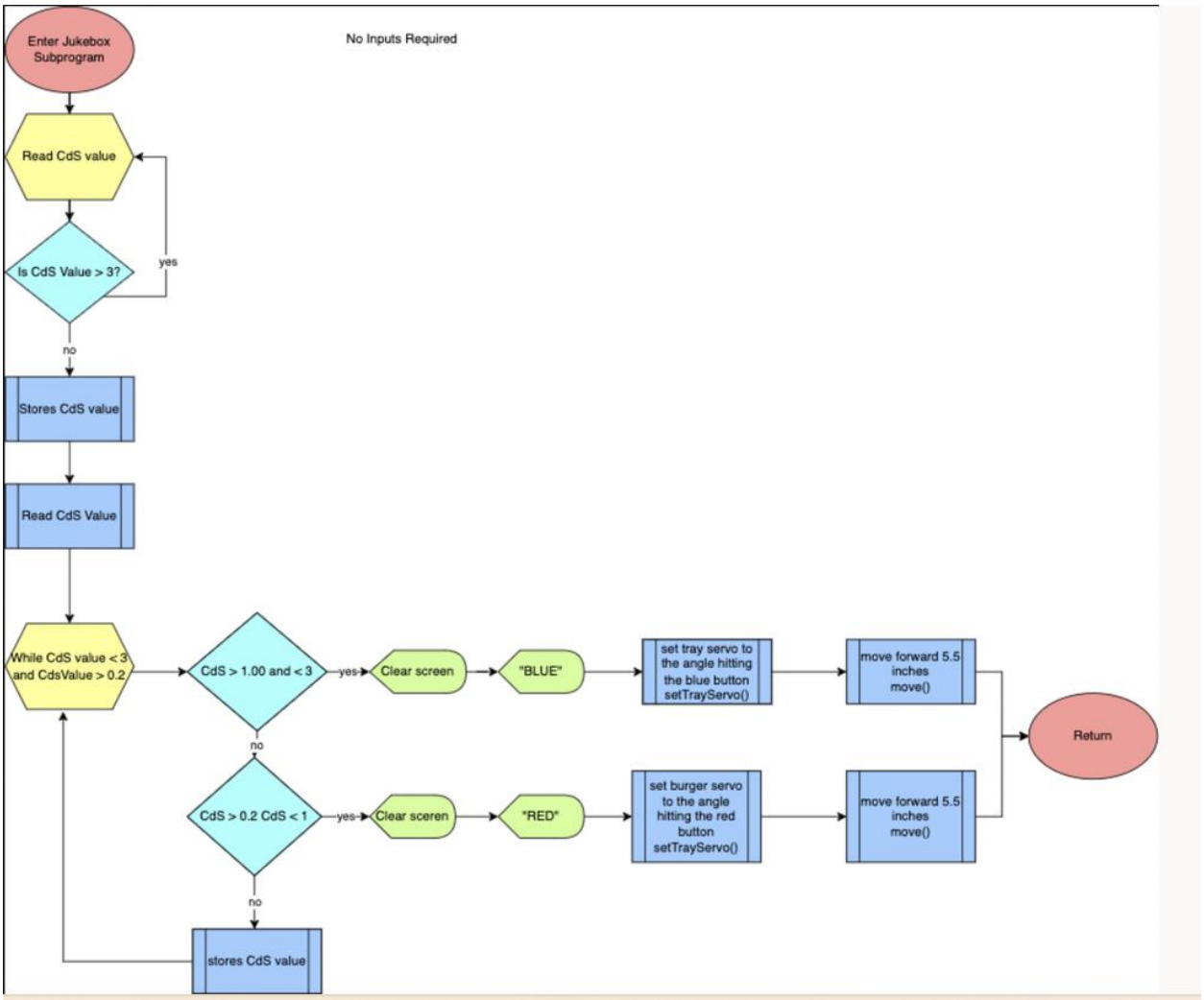

Figure C13: Pressing Jukebox Button Function Flowchart

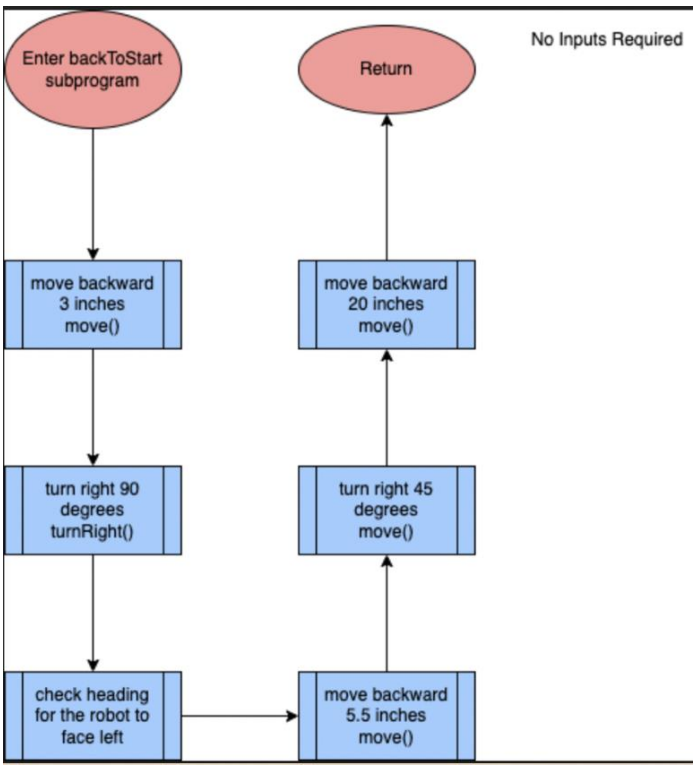

Figure C14: Back to Start Function Flowchart

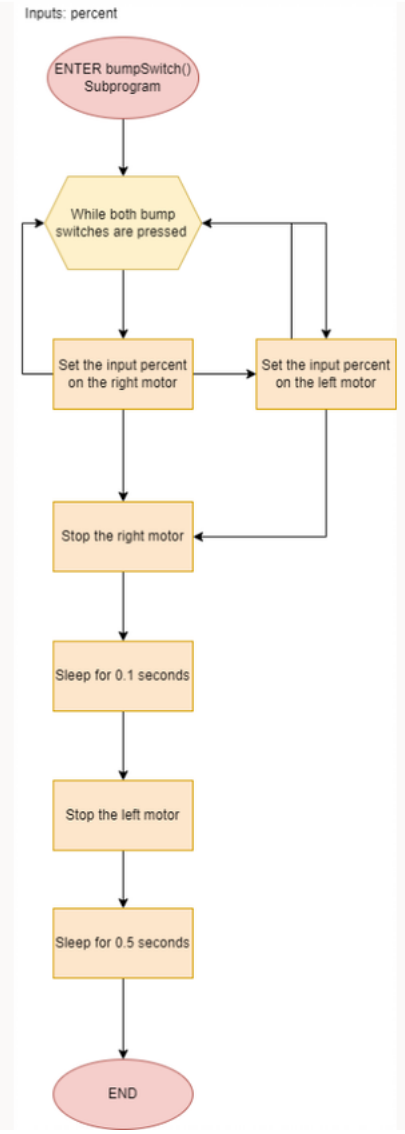

Figure C15: Bump switch Function Flowchart

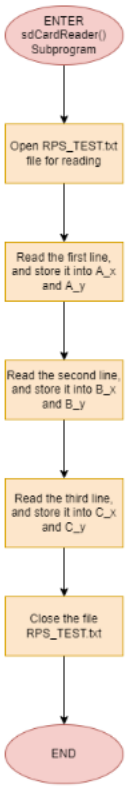

Figure C16: SD Card Reader Function Flowchart

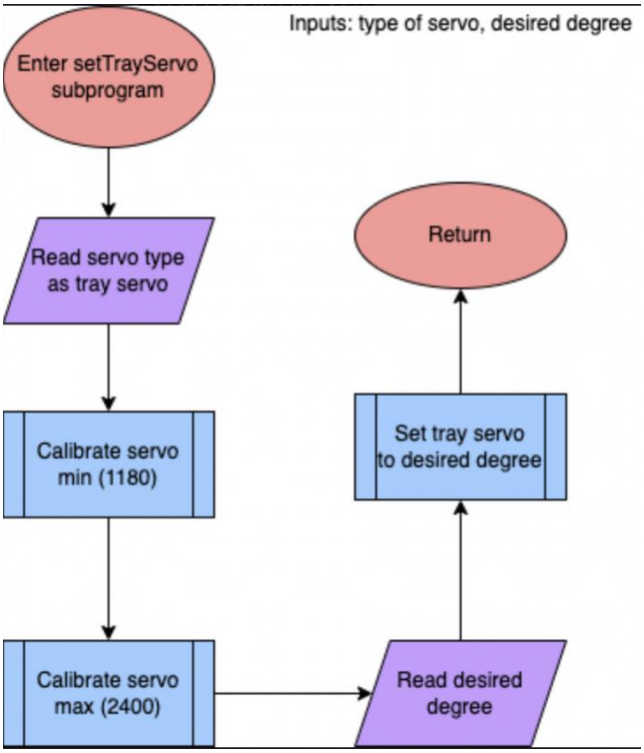

Figure C17: Setting Tray Servo Function Flowchart

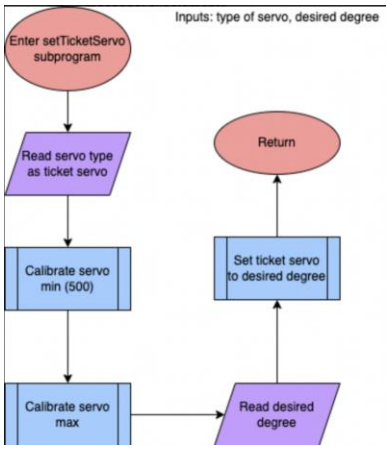

Figure C18: Setting Ticket Servo Function Flowchart

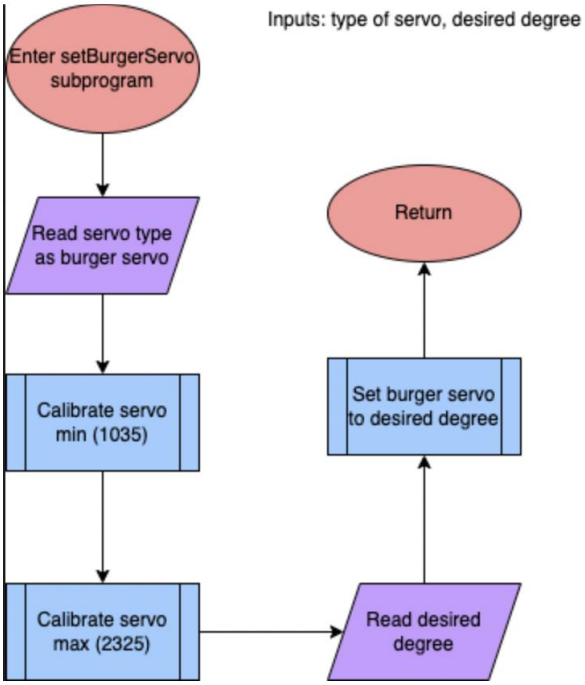

Figure C19: Setting Burger Servo Function Flowchart

```
//Class Function for Robot
class robotFunctions {
    private:
        float counts;
        float CdSValue;
        float motorSpeed;
        float countNumber;
        float A_x;
        float A_y;
        float B_x;
        float B_y;
        float C_x;
        float C_y;
        float D_x;
        float D_y;
        float E_x;
        float E_y;
    public:
        robotFunctions(float c = 0, float CdS = 0, float ms = 0, float cn = 0, float ax = 0, float ay = 0, float bx = 0, float by = 0, float cx = 0, float cy = 0, float dx = 0, float dy = 0, float ex = 0, float ey = 0);
        void move(int percent, int counts);
        void move(int percent, int counts, float timeOut);
        void turnRight(int percent, int counts);
        void turnRight(int percent, int counts, float timeOut);
        void turnLeft(int percent, int counts);
        void turnLeft(int percent, int counts, float timeOut);
        void turnRightWithOneMotor(int percent, int counts);
        void turnLeftWithOneMotor(int percent, int counts);
        void bumpSwitch(int percent);
        void pulseCounterclockwise(int percent, float seconds);
        void pulseClockwise(int percent, float seconds);
        void pulseForward(int percent, float seconds);
        void checkX(float x_coordinate, int orientation);
        void checkY(float y_coordinate, int orientation);
        void checkX(float x_coordinate, int orientation, float timeOut);
        void checkY(float y_coordinate, int orientation, float timeOut);
        void checkHeading(float heading, float timeOut, float pulse_time);
        void findCdSValue();
        void setTicketServo(FEHServo servo, float degree);
        void setTrayServo(FEHServo servo, float degree);
        void setBurgerServo(FEHServo servo, float degree);
        bool isPressed(bool value);
        void changeCounts(float cn);
        void changeMotor(float ms);
        void sdCardReader();
        void stopMotors();
        void cdsDebugger();
        void calibrateRPS();
        //FINAL TASKS
        void finalStart();
        void toSink();
        void putTrayInSink();
        void toTicketSlider();
        void moveTicket();
        void toBurgerFlipper();
        void flipBurger();
        void toIceCreamLever();
        void pushIceCreamLever();
        void downTheRamp();
        void toJukeboxLight();
        void jukeboxButton();
        void backToStart();
        void final();
};
```

Figure C20: Robot Function's Class

APPENDIX D
Equations

(1) $T = f * d * sin\theta$

$T = Torque$

$f = Force$

$d = Distance$