Speed D8ers R17

Engineering 1282H

Spring 2018

Team D8

Justin Banke

Jack McGinness

Kelly Meaden

Colby Williams

A. Urschel Mon/Wed/Fri 10:20

Date of Submission: 04/02/18

Executive Summary

Table of Contents

Executive Summary

Table of Contents i

List of Figures iv

List of Tables vi

1. Introduction 1

2. Preliminary Concepts 2

2.1 Project Requirements and Constraints 3

2.2 Brainstorming and Idea Screening 5

2.2.1 Course Strategy 6

2.2.2 Chassis and Drivetrain 7

2.2.3 Car Jack Mechanism 7

2.2.4 Buttons 7

2.2.5 Fuel Crank 8

2.2.6 Wrench 8

2.3 Preliminary Ideas 9

2.3.1 Preliminary Idea One 10

2.3.2 Preliminary Idea Two 11

2.3.3 Preliminary Idea Three 11

2.4 Mockup 12

Figure 3: Physical mockup. 12

Figure 4: SolidWorks mockup. 13

2.5 Preliminary Code 13

3. Analysis, Testing, and Refinements 15

3.1 Course Measurements 15

3.2 Drivetrain Calculations 16

3.3 Explorations 19

3.3.1 Exploration 1 19

3.3.2 Exploration 2 20

3.3.3 Exploration 3 22

3.4 Testing 23

3.4.1 Performance Test 1 23
3.4.2 Performance Test 2 26
3.4.3 Performance Test 3 27
3.4.4 Performance Test 4 29
3.4.5 Individual Competition 31
3.4.6 Final Competition 32

4. Individual Competition 33
4.1 Strategy 33
4.2 First Trial 34
4.3 Second Trial 35
4.4 Third Trial 36
4.5 Modifications 37

5. Final Design 38
5.1 Features 38
5.1.1 Chassis 38
5.1.2 Drivetrain 39
5.1.3 Electrical System and Sensors 40
5.1.4 Mechanisms 41
5.2 Code 42
5.3 Budget 43
5.4 Schedule 45
5.4.1 Design Schedule 45
5.4.2 Time Log 46

6. Final Competition 47
6.1 Strategy 48
6.2 Round Robin 1 48
6.3 Round Robin 2 48
6.4 Round Robin 3 48
6.5 Single-Elimination 48
6.6 Performance Analysis 48

7. Summary and Conclusions 49
7.1 Summary 49
7.2 Conclusions 50

8. References 51

APPENDIX A: Brainstorming **A1**

APPENDIX B: Preliminary Concept Sketches **B1**

APPENDIX C: Analysis and Explorations **C1**

APPENDIX D: Sample Calculations **D1**

APPENDIX E: Robot Design **E1**

APPENDIX F: Testing Strategies **F1**

APPENDIX G: Electrical Systems **G1**

APPENDIX H: Code **H1**

APPENDIX I: Budget **I1**

APPENDIX J: Schedule **J1**

List of Figures

Figure 1: Overhead view of the pit area and garage, as provided by OSURED [2]. 12
Figure 2: Top-ranked course strategy. 15
Figure 3: Physical mockup. 21
Figure 4: SolidWorks mockup. 22
Figure 5: Robot for Performance Tests 1 and 2. 34
Figure 6: Robot for Performance Test 3. 37
Figure 7: Robot for Performance Test 4. 40
Figure 8: Robot for the individual competition. 40
Figure 9: Course strategy for the individual competition. 43
Figure 10: Drawing of the robot's chassis. 48
Figure 11: Drawing of the robot's drivetrain. 49
Figure 12: Electrical systems diagram. 49
Figure 13: Drawing of the robot's wrench mechanism. 50
Figure 14: Drawing of the robot's fuel crank mechanism. 51
Figure 15: Breakdown of purchases by category [4]. 53
Figure A1: Second-ranked course strategy. 62
Figure B1: Two dimensional sketches depicting Preliminary Idea One. 66
Figure B2: Two dimensional profile sketches of Preliminary Idea Two components. 66
Figure B3: Isometric sketch of Preliminary Idea Two. 67
Figure B4: Sketch depicting Preliminary Idea Three. 67
Figure C1: DC motor torque-speed curves with minimum specifications point. 70
Figure C2: Free-body diagram of the forces acting on the robot traveling up the course ramp. 71
Figure C3: Data log using RPS coordinates on MATLAB. 71
Figure E1: First build of the chassis and drivetrain. 75
Figure E2: Bottom view of the initial build. 75
Figure E3: Plastic spoons added as skids, replacing the back wheels. 76
Figure E4: Initial construction of wrench mechanism. 76
Figure E5: Four pieces of wood underneath chassis to level the robot. 77
Figure E6: 3D printed part for the fuel crank mechanism. 77
Figure E7: Robot with pronged 3D printed fuel crank mechanism. 78
Figure E8: Revised fuel crank mechanism for Performance Test 4. 78
Figure E9: Robot for final competition. 79
Figure E10: Robot's drivetrain. 79

Figure F1: Initial course strategy for Performance Test 1. 81
Figure F2: Second course strategy for Performance Test 1. 81
Figure F3: Final course strategy for Performance Test 1. 82
Figure F4: Course strategy for Performance Test 2. 82
Figure F5: Course strategy for Performance Test 3. 83
Figure F6: Initial course strategy for Performance Test 4. 83
Figure F7: Second course strategy for Performance Test 4. 84
Figure F8: Third course strategy for Performance Test 4. 84
Figure F9: Fourth course strategy for Performance Test 4. 85
Figure F10: Final course strategy for Performance Test 4. 85
Figure F11: Initial course strategy for the individual competition. 86
Figure F12: Final course strategy for the individual competition. 86
Figure I1: Graph of budget remaining over time [4]. 118

List of Tables

Table 1: Combinations of chassis and drivetrains, and mechanisms for the preliminary ideas. 20
Table 2: Measured dimensions of important course components 26
Table 3: Experimental shaft encoder data for intending to travel 6 inches 32
Table 4: Robot performance scoring guidelines. 44
Table 5: Overall timesheet. 57
Table A1: Screening matrix for chassis and drivetrain brainstorming ideas. 64
Table A2: Screening matrix for car jack mechanism brainstorming ideas. 65
Table A3: Screening matrix for fuel crank mechanism brainstorming ideas. 65
Table A4: Screening matrix for wrench mechanism brainstorming ideas. 66
Table A5: Concept scoring matrix for three preliminary ideas. 66
Table C1: Measurements of segmented navigation through robot course. 71
Table C2: Estimated times for robot course tasks. 71
Table C3: Estimated weights for robot components. 72
Table G1: Electrical connections table for GPIO ports 90
Table G2: Electrical connections table for Servo ports 92
Table G3: Electrical connections table for Motor ports 92
Table I1: Log of purchases. 120
Table J1: Design schedule. 122
Table J2: Weekly timesheet for Jan. 27 - Feb. 3. 123
Table J3: Testing log. 123

1. Introduction

Since the invention of the automobile, people from across the world have been fascinated by racing. With dozens of cars competing for the top spot, a precise and efficient pit crew is essential to an individual car's success. The difference between winning and losing can often be traced back to quick pit stops. Pit crews have become even more prominent in recent months, as NASCAR announced that over-the-wall pit crews will be reduced to five members instead of six during the 2018 season [1]. Even more drastic, the Formula EH (FEH) Grand Prix plans to utilize unmanned vehicles in the pit to automate tasks in its inaugural race. Not only will unmanned vehicles save money and eliminate safety concerns, but FEH executives also expect increased efficiency and public attention as a result [2]. FEH executives set up a competition to select a prototype from 63 entries to be used in the Formula EH Grand Prix. The Speed D8ers designed, built, and tested a prototype for the autonomous pit assistant in the Formula EH Grand Prix.

The project officially began on January 26, 2018, when FEH executives presented the problem to engineers. Justin Banke, Jack McGinness, Kelly Meaden, and Colby Williams formed the Speed D8ers company on January 31. During the subsequent two months, team members collaborated to create an autonomous pit assistant for the Grand Prix utilizing programming, computer-aided design (CAD), and other engineering skills acquired in previous coursework and experience. The Speed D8ers' robot was tested individually on March 30, and it competed head-to-head against sixty-two other vehicles on April 7, 2018. The company presented its work to the Ohio State Research and Development (OSURED) team, which selected the prototype to be used at the inaugural Formula EH Grand Prix [2].

The remainder of this report details the Speed D8ers' project from start to finish. The next section, Preliminary Concepts, presents the preliminary ideas and mockup that resulted from group brainstorming sessions. Section 3, Analysis, Testing, and Refinements, details the decisions made leading to the final design, as a result of analysis, calculations, and testing. Section 4, Individual Competition, describes the individual competition and the robot's performance. The final design is depicted in section 5, Final Design, with details of project management and specific features of the robot. Section 6, Final Competition, reports on the final competition and the robot's performance. Section 7, Summary and Conclusions, summarizes the project and provides suggestions for future work and possible enhancements. Finally, section 8 lists the documents referenced throughout the project and this report.

2. Preliminary Concepts

Before forming the company, the engineers individually brainstormed strategies to navigate the course, designs for the robot's chassis and drivetrain, and mechanisms to complete several tasks. Once the Speed D8ers formed, the team members combined their ideas and deliberated further to come up with preliminary ideas based on the project's requirements and constraints. Together, the team conceptualized three full-vehicle designs and then created physical and virtual mockups of the best design.

2.1 Project Requirements and Constraints

The robot was required to complete a number of tasks quickly and precisely to successfully aid pit crews at the Formula EH Grand Prix. OSURED constructed a scale model of the pit and garage area, which was available for use by the company during specified business hours. An overhead view of one section of the course is shown in Figure 1 below. The garage was elevated above the pit, and it was accessible by a grass ramp and a concrete ramp [2].

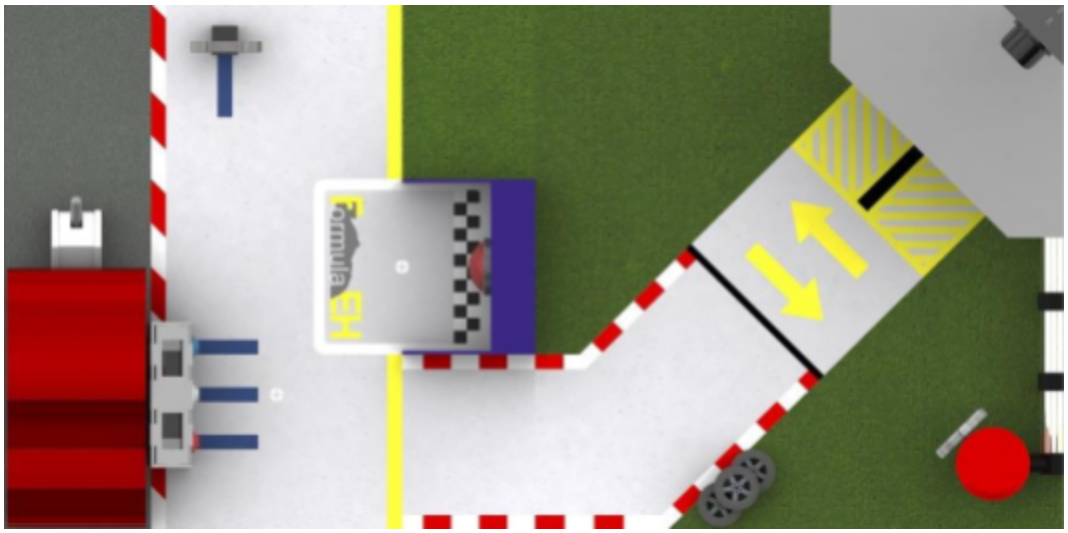


Figure 1: Overhead view of the pit area and garage, as provided by OSURED [2].

According to the project guidelines provided by the Formula EH executives, the following tasks had to be completed by the pit assistant in less than two minutes. First, the robot had to start when the light in the starting area beneath the robot was illuminated, thus clearing it to begin operations. From that point, the pit assistant had two minutes to complete the specified tasks before returning to the starting area and pressing the final charging button. Race cars often enter the pit to refuel with either 98 octane gasoline or nitromethane, so one primary task of the robot was to turn the fuel crank in the direction of the corresponding fuel type. During the pit

stop, the necessary fuel type for the race car was transmitted to the pit assistant wirelessly, and the robot was responsible for turning the crank 180 degrees clockwise for 98 octane gasoline or 180 degrees counterclockwise for nitromethane gasoline. The pit assistant was also responsible for clearing the area of any maintenance equipment. Thus, the robot had to pick up a wrench from the pit and transport it to the garage. Before the race car could leave, the pit assistant had to release it from the car jack used when changing tires. The final task was to perform a diagnostic test on the race car to test either its telemetry sensors, which allow the pit crew to monitor engine speed and provide a live video of the driver, or its electronic control unit (ECU), which controls the vehicle's numerous sensors. The race car only needed one of these tests during each pit stop, so the pit assistant had to scan a light in front of the control panel to determine which test to conduct before pressing the corresponding button. Once these tasks were completed, the robot had to return to the starting area to recharge [2].

The Speed D8ers considered each of the required tasks when developing a design for their prototype. Additional consideration was given to the Robot Positioning System (RPS), which wirelessly provided information regarding the robot's position in specified areas on the course. RPS was active in the pit during the entirety of each run, and communication could be activated on the upper level if an additional white button on the control panel was held for five seconds. The circular area around the garage, with a radius of 20", lacked a signal regardless, as indicated by the experimental guidelines [2].

The robot also had to meet several specifications, as indicated by the experimental guidelines. First, it had to fit within a 9" x 9" footprint before starting, and it could be no more than 12" tall. Its programmable Proteus microcontroller, which was lent to the company by

OSURED, could not be used as a sensor or structural support, and the only allowable adhesive was Velcro. The robot had to be fully autonomous, and a QR code had to be mounted 9” above the course surface for proper communication with the control systems. Finally, the budget for the prototype was $160 in addition to a set of basic sensors and the Proteus. Most parts, including various motors, chassis materials, and drivetrains, could be ordered from the FEH Company Store [2].

2.2 Brainstorming and Idea Screening

Individually, each team member came up with three course strategies, three chassis and drivetrain combinations, and three mechanisms for completing each task. After forming the company on January 31, team members discussed and compared ideas, and generated some new ideas as well. Next, the company screened the top ideas for each chassis and drivetrain combination and for several mechanisms. Each idea was rated on its ability to meet certain criteria, such as speed, cost, and durability. Screening matrices compared each idea to an average reference, with a “+” representing better performance than the reference, a “-” representing worse performance than the reference, or a “0” suggesting similar performance to the reference. To decide which concepts to proceed with, the company totaled the “+”s, “-”s, and “0”s to form a net score of each idea. This was completed with at least four ideas for the chassis and drivetrain, the car jack mechanism, the fuel crank mechanism, and the wrench mechanism. Each screening matrix can be found in Tables A1-A5 in Appendix A, and is discussed in detail in sections 2.2.2 through 2.2.6.

2.2.1 Course Strategy

The discussion regarding course strategy centered around three main ideas: starting with the buttons and moving clockwise around the course, starting with the wrench and moving counterclockwise around the course, or using two separate chassis connected by a wire to complete two tasks at a time. Additional deliberation focused on the following ideas: pressing the white button to activate RPS, depositing the wrench immediately after picking it up, and the possibility of pressing the final button from the upper level. Ultimately, the team settled on using only one chassis to limit complexity and cost. The top strategy suggested navigation in the following order: pressing the white RPS button followed by the red or blue button on the control panel, toggling the car jack, picking up the wrench, depositing the wrench, rotating the fuel crank, and pressing the final recharging button from the starting area on the lower level. This strategy is depicted in Figure 2 below. The second-ranked strategy, which is depicted in Figure A1 in Appendix A was similar, but it started with the wrench, moved counterclockwise, and finished by pressing the final button from the upper level.

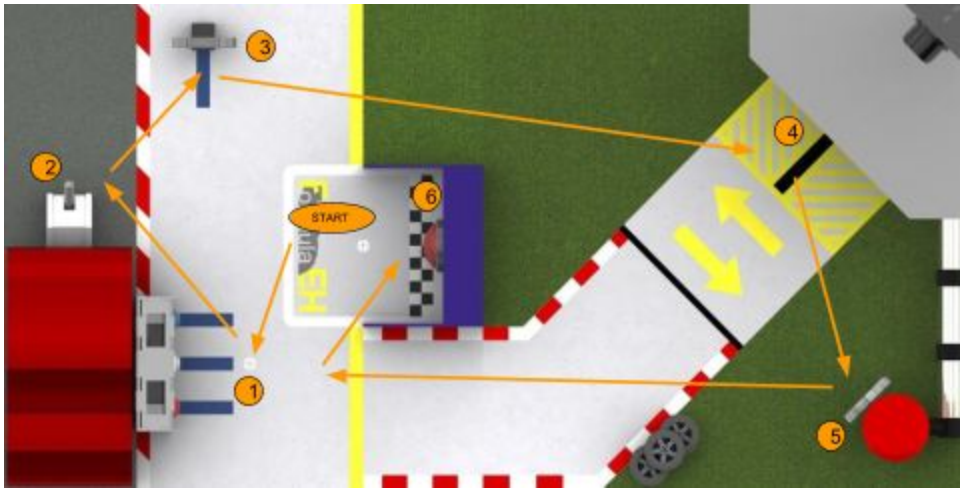


Figure 2: Top-ranked course strategy.

2.2.2 Chassis and Drivetrain

The discussion regarding the chassis and drivetrain centered around wheel orientation. The team came up with five chassis and drivetrain options: four powered wheels, two powered and two unpowered wheels, a circular layout with OMNI wheels perpendicular to the turning direction, two powered wheels and a chain attached to two unpowered wheels, or two robots each with a triangular frame. The five ideas were screened, as shown in Table A1 in Appendix A. As a result of the screening, the team chose to proceed with four powered wheels, two powered wheels and two unpowered wheels, and a circular wheel layout.

2.2.3 Car Jack Mechanism

The team came up with four ideas for the car jack mechanism. A sloped front could run into the jack or a lever could lift it. Also, team could construct an arm to clamp onto and lift the jack, or they could place a rod on a gear that could turn upwards to toggle the jack. Since the jack could be toggled with relative ease, the team focused on simplicity when screening the ideas. The screening matrix is shown in Table A2 in Appendix A. As a result of the screening, the team proceeded with the sloped front and lever ideas.

2.2.4 Buttons

The team brainstormed four ideas for pressing buttons on the control panel. The team considered using a narrow part on the front of the robot to press buttons, or adding a plate that extended out of the robot. The team also considered attaching three rods to the robot, and sliding

two of them forward to press the white RPS button and the specified test button, or simply running into the buttons with the front of the robot.

The team also brainstormed five ideas for pressing the final button. The team considered driving into the button with the chassis, driving into it with a slanted sled, pressing the button with mechanisms used for other tasks such as a rod or a gear, constructing an arm to slap the button from the upper level, or adding a plate that extends from the robot to press the button. Since the ideas considered primarily used mechanisms from other tasks, the ideas were not screened using a matrix.

2.2.5 Fuel Crank

The team came up with four mechanisms for rotating the fuel crank. Team members considered using a rod on a rotating gear that would go into the crank and spin. Using a rotating disk that would align with the crank and spin with it was also considered, or constructing an arm that could move up and down to spin the crank. A final idea was placing two rods on opposite sides of the gear to rotate the crank. The screening matrix for these ideas is shown in Table A3 in Appendix A. The team proceeded with the rod on a gear and the rotating disk after screening the ideas.

2.2.6 Wrench

The Speed D8ers brainstormed four mechanisms for controlling and depositing the wrench. The company considered building an arm that could slide through the hole of the wrench and turn or lift to pick it up, or adding a forklift to the front of the robot to carry the wrench. Team members also discussed the possibility of using a rod with a stopper that could enter the

hole and drag the wrench to the garage. A final idea was constructing an arm with clamping ability to grab the wrench. These ideas were screened using the matrix shown in Table A4 in Appendix A. The team proceeded with all of the ideas except for the arm with a clamp due to its cost and inefficiency.

2.3 Preliminary Ideas

Next, the Speed D8ers took the top ideas produced from each of the screening matrices and produced three combinations for a full robot layout. These combinations became the company's preliminary ideas for the robot layout. In their basic form, the combinations are shown in Table 1 below. Each combination is discussed in more detail in sections 2.3.1 through 2.3.3.

Table 1: Combinations of chassis and drivetrains, and mechanisms for the preliminary ideas.

Components	Idea One	Idea Two	Idea Three
Chassis and Drivetrain	aluminum, 2 motored wheels	PVC, 2 motored wheels + chain	wood, 4 motored wheels
Car Jack	Sloped Front	Lever	Lever
Fuel Crank	Rotating Disk	Rod on Gear	Two Rods on Gear
Wrench	Rod with Stopper	Arm In Hole	Forklift

The team used a concept scoring matrix, as shown in Table A5 in Appendix A, to rate each idea on its ability to meet specified criteria. In the concept scoring matrix, each combination was given a score from 1-5 for each criteria and compared to a reference concept, which received an average score of 3. Each criteria was weighted, so the importance of each requirement was reflected in each concept's score. For example, the company determined that maneuverability

was the most important criteria and should be 25% of the total score, while less prominent criteria were a smaller percentage. The weighted score for each criteria was calculated by multiplying its rating (on a scale from 1-5) by the criteria's percentage of the total score. Then, the team added the weighted scores to give each idea a total score. Finally, team members sketched each robot concept, and weighed the pros and cons of each. Then, the team produced a physical and CAD mockup of their top idea while considering the practicality of the preliminary concepts and making adjustments.

2.3.1 Preliminary Idea One

The first preliminary idea is depicted in Figure B1 in Appendix B. It consisted of a rectangular aluminum chassis with four wheels attached. The front two wheels were powered, giving the robot front wheel drive and limiting the cost associated with additional motors. In this design, half of the front face was a sloped ramp, which could lift the car jack by running into it. The other half of the front face consisted of a disk connected to a motor shaft so it could spin. When pressed up against the fuel crank, the motor could rotate the disk, thereby spinning the crank. On the top of the robot was a rod attached to a servo motor. The robot could drive so that the rod went through the hole of the wrench. The servo could then rotate upwards, lifting the wrench off of its stand. The stopper attached to the rod could catch the wrench and prevent it from getting too close the robot. The servo could rotate the rod downward to deposit the wrench. All of the buttons could be pressed by driving into them with either the ramp or the rod.

2.3.2 Preliminary Idea Two

Figures B2 and B3 in Appendix B illustrate the second preliminary concept. The chassis consisted of a PVC plate attached to four wheels. The front two wheels were powered by motors and the back two wheels were linked to the front two wheels via a sprocket and a chain. The car jack could be toggled using a servo motor attached to an arm that could lift the jack. The same mechanism was used for both the fuel crank and wrench. A DC motor was attached to a rod with two bends. The rod could slide into the holes of the crank and spin the appropriate direction, as measured by the motor encoder. This mechanism could also lift the wrench. The robot would drive so that the arm slides through the hole of the wrench. The arm could then turn, raising and holding the wrench until it was dispensed.

2.3.3 Preliminary Idea Three

Figure B4 in Appendix B represents the third preliminary idea. A square, wooden chassis was supported by four wheels, each one powered by its own motor and attached to the underside of the chassis. On the front of the chassis was a mechanism with the ability to function as a forklift that could spin in addition to moving vertically. This mechanism could complete all of the tasks on the course. The two prongs of the mechanism could be inserted into the holes of the wrench. The forklift could then be raised to lift the wrench off of its stand and lowered to deposit it in the garage. The prongs could also be raised to the appropriate height in order to press the buttons. Finally, the height of the forklift could be adjusted so that it fit into the fuel crank. The motor could then rotate 180 degrees in either direction to turn the fuel crank.

2.4 Mockup

After reviewing the concept scoring matrix, shown in Table A5 in Appendix A, the team initially decided to proceed with Preliminary Idea One, which is described in section 2.3.1. However, after further discussion of course measurements, team members decided to proceed with a single mechanism to complete all three tasks, as detailed in Preliminary Idea Three. Thus, the mock-up was a combination of the two preliminary concepts. Both a SolidWorks mockup and a physical mockup were constructed. The physical mockup was created using cardboard, paper clips, and tape. A picture is shown in Figure 3 below. It represented a rectangular aluminum chassis folded up on all four sides, four wheels (powered in the front), two motors, and the Proteus. The mechanism added to the front was able to both spin and slide vertically. It represented a rack and pinion connected to two servo motors. A plastic disk was also added to the back as an additional means of pressing the buttons.

Figure 3: Physical mockup.

A SolidWorks model was then constructed to represent the mockup digitally. Measurements were made to ensure that the SolidWorks design resembled the physical mockup closely. The SolidWorks model is shown in Figure 4 below.

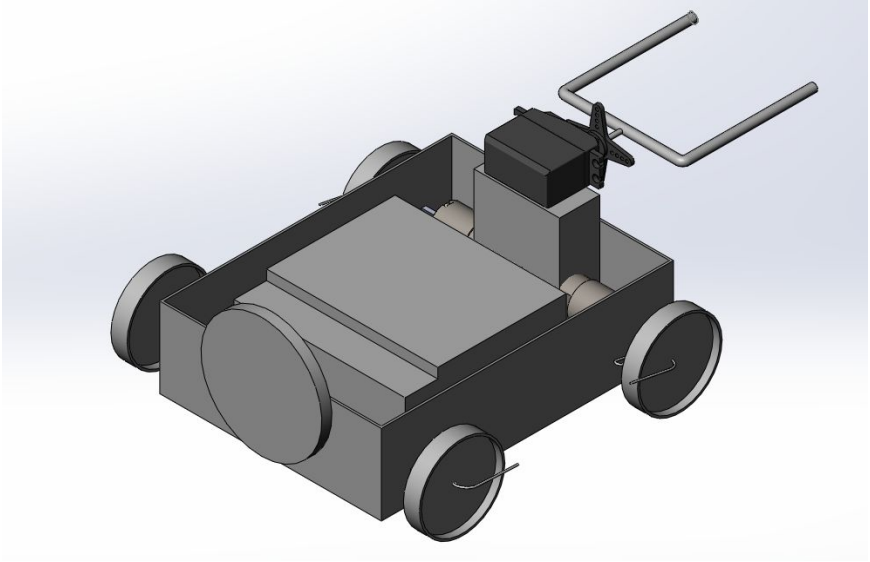


Figure 4: SolidWorks mockup.

2.5 Preliminary Code

While brainstorming ideas for the code, several different decisions were discussed. Most importantly, it was decided that a mixture of RPS and sleep statements would be used. The RPS would be utilized when the robot's position was crucial, as well as generally to make sure the robot was in the correct position. The sleep statements would be used to get the robot the majority of the distance to these positions, as the RPS would be slow. Sleep statements were decided on over encoders due to uncertainty over skidding at higher speeds. After some deliberation, a preliminary algorithm was made, shown on the following page.

Preliminary Algorithm:

1. Wait for the starting light to turn on
2. Drive forward until lined up with wrench
3. Turn right 90 degrees
4. Lower wrench servo
5. Move forward, grabbing wrench with servo
6. Raise wrench servo
7. Move backwards, lining up with jack area
8. Turn left 90 degrees
9. Move forwards, lining up with jack
10. Turn left 90 degrees
11. Move forwards, pushing jack up
12. Move backwards, lining up with ramp
13. Turn right 90 degrees
14. Move backwards until fully up ramp
15. Turn right until back of robot is lined up with fuel crank
16. Move backwards until flush with fuel crank
17. Rotate fuel servo based on the direction desired by the specified run
18. Move forward, lining up with garage
19. Turn right, lining up with garage
20. Move forward until flush with garage
21. Lower wrench servo
22. Move backwards until at the smooth ramp
23. Raise wrench servo
24. Turn right, lining up with ramp
25. Move backwards down the ramp until over button board light
26. Read the color of the light.
27. If the color was red, move at a right angle and push into red button
28. If the color was blue, move at a left angle and push into left button
29. Reset on light
30. Turn left 90 degrees
31. Move forward, lining up with final button
32. Turn right 90 degrees
33. Move forward until the final button is pressed

3. Analysis, Testing, and Refinements

Throughout the duration of the project, the Speed D8ers analyzed, tested, and refined the robot. Team members took measurements of the course, calculated the torque and speed required for the motor, and explored color sensing and filtering, line following and shaft encoding, and RPS navigation. The company modified the prototype's design and code as a result of testing on the course provided by OSURED. The robot completed four performance tests and participated in an individual competition before the final product competed head-to-head in the final competition on April 7, 2018.

3.1 Course Measurements

Initially, the dimensions of many important course elements were measured. These findings directly influenced the design of the robot.

Table 2: Measured dimensions of important course components

Course Component	**Dimension**
Distance between centers of holes of the wrench	3.5"
Distance between centers of white button and red/blue button	3.25"
Crank diameter (excluding spindles)	3"
Diameter of holes of wrench	0.875"
Height of buttons off of ground	1.9375"
Height of center of wrench holes off of ground	2.0625"
Height of center of fuel crank off of ground	3.0625"

The close proximity of the first three dimensions listed in Table 2 on the previous page induced the idea of an all-in-one mechanism consisting of two prongs spaced 3.25” apart. The prongs would fit through the holes of the wrench. They would also be used to simultaneously press the white button and the red or blue button, depending on the color of the floor light. Finally, the prongs would be spaced to that they would fit on the outside of the crank and spin it using the spindles. Importantly, the buttons, wrench, and fuel crank were measured to have different heights. The disparity in heights required a method of raising the prongs. It was decided that the prongs should be able to spin in order to turn the crank, and should be able to translate vertically to lift the wrench and to raise to the fuel crank’s height. Thus, the team decided to use a rack and pinion mechanism. This idea was used in the team’s mock-up, which is described in section 2.4.

However, the complexity of the all-in-one mechanism, combined with the difficulty in buying a rack and pinion from an external source, led to a change in design. Two separate prong mechanisms were created, one for the buttons and wrench and one for the fuel crank. The button and wrench mechanism was the appropriate height and could lift upwards. The fuel crank mechanism was placed higher on the robot and had 3D printed prongs that rotated.

3.2 Drivetrain Calculations

To determine which motors were capable of successfully driving the robot through the course, calculations were performed. First, the speed necessary for the robot to complete the appropriate tasks in less than two minutes was determined. The expected travel distance was found by adding the distance of each segment of the course navigation strategy. Table C1 in

Appendix C lists the distance of each segment, totaling 193". Next, the expected time needed to complete each action – excluding driving – was summed, as presented in Table C2 in Appendix C. This totaled to 26 seconds. After adding 9 seconds for buffer, it was determined that 35 seconds of the 2 minute round would be needed for tasks other than driving. This left 85 seconds to cover the 193 inches necessary to reach all of the tasks. The minimum linear speed of the robot was then calculated to be 2.3 inches per second using Equation 1 below. This calculation can be found in Appendix D.

$$v = \frac{distance}{time} \tag{1}$$

The respective minimum angular speed of the motor using a wheel of 1 inch radius was calculated to be 22 revolutions per minute using Equation 2, shown below. This calculation is shown in Appendix D.

$$\omega = \frac{v}{2\Pi r} \tag{2}$$

Next, the minimum torque provided by the motors to drive the robot up the ramp was calculated. By summing up the weight of each proposed component of the robot, an estimated weight of the robot was calculated. As demonstrated in Table C2 of Appendix C, the approximate weight of the robot was found to be 1059 g. This was converted to 37.36 ounces using Equation 3 below. This calculation can be found in Appendix D.

$$ounces = grams \cdot \frac{1\ oz.}{28.35\ g} \tag{3}$$

Figure C1 in Appendix C illustrates the forces acting on the robot as it drives up the ramp. The internal friction of the robot was assumed to be 8 oz. Using the measured height of the ramp, 3.125", and the measured length of the ramp, 9", the incline was found to be 19.15^{o}. In

order for the motor to successfully propel the robot up the ramp, the force delivered by the motors had to at least equal the counter forces parallel to the ramp, weight and friction. Equation 4 below was used to calculate the required force of the motors. By the calculation shown in Appendix D, the force was found to be 20.26 ounces.

$$F_{motors} = F_{friction} + W sin(19.15) \tag{4}$$

Equation 5, found below, was used to calculate the torque necessary to deliver 20.26 oz of force using a 1" diameter wheel. The necessary torque was 20.26 ounce-inches, as found with the calculation shown in Appendix D.

$$\tau_{motors} = F_{motors} \times r_{wheel} \tag{5}$$

Because two motors would be used to drive the robot, this torque was halved; each motor would have to exert a minimum torque of 10.13 oz-in. Six motors were offered to power the robot: Igwan, Acroname, VEX 393, GMH-34, FUTABA Servo, FITEC Servo. The minimum specifications of the motor for the robot – a speed of 22.0 revolutions per minute and a torque of 10.13 oz-in – were plotted on a Torque-Speed Curve for the six motors. This is illustrated in Figure C2 in Appendix C. Because the point lays below the torque-speed curves of all six motors, a pair of any of the motors would be capable of driving the robot up the ramp and to all of the tasks in less than two minutes. However, Igwan motors were chosen as the preferred motor due to its reliability, built-in encoders, and efficient attachment options.

3.3 Explorations

Three explorations were conducted to further the general knowledge of the team and provide potential methods to improving the success of the robot.

3.3.1 Exploration 1

The first exploration investigated the use of motors and sensors in the robot competition. A green filter paper was placed over a CdS cell, which is a small sensor used to detect the color of emitted light, to experiment in differentiating between a blue light, a red light, and the floor of the course. The CdS cell voltage reading for these three surfaces was recorded. The Proteus was then programmed so that a servo motor would be in position 0 when the CdS cell was in full light, position 180 when the CdS cell was in complete darkness, and any corresponding position in between. The Proteus was then wired to a Crayola Bot and programmed to navigate the robot through a simple maze using four bump switches.

The green filter proved to be unhelpful in distinguishing the red and blue lights; the values were in close proximity since green is close in color to both red and green. This result suggested the use of a blue or red filter, which would differentiate the readings better. The servo motor successfully represented the intensity of light received by the CdS cell and demonstrated its ability to hold a specific position. The Crayola Bot successfully navigated the course. The Igwan motors smoothly translated the robot and the bump switches were essential in order to turn after hitting each wall. Minor issues were faced when the robot backed up after hitting a wall. Occasionally, one bump switch on a side would be pressed, but the robot continued turning

into the wall in a way such that the second bump switch on that side would never be pressed. Consequently, the robot would continue to be stuck on the wall indefinitely, waiting for the second bump switch to be pressed. This was resolved by editing the code. In this situation, the wheel opposite of the pressed bump switch was reversed.

Ultimately, the team decided to use a CdS cell, taped underneath the chassis, to read the lights on the course. No filter was used, as the CdS cell adequately portrayed the difference in voltage between red and blue light. The Speed D8ers opted against using bump sensors on the robot, as straightening into the wall was adequate while testing.

3.3.2 Exploration 2

The second exploration focused on line following and encoder-based navigation. Initially, the readings of an optosensor on a black line and on the background around it were recorded. The Proteus was wired to a Crayola Bot and programmed to follow a straight line using the optosensor. This code was enhanced to allow following of a line with curves. Calculations were also made to convert shaft encoder counts to inches travelled by the robot. Equation 6, shown below, was used to determine that there were 40.489 encoder units per inch that the robot travels, given that there were 318 encoder units per rotation and the wheel diameter was 2.5". This calculation is shown in Appendix D.

$$\frac{encoder\ units}{1\ inch} = \frac{encoder\ units}{rotations} \cdot \frac{rotations}{\Pi d\ inches} \tag{6}$$

The Proteus was then programmed to make the Crayola Bot travel 6 inches at three different motor power percents: 25%, 40%, and 60%. The experimental encoder values for the left and

right motors were recorded after each trial. Finally, the Proteus was coded so the Crayola Bot would drive a specific sequence.

The optosensor was successfully used to follow the straight and curved lines. However, it was most successful following a straight line and at lower motor percents. The success demonstrated the potential benefit of following the lines on the robot course. For all three trials of travelling six inches, the experimental encoder values were different than the theoretical and not equal between the left and right motors. Table 3 below describes the experimental encoder values of the right and left motor shafts when set to travel 243 encoder units, for a theoretical distance of 6 inches.

Table 3: Experimental shaft encoder data for intending to travel 6 inches

Motor Percent	Experimental Distance Travelled (inches)	Experimental LE Counts Travelled	Experimental RE Counts Travelled
25%	5 ⅞”	256	250
40%	6 5/16”	276	270
60%	7 ⅛”	301	299

This was explained by the motor shafts continuing to turn after power was ceased. The discrepancy increased as motor power increased, revealing the greater accuracy presented by travelling at slower speeds. Additionally, the disparity between the right and left motors illustrated the slight inaccuracy of encoder based navigation.

While the Crayola Bot was able to perform the specific navigation sequence with little error, the team elected to primarily rely on time-based navigation rather than shaft encoding. Furthermore, the robot was able to press the buttons and pick up and deposit the wrench without

following lines, so the team determined that line following was not necessary to complete the required tasks. Thus, the robot did not utilize optosensors.

3.3.3 Exploration 3

The final exploration explored the Robot Positioning System (RPS) and data logging and their ability to enhance the success of the robot. A QR code was placed on the top of a Crayola Bot in order to be read by the RPS above the robot course. The RPS x and y coordinates sent to the Proteus were recorded at four specific locations on the robot course. The Proteus was then wired to the Crayola Bot and programmed to use RPS to check and adjust the robot's position after travelling measured distances using encoders. The driving motors adjusted the robot until its x position, y position, and heading were all within 1 unit of the intended position. Next, the position of the robot was logged on the microSD card every 10 milliseconds during an RPS enhanced sequence on the course. These positions were then plotted on a picture of the robot course using MATLAB, as shown in Figure C3 in Appendix C.

RPS proved to be very helpful in correcting encoder based movements. MATLAB properly used to map out the movement of the robot over an image of the robot course. This system of data logging offered a helpful way of troubleshooting robot issues. Data stored during the run could be analyzed to determine which sensor, motor, or movement was responsible for causing problems on the run.

At first, the team only used RPS to receive the signal regarding which fuel type was necessary, but during Performance Test 4 testing, the team elected to rely on RPS for navigation. Team members decided not to utilize data logging.

3.4 Testing

The OSURED team requested that each company complete four performance tests to ensure that the prototypes were making satisfactory progress. OSURED also held an individual competition on March 30 and a final competition on April 7 to examine each robot's final design and performance. The Speed D8ers conducted numerous tests on the course leading up to each performance test and competition. The testing often resulted in changes to the physical design and to the code.

3.4.1 Performance Test 1

For the first performance test, the robot was required to maneuver to the car jack, toggle the car jack to remain in the up position, and drive up a ramp to get to the upper level by February 23 [3]. Before testing, team members constructed the chassis and drivetrain to include a 6.5" by 7.5" wooden chassis, two IGWAN motors secured underneath the chassis with 3D printed mounts, two Du-Bro wheels connected to the motors with 3D printed pieces, and two additional Du-Bro wheels in the rear mounted with double bent strips and 3" axle rods, and secured in place with shaft lock collars. A picture of the robot in this stage can be found in Figure E1 in Appendix E. A CdS cell was taped to the bottom of the chassis so the robot could start when the course light was activated, as shown in Figure E2 in Appendix E.

Testing began on February 20, when team members began checking the accuracy of time-based movement. Team programmers defined values for high, medium, and low motor power, and created functions for moving forward based on specified power percentages and

time-based distances, and for turning based on specified angles. Using trial-and-error methods, the team determined distances to input into drive functions in the code in order to navigate successfully to the car jack. While testing on February 20, the team used the course strategy depicted in Figure F1 in Appendix F. The robot was programmed to move forward once the starting light was activated, turn right 90 degrees after tapping the control panel, turn left before the wrench, and then hit and align with the wall. While in the corner between the jack and the wrench, the robot made numerous 90 degree turns and aligned with both walls to straighten out before and after toggling the jack. Then, it proceeded up the grass ramp to complete the final performance test task. The robot was able to navigate using this method, but it was very inconsistent.

In an attempt to improve the consistency, team members tried using shaft encoding to navigate with the same course strategy, which is described above. After a couple tests with shaft encoding, the team remained unsatisfied with the robot's consistency. Throughout the testing period, team members noticed that the back wheels were loose, and the shaft lock collars had to be re-tightened several times. Finally, the team decided to remove the back, unpowered wheels, and test the robot on the course. The robot finally moved consistently using makeshift skids rather than wheels. The extra friction caused by the back wheels was causing the inconsistency during trials, so the Speed D8ers proceeded without the wheels. From then on, plastic spoons were used as skids, as shown up close in Figure E3 in Appendix E. The chassis and drivetrain used in Performance Test 1 is shown in Figure 5 on the following page.

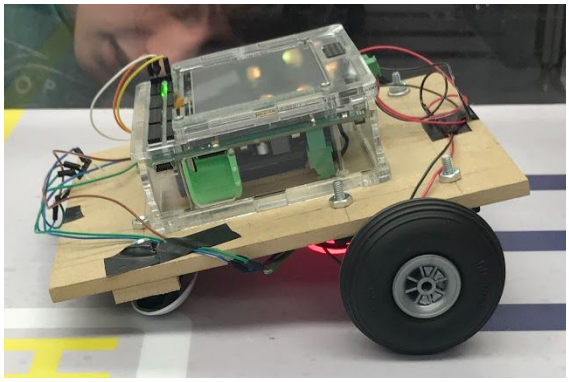


Figure 5: Robot for Performance Tests 1 and 2.

On February 21, the Speed D8ers refined the time-based program and completed additional tests with the newly-secured skids. The team conducted an official performance test using the course strategy depicted in Figure F1 in Appendix F. The robot got caught on the wall beside the ramp, and failed to complete any tasks other than touching the ramp. The team shifted its strategy thereafter to go up the other ramp instead to avoid unnecessary tight turns. This strategy is depicted in Figure F2 in Appendix F. The team used trial-and-error to come up with the correct distances to input into the sleep statements in the time-based code. Then, a second official run was conducted. The course strategy seemed to work successfully, but the jack was not toggled to remain in the up position. Team programmers quickly adjusted the distance the robot traveled to run into the jack, and a third official test was conducted. Unfortunately, the robot got caught on the wall surrounding the starting area and was not able to make it to the concrete ramp. As a result, the team re-evaluated the course strategy and decided that the tight 90 degree turns left little room for error.

On February 22, team members adjusted the program to follow the course strategy depicted in Figure F3 in Appendix F. The robot then made 45 degree turns to and from the

corner near the jack to limit the number of tight turns. This code was tested several times for consistency, and then officially tested for OSURED on February 23. The performance test was successful.

3.4.2 Performance Test 2

On February 22, team programmers began working on the code for the second performance test. In this test, the robot was required to read and display the correct color of the diagnostic light, press the corresponding diagnostic button, and press the final button by March 2 [3]. The robot design remained the same from Performance Test 1, as depicted in Figure 5 on the previous page.

The course strategy for this performance test is depicted in Figure F4 in Appendix F. First, the robot turned right to touch the wrench (which proved that the team was ahead of schedule for bonus consideration from OSURED), then it moved back to align with the wall beyond the control panel. The robot then drove over the diagnostic light to read it, turned left toward the control panel, and drove into the white RPS button while turning into the correct diagnostic button. To accomplish this task, team programmers created a function to read the value detected by the CdS cell, which read less than 0.80 V if the light was blue. If the function returned a “true” value, suggesting that the diagnostic light was blue, then a separate function programmed the robot to turn into the blue button. Otherwise, if the CdS cell read greater than 0.80 V, the robot was programmed to turn into the red button. After the diagnostic button was pressed, the robot backed up and turned toward the final button to finish the performance test.

On February 23, the team tested the consistency of the code and noticed that the red button was not completely pressed. Team programmers revised the code to increase the power of the turn into the red button. Then, the team conducted an official test for the OSURED team. On the first test, the diagnostic light was blue, which the robot successfully detected and displayed on the Proteus screen. However, it did not press the button with enough power. The team conducted a second official test, where the diagnostic light was red, and it worked successfully.

3.4.3 Performance Test 3

On February 26, the team began working on the third performance test. In this test, the robot was required to pick up the wrench and deposit it in the garage by March 9 [3]. The Speed D8ers used the course strategy depicted in Figure F5 in Appendix F, moving directly from the starting position to the wrench, then directly to the garage, and finally to the fuel crank.

In order to control the wrench, the robot needed an additional mechanism. Team members constructed a pronged mechanism made out of a double angle strip and popsicle sticks, which was controlled using a Futaba servo motor. The servo motor was taped to the front of the robot on top of the wooden chassis for the test. Initially, the popsicle sticks were attached to the double angle strip with screws (as shown in Figure E4 in Appendix E), but the sticks did not stay in place, so electrical tape was found to be a better adhesive. The team also leveled the robot, which was previously angled following the removal of the back wheels, by adding four 6.5” x 1” x 0.25” pieces of wood underneath the chassis. This is depicted in Figure E5 in Appendix E. Finally, a QR code stand was added using two 5.5” double angle strips and additional wood. A picture of the robot in this stage is shown in Figure 6 on the following page.

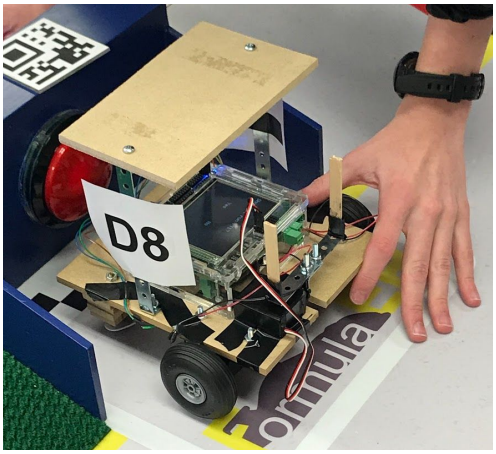


Figure 6: Robot for Performance Test 3.

Team programmers continued to rely solely on time-based movement to complete this test. Trial-and error was used to determine correct distances for the sleep statements within the code, so numerous tests were conducted on March 1 to ensure that the distances to the wrench and the garage were precise. Additional testing was performed to adjust the movement of the servo to the correct height of the wrench when grabbing it, to ensure that an appropriate power level was used to get up the ramp, and to avoid hitting the control panel.

On March 2, the team tested the consistency of the program before conducting an official test. On the first official test, the robot's wrench mechanism did not successfully enter the holes of the wrench to grab it. The Speed D8ers conducted a successful unofficial test before running a second official test. In the second official test, the robot was too far to the right when attempting to deposit the wrench, so the wrench was only partially deposited. Finally, on the third official test, the robot performed all necessary tasks successfully and even touched the fuel crank to earn bonus consideration from the OSURED team. Following the test, team members discovered that one possible source of error in failed runs was that the capacitors on the IGWAN motors were

bent and rubbing against the motors. This possibly caused the left and right motors to run inconsistently. As a result, team members became more vigilant in checking the motors.

3.4.4 Performance Test 4

The Speed D8ers began working on the fourth performance test on March 7. For this test, the robot was required to read in the fuel type with RPS and spin the crank 180 degrees in the corresponding direction by March 23 [3]. The team added a mechanism to turn the fuel crank in order to complete the test. They designed a 3D printed part with a circular base and two prongs to fit inside the fuel crank, as shown in Figure E6 in Appendix E. The part was screwed into a disk that attached to a Futaba servo motor. The motor was taped to three 2.5”x1.5” pieces of wood that were screwed onto the chassis to ensure that the mechanism was at the necessary height to spin the fuel crank. A picture of the robot in this stage is shown in Figure E7 in Appendix E.

The Speed D8ers began testing the code for Performance Test 4 on March 9 using the course strategy depicted in Figure F6 in Appendix F, where the robot took the concrete ramp to the upper level and followed the concrete path before turning towards the fuel crank. The team relied on time-based movement, and conducted numerous tests on the course to determine the necessary distances, angles, and power levels needed to complete the performance test. Overall, the team decided that using the grass ramp would lead to more consistent runs, so the course strategy depicted in Figure F7 in Appendix F was tested. During initial trials using the modified code for the new course strategy, the robot bumped into the wrench. Team members then decided to use the code for Performance Test 3 to get up the ramp with 45 degree turns rather

than 90 degree turns. Team programmers modified the code from the third performance test to resemble the course strategy depicted in Figure F8 in Appendix F. The team needed to align the robot to avoid the slight dip in the course on its way to the fuel crank before conducting an official test. While testing, the team noticed that the prongs on the fuel crank mechanism were hindering its ability to turn the crank when the robot was not exactly aligned. Thus, the team removed the prongs from the mechanism and added a rubber ping pong paddle cover to the disk for increased friction. This is depicted in Figure E8 in Appendix E.

The robot did not go up the grass ramp consistently during tests, so team members altered the route to make the robot straighten up against the wall before turning towards the fuel crank. The servo also had trouble rotating the crank the full 180 degrees, so team programmers had the robot move backwards after the first rotation, reset, and spin the crank again. This strategy is depicted in Figure F9 in Appendix F. The team tested the route for consistency on March 21 before conducting official tests on March 23. In the first official test, the robot spun the crank adequately on the first rotation, but then proceeded to spin the crank further, exceeding 180 degrees. Team members removed the second spin, resulting in the course strategy shown in Figure F10 in Appendix F. The team also removed the electrical tape that was holding the servo motors in place and hot-glued them to the chassis. The robot is shown testing for Performance Test 4 in Figure 7 on the following page.

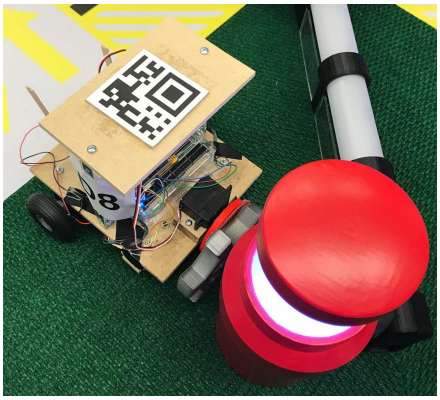


Figure 7: Robot for Performance Test 4.

The robot did not consistently spin the crank 180 degrees, but the team conducted two official tests using the revised code. On the first attempt, the crank failed to rotate the full 180 degrees, but the robot successfully completed Performance Test 4 on the second attempt.

3.4.5 Individual Competition

Before the individual competition, team members hot-glued the popsicle sticks to the double angle strip and the plastic spoons to the double bent strips, labeled the various wires with tape, and added sponsor logos to the side of the chassis. The robot in this stage is shown in Figure 8 below.

Figure 8: Robot for the individual competition.

The Speed D8ers completely altered the code for the individual competition. Instead of relying solely on time-based navigation, the team utilized RPS to run the course during competition. During initial testing on March 26, the team used the course strategy depicted in Figure F11 in Appendix F, where the robot only utilized RPS on the lower level of the course and went to the control panel at the end of the run. However, team members decided that RPS would lead to greater consistency on the upper level, so they switched to the course strategy depicted in Figure F12 in Appendix F on March 28, where the control panel was the first destination. The strategy is discussed in detail in section 4.1.

Tests preceding the individual competition were focused on determining x and y coordinates for the RPS checks. Team members first tested with many RPS checks to get to the desired location, leading to very slow runs. Then, team programmers added time-based code for most movements and RPS heading and location checks before turns. The individual competition is discussed in more detail in section 4.

3.4.6 Final Competition

Following a strong showing during the individual competition, the robot did not need many additional refinements before the final competition. The team added black paper and the team logo to the chassis so the robot's physical appearance was enhanced for the competition, as shown in Figure E9 in Appendix E. The code on the upper level, with and without RPS, was improved so the robot aligned with the fuel crank consistently. The final competition is discussed in detail in section 6.

4. Individual Competition

On March 30, 2018, the Speed D8ers' pit assistant prototype competed three full-course runs in an individual competition in Hitchcock Hall. First, the OSURED representatives selected the course, fuel type, and electrical test for the prototype. Then, the course, fuel type, and electrical test were chosen at random. For the final run, the team chose the course, fuel type, and electrical test. OSURED representatives scored each robot based on the guidelines outlined in Table 4 below [2].

Table 4: Robot performance scoring guidelines.

Primary Tasks	**Points**	**Secondary Tasks**	**Points**
Initiate on start light	9	Deposit wrench	8
Touch wrench	8	Rotate fuel pump crank in correct direction	10
Control wrench	12	Rotate fuel pump crank 180 degrees	7
Toggle jack lever	10	**Possible Secondary Task Points**	**25**
Rotate fuel pump crank	8		
Press and electrical test button	7	**Total Possible Task Points**	**100**
Press the correct electrical test button	12		
Press the final charging button	9	**Penalties**	
Possible Primary Task Points	**75**	Interfering with a competitor's robot	DQ

4.1 Strategy

The Speed D8ers decided that the most efficient course strategy was to begin with the button control panel, pressing both the electrical test button and the RPS button while there.

Then, the robot aligned with the wrench using RPS heading and coordinate checks, and moved forward to pick it up. Then, it moved back, turned left toward the wall, and turned left again before driving into the car jack. After toggling the jack, the robot backed up and turned toward the grass ramp. It made a slight turn to line up with the center of the ramp, and drove to the upper level. Then, the robot turned 45 degrees to the fuel crank. After spinning the crank, it backed up and turned 90 degrees to deposit the wrench in the garage. Then, the robot took the concrete path back to the lower level and finished the course by pressing the final charging button. This strategy is depicted in Figure 9 below.

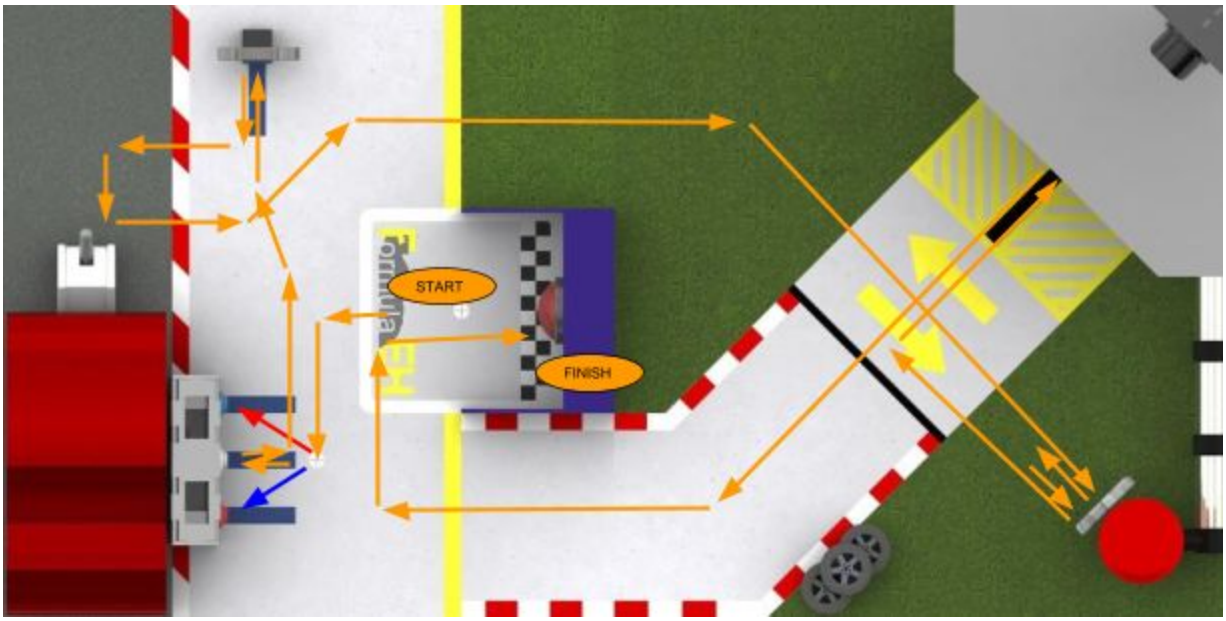


Figure 9: Course strategy for the individual competition.

4.2 First Trial

In the first run at the individual competition, OSURED representatives selected Course H, nitromethane fuel (corresponding to a counter-clockwise turn of the fuel crank), and telemetry sensors test (corresponding to the blue button). The Speed D8ers' robot successfully initiated on

the start light before driving toward the diagnostic light. The robot incorrectly read the light, and it proceeded to press the red button instead of the blue button. It did press the RPS button, thus activating RPS on the upper level. The robot successfully aligned with, touched, and picked up the wrench before toggling the car jack. Then, it went up the ramp and headed toward the fuel crank. However, the prototype lined up too far to the right of the crank and turned it slightly in the incorrect direction. Then, it backed up and successfully deposited the wrench in the garage. The robot was unable to press the final button, so the team ended the run with the kill switch after 1 minute, 19 seconds, and 68 milliseconds. Overall, the robot earned 54 points for completing primary tasks and 8 additional points for completing secondary tasks, for a total score of 62 points.

Following the run, the Speed D8ers increased three angles for the turns leading up to the final button. Team members also noticed that the prototype did not use RPS on the upper level even though it was activated. Programmers located and corrected the error in the code that caused this problem. Although the robot failed to detect the blue diagnostic light, the team did not attempt to correct this error during the individual competition.

4.3 Second Trial

In the second run at the individual competition, Course G, 98 octane fuel (corresponding to a clockwise turn of the fuel crank), and the Electrical Control Unit (ECU) test (corresponding to the red button) were randomly selected. The robot successfully initiated on the start light before driving toward the diagnostic light. It correctly read the light and pressed the red button, but not the RPS button. The robot successfully aligned with, touched, and picked up the wrench

before toggling the car jack. Then, it went up the ramp and headed toward the fuel crank. However, the prototype again lined up too far to the right of the crank and turned it slightly in the incorrect direction. Then, it backed up and successfully deposited the wrench in the garage. The robot successfully pressed the final button, but it rubbed against the wall on its way down the concrete ramp. The run lasted for 1 minute, 12 seconds, and 10 milliseconds. Overall, the robot earned all 75 points for completing primary tasks and 8 additional points for completing secondary tasks, for a total score of 83 points.

Following the run, the team added two inches to the distance traveled on the concrete path before turning to go down the ramp. This was done to avoid rubbing against the wall. The team also adjusted the distance traveled to get to the red button from the diagnostic light so the robot would be closer to the white button. Otherwise, the Speed D8ers were satisfied with the code and the robot's performance.

4.4 Third Trial

In the final run at the individual competition, the team selected Course G, 98 octane fuel (corresponding to a clockwise turn of the fuel crank), and the ECU test (corresponding to the red button). The robot successfully initiated on the start light before driving toward the diagnostic light. It correctly read the light and pressed the red and white buttons, thus activating RPS on the upper level. The robot did not align with the wrench enough to pick it up, but the wrench was touched and knocked over. Then, the robot toggled the car jack before driving up the ramp and turning toward the fuel crank. However, the prototype lined up too far to the right of the crank using RPS. It did turn the crank slightly in the correct direction. Then, it backed up and drove to

the garage, but it did not have anything to deposit. The robot successfully pressed the final button, but it ran into the blue diagnostic button after coming down the ramp, thus negating its points for pressing the correct electrical button earlier in the run. This run lasted for 1 minute, 22 seconds, and 72 milliseconds. Overall, the robot earned 63 points for completing primary tasks and 10 additional points for completing secondary tasks, for a total score of 73 points.

4.5 Modifications

Following the individual competition, the Speed D8ers considered implementing several modifications. First and foremost, the robot failed to align with the fuel crank on all three trials during the competition, so the team needed to modify the code on the upper level with and without RPS. The team also needed to modify the code on the robot's route from the garage to the final button due to inconsistencies and errors made during the individual competition. Previously, the team did not use RPS on this route, but team members considered utilizing the technology to maximize precision. The robot did not consistently pick up the wrench, so the team considered modifying the course strategy to pick up the wrench first, toggle the fuel jack, and then head to the control panel. The robot also struggled to press the white button to activate RPS consistently. The Speed D8ers discussed pressing the buttons with the fuel crank mechanism to increase the area pressing the buttons and potentially increase consistency. Finally, team members noticed that the left IGWAN motor was tilted, so they planned to address that area of concern before the final competition.

5. Final Design

Throughout the duration of the project, the Speed D8ers modified the robot's physical design and code to create a competitive prototype for the final competition. Both the design and code had strengths and weaknesses. On the way to the final design, the team monitored the budget and followed a schedule, while also logging the time spent working on each area of the project.

5.1 Features

As the project progressed, changes were continually made to the original, brainstormed designs, culminating in a fully functional final design. The final chassis, drivetrain, electrical systems, and mechanisms were put to the test at the final competition.

5.1.1 Chassis

The chassis of the final designed was made of MDF Engineered plywood. Its dimensions were 6.5" x 7.5". The material provided an affordable and sturdy structure, and the scrap wood was repurposed for other supporting roles on the robot. Attached to the wooden chassis were four wooden beams supporting the two wooden spoons. IGWAN motor mounts were screwed into the chassis from the underside. A CdS cell was taped to the underside of the chassis in order to detect the lights on the playing surface. Two long erector pieces connect the chassis to a rectangular piece of wood raised approximately 8" above the course surface. The QR code was mounted to the this upper wooden plate. The two servo motors for the mechanisms were glued to

the front and back of the chassis. Finally, the Proteus sat on the top side of the chassis, surrounded by screws and other components so that it could not fall out. This is shown in Figure 8 on page 31 and illustrated in Figure 10 below.

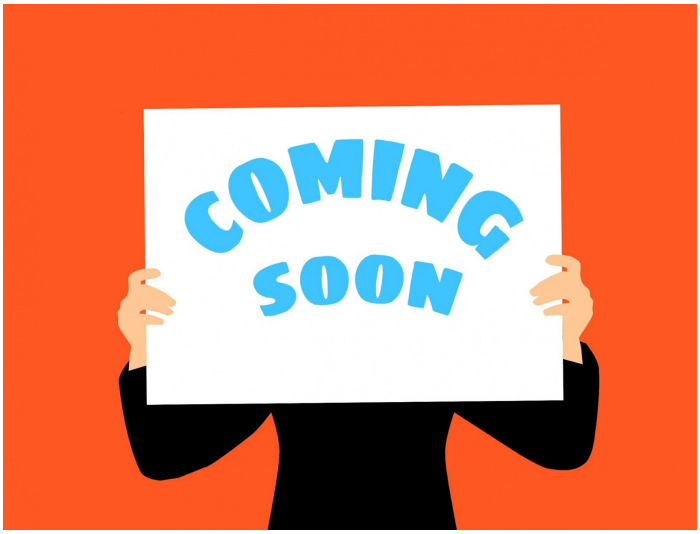


Figure 10: Drawing of the robot's chassis.

5.1.2 Drivetrain

The final drivetrain consisted of two Igwan motors attached to two 2.5" diameter DuBro wheels. The proposed non-powered back wheels were replaced by plastic spoons serving as skids. The spoons experienced relatively low levels of friction with the field surface. The skids were taped and glued to double bent erector pieces, where were screwed into four wooden beams lowering from the underside of the chassis. A picture of the drivetrain is shown in Figure E10 in Appendix E, while a drawing of the drivetrain is displayed in Figure 11 on the following page. Both front-wheel and back-wheel drive were implemented, with back-wheel drive being significantly advantageous when travelling up the ramp.

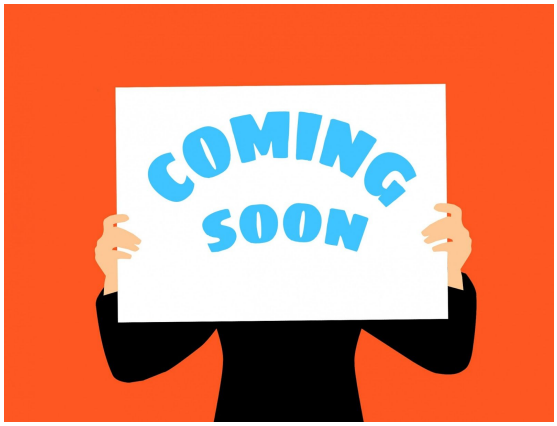


Figure 11: Drawing of the robot's drivetrain.

5.1.3 Electrical System and Sensors

A total of three sensors were involved in the final design. A CdS cell was taped to the underside of the chassis. The cell was used to detect the red start light, initiating each run, and to determine the color of the floor light corresponding to the red or blue buttons. Also, both Igwan motors contained built-in shaft encoders. The encoders were wired to the Proteus, although they were used sparingly. The electrical connections for the GPIO, Servo, and Motor ports can be found in Tables G1, G2, and G3 in Appendix G, respectively. Each wire was labeled and a diagram displaying the Proteus and each connection can be seen in Figure 12 below.

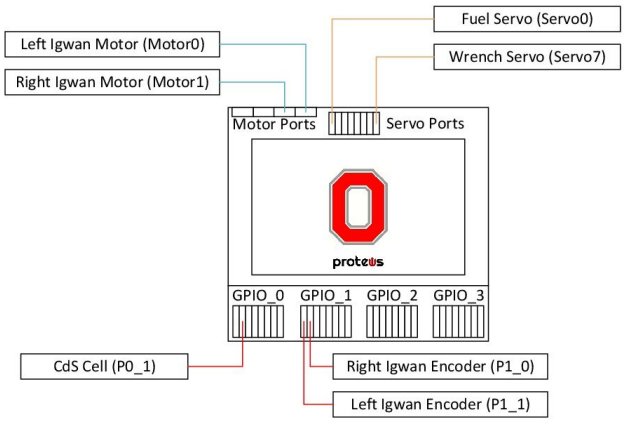


Figure 12: Electrical systems diagram.

5.1.4 Mechanisms

Two mechanisms were used to complete the necessary tasks on the robot course. The first mechanism was used to transport the wrench. It consisted of a servo motor attached to a double-bent erector piece. Two half popsicle sticks were glued to the erector piece, creating a two-pronged fork. The prongs fit through the holes of the wrench, and the servo would rotate to lift the wrench. Once the robot reached the garage, the servo would lower, causing the prongs to push the garage door open and the servo to drop the wrench. The mechanism was attached to the front of the chassis by gluing the servo to the wooden frame. A picture of the mechanism can be found in Figure 6 on page 28, while a drawing of the wrench mechanism is depicted in Figure 13 below.

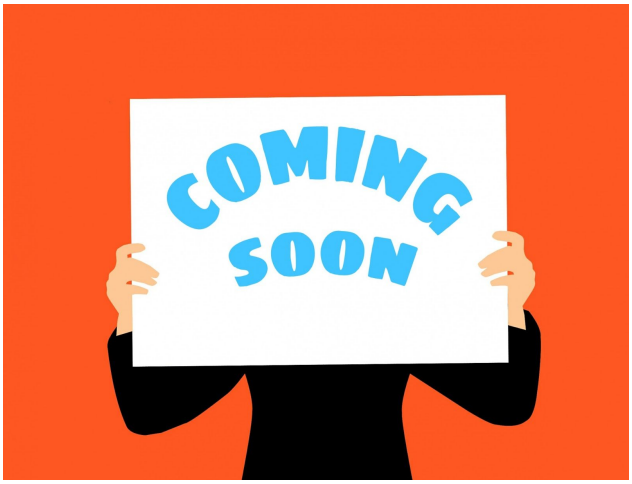


Figure 13: Drawing of the robot's wrench mechanism.

The second mechanism was used to spin the fuel crank. It consisted of a servo motor attached to a 3D printed circular piece. A circular patch from a ping-pong paddle was adhered to the outer surface of the disk to increase the friction between the disk and the fuel crank. Before

the robot reached the fuel crank, the servo would rotate to the left or right to allow it 180 degrees of motion in the correct direction, according to the type of fuel required. Once pressed up against the fuel crank, the servo would rotate, thereby spinning the fuel crank. A picture of the mechanism can be found in Figure E8 in Appendix E, while a drawing of the fuel crank mechanism can be seen in Figure 14, shown below.

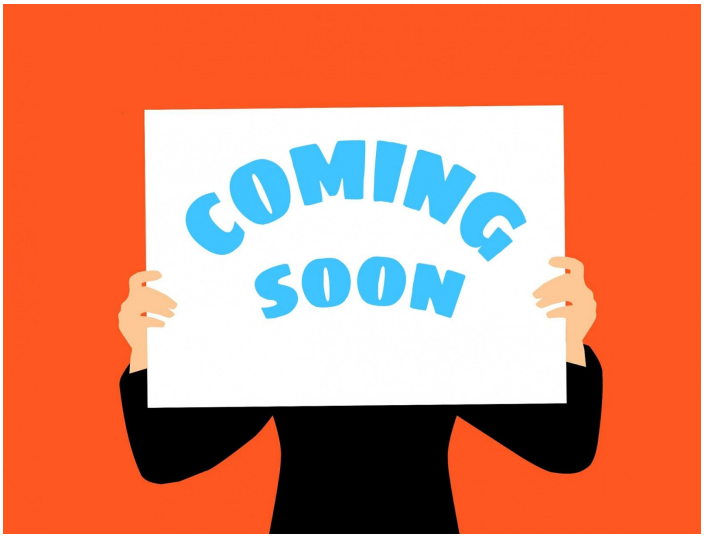


Figure 14: Drawing of the robot's fuel crank mechanism.

Mechanisms were not needed for the car jack or any buttons. The car jack was raised by simply driving the robot into it. The buttons were pressed by driving the robot into them with the edge of the wooden chassis.

5.2 Code

The code for the robot was created using QT Creator and the Proteus, and it is provided in Appendix H. The script started with the main function, which ran a function to start the robot, then another function to run the course. The startup function took in user input, figuring out if it

should run the motor testing script, as well as which course should be run. The motor testing script ran each motor, letting the user see if any of the motors were not working properly. The function to run the course took in the course specified earlier in the startup function and ran that course. This was done through turn functions, movement functions, and functions specifically designed for completing a task.

The first function that was run was the readyStart function. This function either waited for the starting light, or, if a starting light was not detected, started in thirty seconds. After this, the fuel servo was set to the corresponding angle with another function based on the direction that needed to be turned. Two types of movement and turn functions were used. One of these used sleep statements, and the other function used RPS. The sleep statements were used for longer movements, while the RPS was used for short, quick corrections so that the robot knew where it was.

The control panel had two functions specifically for completing that task. The first function read the color of the light, assuming that the robot is above the light, and the second function used this information to move to the correct spot and hit the corresponding button. A separate function was used for the fuel crank as well, which turned the crank in a direction based on the RPS information it collected at the lower section of the course.

5.3 Budget

The team was provided with a budget of $160 to construct the robot. Plans were drawn out for initial construction, and the majority of the budget would be going toward motors and the drivetrain. Due to the decision to go with the most expensive motor option, the IGWAN, it was

decided that the chassis should be constructed out of a cheaper material, MDF wood, to maintain a surplus in the budget. The first order placed cost the most money, coming in at a total of $75.96. This order cost the most because the original construction of the robot came first, and all orders placed after were for accessories or additional mechanisms. A diagram of money spent over time can be seen in Appendix I as Figure I1. The breakdown of what was spent can be divided into categories of motor, drivetrain, fastener, chassis, erector pieces, electrical, services, and miscellaneous parts. How the budget was proportioned to these different categories of objects can be seen in Figure 15 below [4].

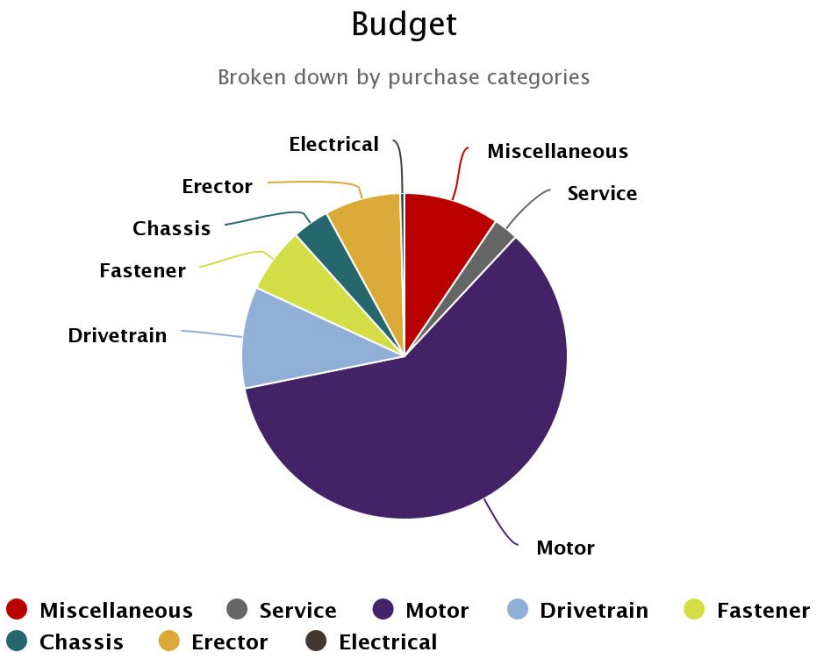


Figure 15: Breakdown of purchases by category [4].

The initial team plan was to end up with at least $40 left over, so there would be money available in case anything broke or needed to be repaired. After the team purchased all materials for the full robot, there was $44.96 remaining in the budget. Then, it was necessary to purchase

two extra wheel adapters, as the spokes broke off due to wear down over time. The team finished with $42.96 remaining in the budget. At times, the team underestimated how many of some objects were going to be required, but the Speed D8ers met their goal of maintaining the budget and keeping a reserve of at least $40. A detailed log of purchases can be found in Table I1 in Appendix I.

5.4 Schedule

To ensure a timely completion of the robot prototype, the Speed D8ers produced a design schedule to plan when each task would be completed. While working on the prototype, the team kept a time log to record the time each team member spent on each part of the project.

5.4.1 Design Schedule

After forming the company, team members produced a list of primary and secondary tasks to complete, projected start and finish dates for each task, predicted the time that each task would take, and assigned primary and secondary contributors for each task. As the project was completed, team members updated the design schedule with actual start and finish dates, and the actual time each task required. The full design schedule is shown in Table J1 in Appendix J.

For the most part, the Speed D8ers stuck to the original schedule. Based on the level of difficulty of each task, some were finished early while others weren't finished until the day they were due. For example, the team completed Performance Tests 1 and 4 on the due dates, a day after the projected finish date, yet they finished Performance Tests 2 and 3 each a week in

advance. The team did not project the number of hours to complete each task with great accuracy, especially on larger tasks related to testing and documentation.

5.4.2 Time Log

Throughout the duration of the project, each team member maintained a time log to track how much time was spent on each portion of the project. Time was spent on project management, documentation, coding, building, CAD modeling, testing, and other activities. Team members filled out a log each week documenting which times they worked on tasks in each category. Time logs for each week are shown in Table J2 in Appendix J. An overall time log is shown in Table 5 below.

Table 5: Overall timesheet.

Speed D8ers	Total Hours	Documentation	Management	Coding	Testing	CAD	Building	Other
Justin Banke	30	5	4.5	0	3	2	11.5	4
Jack McGinness	33.5	9	2	5.5	5	4.5	3.5	4
Kelly Meaden	34	11	7	0	6	2	4	4
Colby Williams	34	4	2	12	6	3	3	4
Team Totals	131.5	29	15.5	17.5	20	11.5	22	16

In addition to the weekly time logs, the Speed D8ers also kept a testing log for time spent using the courses provided by OSURED. The testing log provides the dates and times spent testing on each course, as well as who was testing the robot and what they were testing.

Furthermore, the testing log provides a brief description of the result of each test on the course. The full testing log is shown in Table J3 in Appendix J.

6. Final Competition

On April, 7, 2018, the Speed D8ers' prototype competed head-to-head against 62 other prototypes at a final competition in the Recreation and Physical Activity Center (RPAC). The competition consisted of three rounds of round robin play with four robots performing at a time. Then, there was a single-elimination tournament, where the robot that earned the most points moved on, and the other three robots were eliminated. In the event of a tie, the robot that completed the tasks in the least amount of time was declared the winner of that round. All scores were determined by the Course Automated Software (CAS) system [2].

6.1 Strategy

In the final competition, the Speed D8ers used the same strategy as in the individual competition, depicted in Figure 9 on page 34. The robot began with the button control panel, pressing both the electrical test button and the RPS button while there. Then, the robot aligned with the wrench using RPS heading and coordinate checks, and moved forward to pick it up. Then, it moved back, turned left toward the wall, and turned left again before driving into the car jack. After toggling the jack, the robot backed up and turned toward the grass ramp. It made a slight turn to line up with the center of the ramp, and drove to the upper level. Then, the robot turned 45 degrees to the fuel crank. After spinning the crank, it backed up and turned 90 degrees

to deposit the wrench in the garage. Then, the robot took the concrete path back to the lower level and finished the course by pressing the final charging button.

6.2 Round Robin 1

The Speed D8ers' robot completed all primary and secondary tasks in 58 seconds and 32 milliseconds. No changes needed to be made following the first round.

6.3 Round Robin 2

The Speed D8ers' robot completed all primary and secondary tasks in 48 seconds and 35 milliseconds. No changes needed to be made following the second round.

6.4 Round Robin 3

The Speed D8ers' robot completed all primary and secondary tasks in 36 seconds and 72 milliseconds. No changes needed to be made following the third round.

6.5 Single-Elimination

The Speed D8ers' robot completed all primary and secondary tasks in 18 seconds and 31 milliseconds in every round, thus winning the competition.

6.6 Performance Analysis

Everything was perfect.

7. Summary and Conclusions

During this project, the Speed D8ers designed, built, and programmed a robot to complete tasks in the Formula EH Grand Prix pit area. As described in section 1, the project was announced to team members on January 26. The company formed on January 31, and came up with three preliminary concepts and a mock-up, as detailed in section 2. Throughout the project, the team made critical decisions while constructing and coding the robot using analysis and testing, as described in section 3. The robot competed in an individual competition on March 30, which is summarized in section 4. The robot's final design is detailed in section 5. Finally, the robot competed in a final competition on April 7, and its performance is analyzed in section 6.

7.1 Summary

After much deliberation and brainstorming, the Speed D8ers constructed their chassis and drivetrain using MDF ¼" wood, two wheels powered by IGWAN motors and two plastic-spoon-skids. The team added a mechanism to pick up the wrench using erector set pieces and popsicle sticks, powered by a Futaba servo motor. A mechanism to spin the fuel crank was made using a 3D printed disk and a ping-pong paddle cover, powered by a Futaba servo motor.

Team programmers initially relied on time-based navigation for the first four performance tests, but added in RPS checks for optimal performance during the individual and final competitions. During the individual competition, the robot consistently activated with the starting light, toggled the car jack, and deposited the wrench. The robot inconsistently pressed

the correct diagnostic button, picked up the wrench, and pressed the final button. The robot failed to spin the fuel crank 180 degrees in the correct direction. However, with additional modifications to the code, the robot successfully completed all primary and secondary tasks during the final competition.

7.2 Conclusions

Overall, the Speed D8ers' robot performed satisfactorily. The height of the chassis was optimal for pressing the buttons on the control panel and toggling the jack without having to add and additional mechanism to complete those tasks. The IGWAN motors were long-lasting and high-performing. However, the mechanisms for the wrench and fuel crank were inconsistent. Although the popsicle sticks were able to pick up the wrench most of the time, the robot occasionally failed to align perfectly with the wrench. Likewise, the spinning disk had to be perfectly aligned with the crank in order to spin it 180 degrees in the correct direction, and it was often misaligned. Since the servo motor could not rotate completely, the robot had to approach the fuel crank twice to spin it the full amount.

In the future, companies should hack the servo motors so they are able to rotate 360 degrees. This would enhance the speed of the robot on the course and lead to more accuracy while spinning the fuel crank. Companies should also consider using a magnet to pick up the wrench instead of a pronged mechanism that only works when it is perfectly aligned. With these changes, the prototype would have a better chance of being selected as the pit assistant for the Formula EH Grand Prix.

8. References

[1] "With new pit-stop rules in place, NASCAR teams in development mode." 2018, January 24. www.nascar.com.

[2] Robot Course Scenario. 2018, January 26. www.carmen.osu.edu.

[3] Performance Tests Robot 2018. 2018, January 23. www.carmen.osu.edu.

[4] FEH Robot Store. 2018, March 31. www.feh.osu.edu.

APPENDIX A

Brainstorming

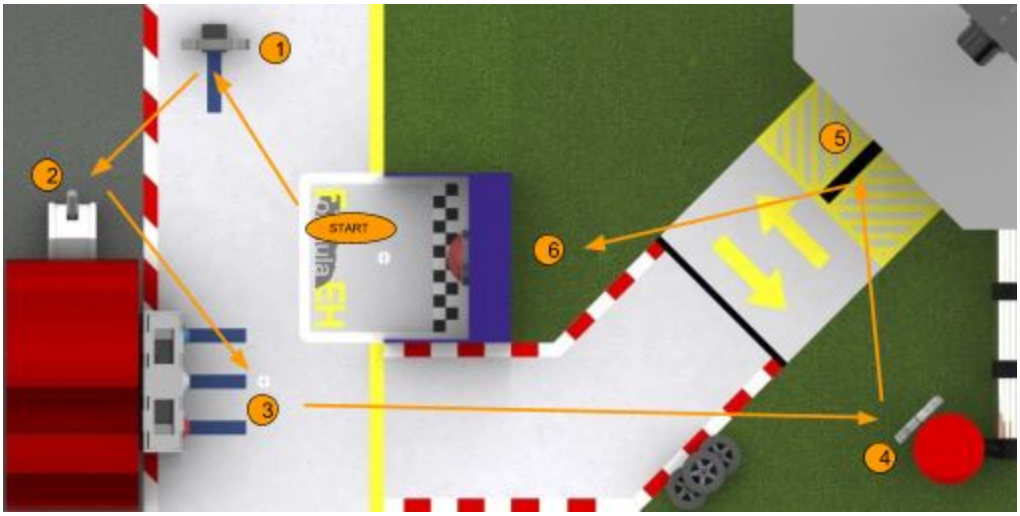


Figure A1: Second-ranked course strategy.

Table A1: Screening matrix for chassis and drivetrain brainstorming ideas.

Chassis and Drivetrain						
Success Criteria	**Ref**	**Design A**	**Design B**	**Design C**	**Design D**	**Design E**
Description		aluminum, 4 motored wheels	PVC, 2 motored wheels	Acrylic, 2 motored wheels + chain	Wood, Circular wheel layout (4)	2 robots
Speed	0	+	0	0	0	+
Power	0	+	0	0	0	-
Maneuverability	0	0	-	-	+	+
Cost	0	-	+	+	-	-
Maintenance	0	-	+	-	+	-
Durability	0	0	0	+	0	-
Sum +	0	2	2	2	2	2
Sum 0	6	2	3	2	3	0
Sum -	0	2	1	2	1	4
Net Score	0	0	1	0	1	-2
Continue?		Yes	Yes	Yes	No	No

Table A2: Screening matrix for car jack mechanism brainstorming ideas.

Car Jack					
Success Criteria	**Ref**	**Design A**	**Design B**	**Design C**	**Design D**
Description		Sloped Front	Lever	Arm With Clamp	Rod on Gear
Accuracy	0	-	+	+	0
Ease of Control	0	0	+	-	0
Speed	0	+	0	-	0
Consistency	0	0	-	+	0
Cost	0	+	-	-	-
Sum +	0	2	2	2	0
Sum 0	5	2	1	0	4
Sum -	0	1	2	3	1
Net Score	0	1	0	-1	-1
Continue?		Yes	Yes	No	No

Table A3: Screening matrix for fuel crank mechanism brainstorming ideas.

Fuel Crank					
Success Criteria	**Ref**	**Design A**	**Design B**	**Design C**	**Design D**
Description		Rod on Gear	Rotate Disk	Vertical spoke arm	Two rods on gear
Accuracy	0	0	-	+	+
Ease of Control	0	+	0	0	0
Speed	0	0	+	-	0
Consistency	0	0	0	-	+
Cost	0	0	0	0	0
Sum +	0	1	1	1	2
Sum 0	5	4	3	2	3
Sum -	0	0	1	2	0
Net Score	0	1	0	-1	2
Continue?		Yes	Yes	No	Yes

Table A4: Screening matrix for wrench mechanism brainstorming ideas.

Wrench					
Success Criteria	**Ref**	**Design A**	**Design B**	**Design C**	**Design D**
Description		Arm Go In Hole	Fork Lift	Rod with Stopper	Arm with Clamp
Accuracy	0	0	+	-	+
Ease of Control	0	+	0	+	0
Speed	0	0	-	+	-
Consistency	0	0	+	-	+
Cost	0	+	0	+	-
Sum +	0	2	2	3	2
Sum 0	5	3	2	0	1
Sum -	0	0	1	2	2
Net Score	0	2	1	1	0
Continue?		Yes	Yes	Yes	No

Table A5: Concept scoring matrix for three preliminary ideas.

		Reference		**Idea One**		**Idea Two**		**Idea Three**	
Success Criteria	Weight	Rate	Weight	Rate	Weight	Rate	Weight	Rate	Weight
Speed	20%	3	0.6	2	0.4	2	0.4	4	0.8
Power	5%	3	0.15	2	0.1	3	0.15	5	0.25
Maneuverability	25%	3	0.75	4	1	5	1.25	5	1.25
Cost	20%	3	0.6	5	1	4	0.8	2	0.4
Consistency	20%	3	0.6	4	0.8	3	0.6	3	0.6
Durability	10%	3	0.3	5	0.5	4	0.4	4	0.4
Total Score	100%		3		3.8		3.6		3.7
Continue		No		Yes		Yes		Yes	

APPENDIX B

Preliminary Concept Sketches

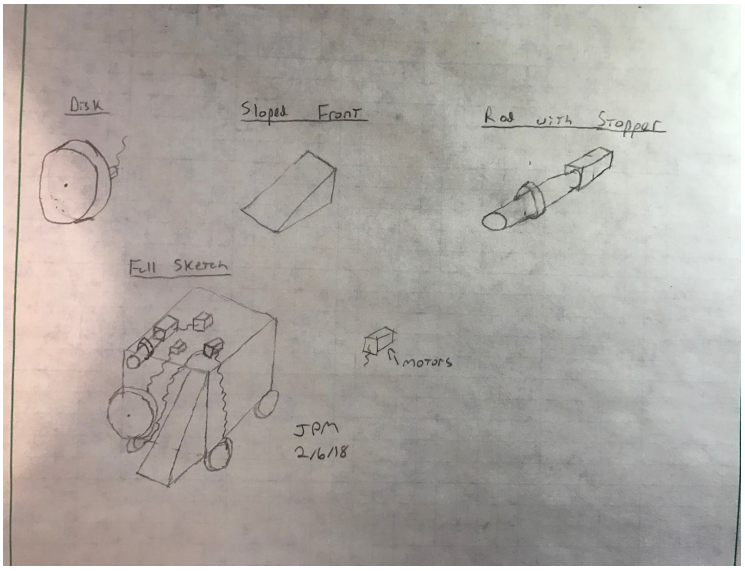


Figure B1: Two dimensional sketches depicting Preliminary Idea One.

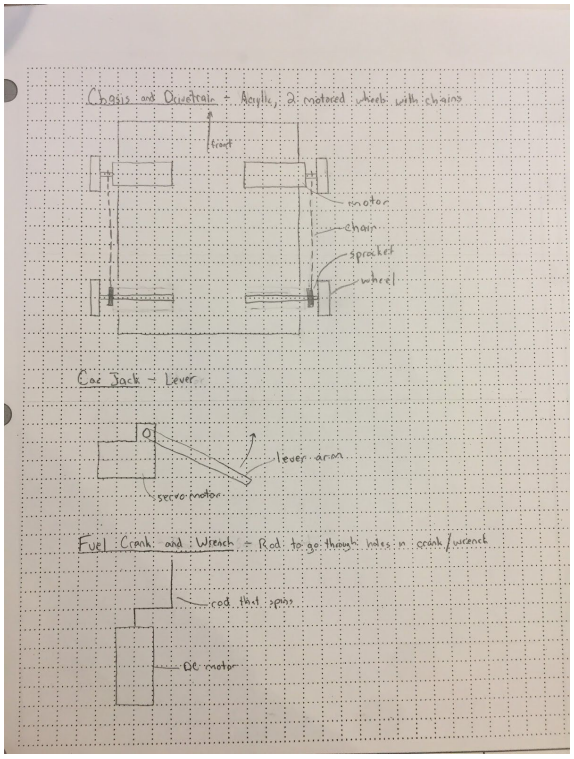


Figure B2: Two dimensional profile sketches of Preliminary Idea Two components.

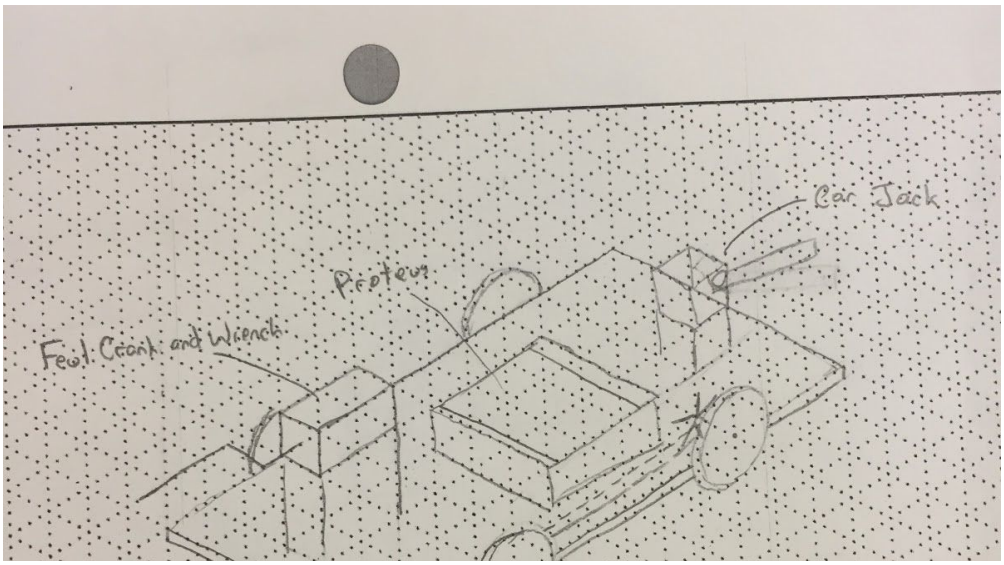


Figure B3: Isometric sketch of Preliminary Idea Two.

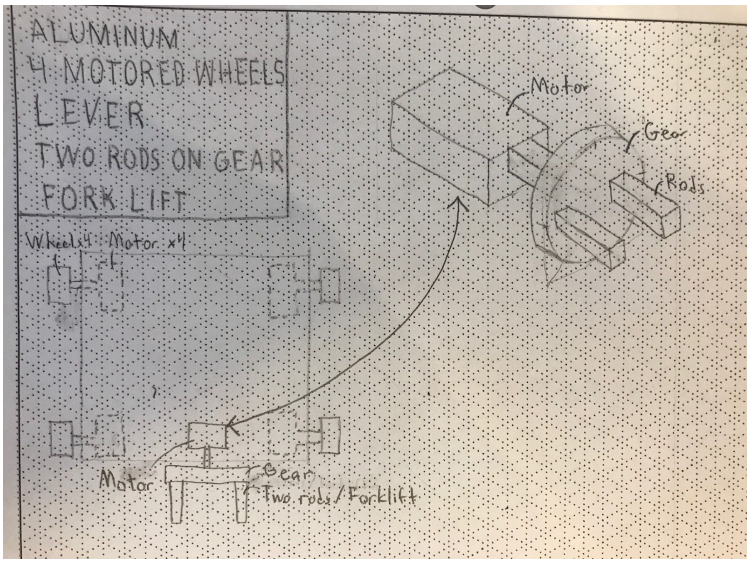


Figure B4: Sketch depicting Preliminary Idea Three.

APPENDIX C

Analysis and Explorations

Table C1: Measurements of segmented navigation through robot course.

Start Location	**End Location**	**Distance (inches)**
Start Zone	Buttons	16
Buttons	Car Jack	24
Car Jack	Wrench	16
Wrench	Garage	52
Garage	Fuel Crank	25
Fuel Crank	End Button	60
	Total	**193**

Table C2: Estimated times for robot course tasks.

Action	**Description**	**Estimated Duration (seconds)**
press correct button	read color, turn right or left, drive into correct colored button and white button for 5 seconds	8
lift car jack	drive into car jack	2
pick up wrench	drive so arms go through holes, lift servo	7
deposit wrench	drive into garage, lower servo, back up	3
turn fuel crank	turn servo correct direction, drive so arms go through holes, turn servo correct direction	5
press end button	drive into end button	1
	Total	**26**

Table C3: Estimated weights for robot components.

Component	Approximate Unit Weight (g)	Quantity	Approximate Total Weight (g)
aluminum chassis	200	1	200
Igwan motor	71	2	142
2” DuBro wheel	20	4	80
Futaba servo motor	37	2	74
Proteus	437	1	437
CdS cell and case	15	1	15
aluminum button plate	38	1	38
PVC mechanism	6	1	6
wooden mechanism support	52	1	52
wires, screws, buffer	15	1	15
		Total	**1059**

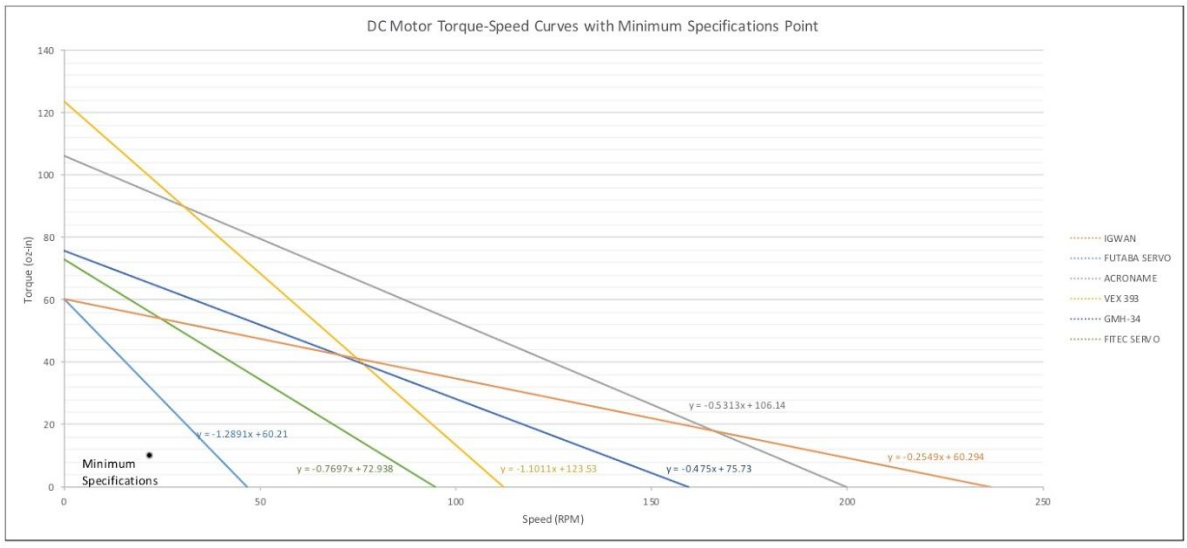


Figure C1: DC motor torque-speed curves with minimum specifications point.

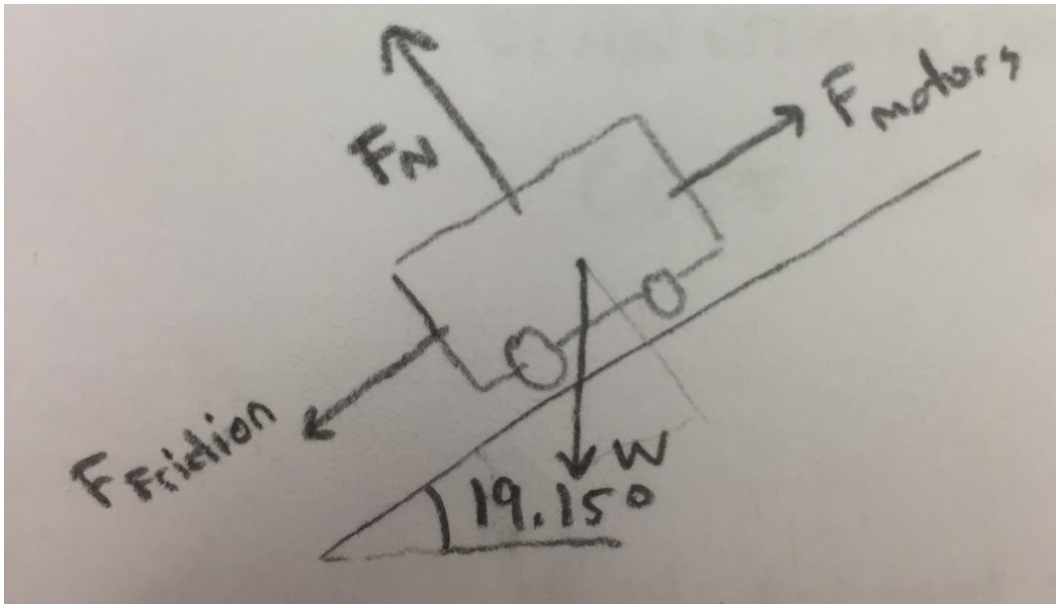


Figure C2: Free-body diagram of the forces acting on the robot traveling up the course ramp.

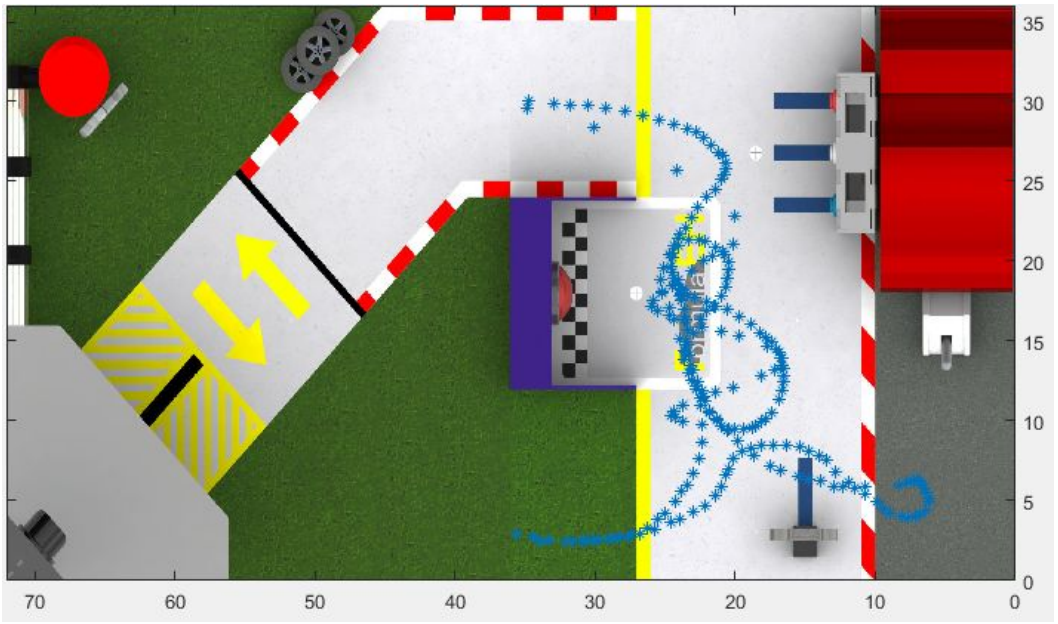


Figure C3: Data log using RPS coordinates on MATLAB.

APPENDIX D

Sample Calculations

Sample Calculation for Equation 1:

$$v = \frac{distance}{time} = \frac{193\ inches}{85\ seconds} = \textbf{\textit{2.3 in/sec}}$$

Sample Calculation for Equation 2:

$$\omega = \frac{v}{2\Pi r} = \frac{2.3\ in/sec}{2\Pi \cdot 1\ in} = \frac{0.366\ rev}{1\ sec} \cdot \frac{60\ sec}{1\ min} = \textbf{22.0 rev/min}$$

Sample Calculation for Equation 3:

$$ounces = grams \cdot \frac{1\ oz.}{28.35\ g} = 1059\ g \cdot \frac{1\ oz.}{28.35\ g} = \textbf{37.36 oz.}$$

Sample Calculation for Equation 4:

$$F_{motors} = F_{friction} + W sin(19.15) = 8\ oz + (37.36\ oz \cdot sin(19.15)) = \textbf{20.26 oz.}$$

Sample Calculation for Equation 5:

$$\tau_{motors} = F_{motors} \times r_{wheel} = 20.26\ oz \times 1.00\ in = \textbf{20.26 oz-in}$$

Sample Calculation for Equation 6:

$$\frac{encoder\ units}{1\ inch} = \frac{encoder\ units}{rotations} \cdot \frac{rotations}{\Pi d\ inches} = \frac{318\ encoder\ units}{1\ rotation} \cdot \frac{1\ rotation}{\Pi \cdot 2.5\ inches} = \textbf{40.489 units/inch}$$

APPENDIX E

Robot Design

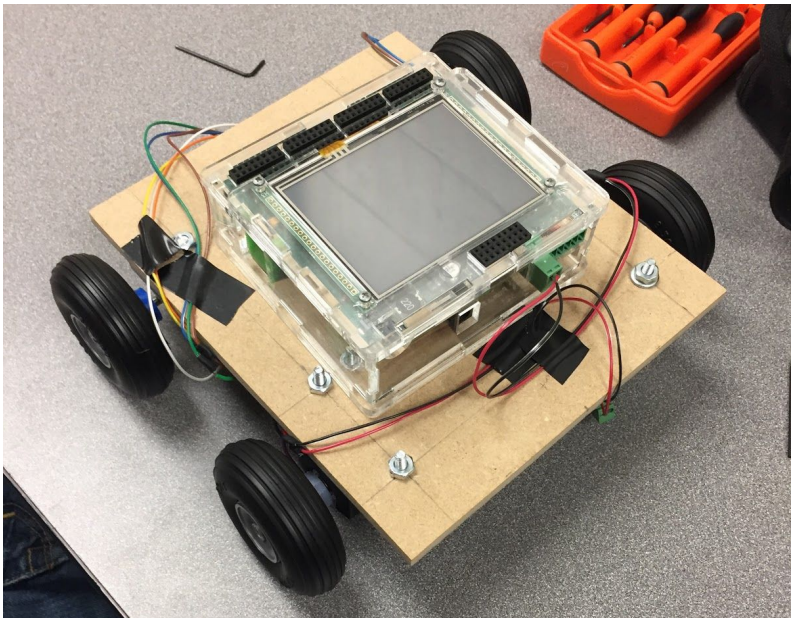


Figure E1: First build of the chassis and drivetrain.

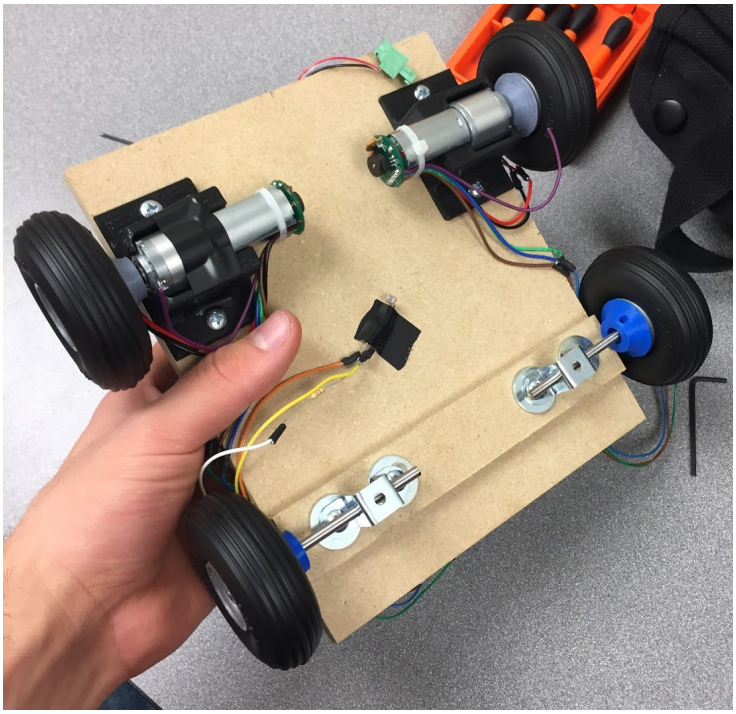


Figure E2: Bottom view of the initial build.

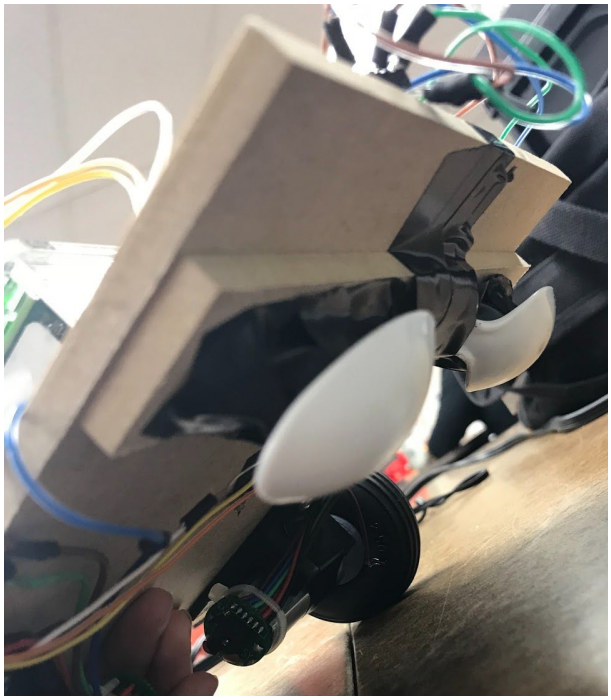


Figure E3: Plastic spoons added as skids, replacing the back wheels.

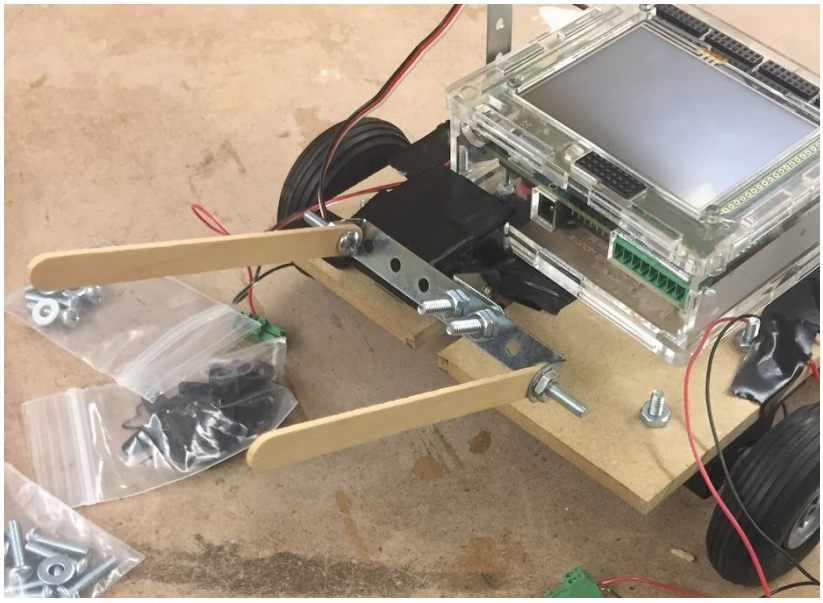


Figure E4: Initial construction of wrench mechanism.

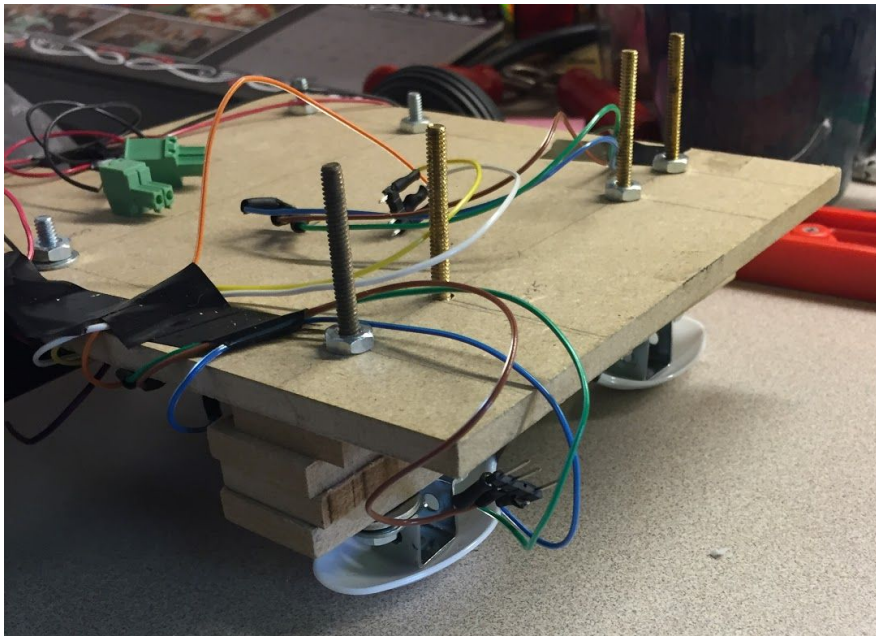


Figure E5: Four pieces of wood underneath chassis to level the robot.

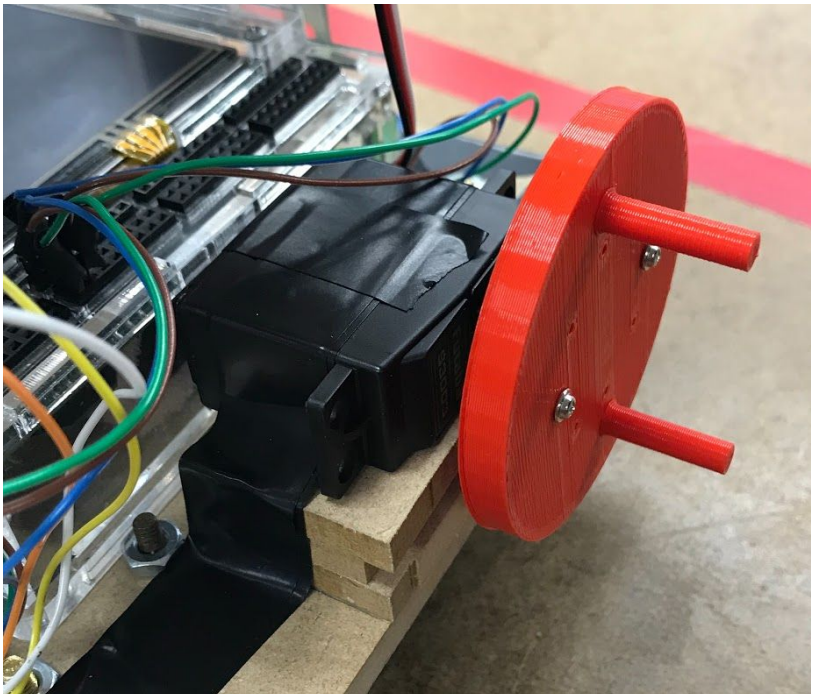


Figure E6: 3D printed part for the fuel crank mechanism.

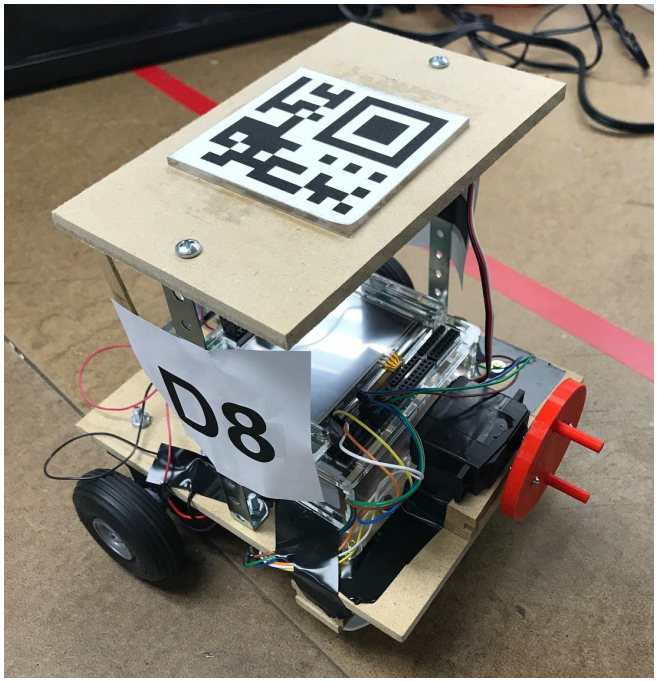


Figure E7: Robot with pronged 3D printed fuel crank mechanism.

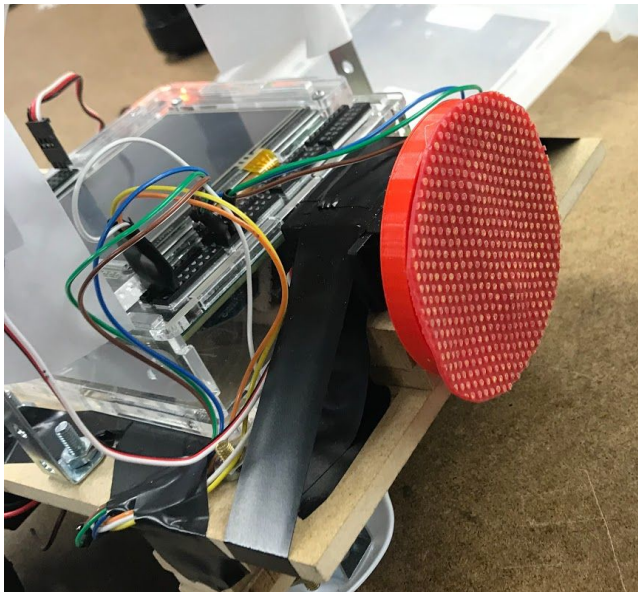


Figure E8: Revised fuel crank mechanism for Performance Test 4.

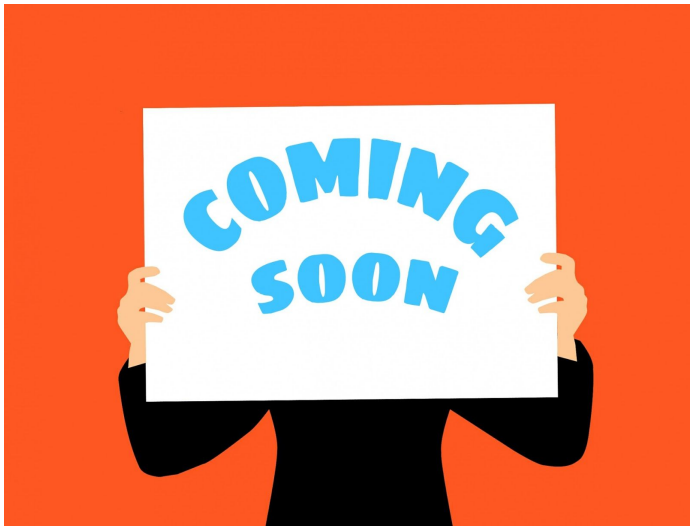


Figure E9: Robot for final competition.

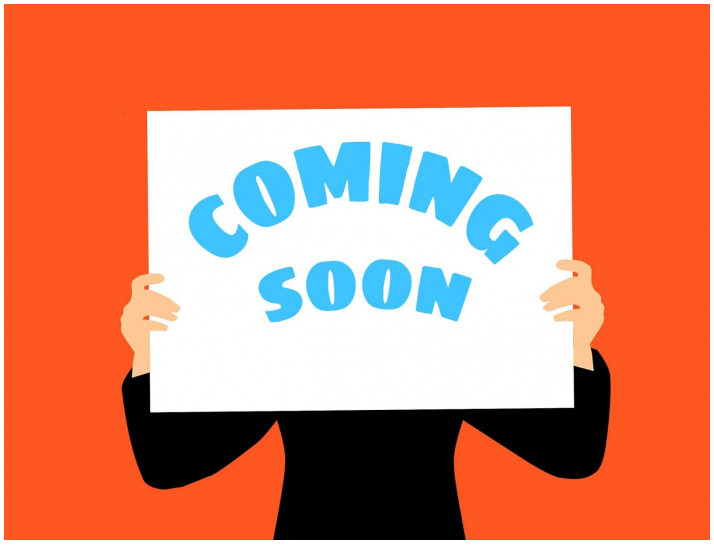


Figure E10: Robot's drivetrain.

APPENDIX F

Testing Strategies

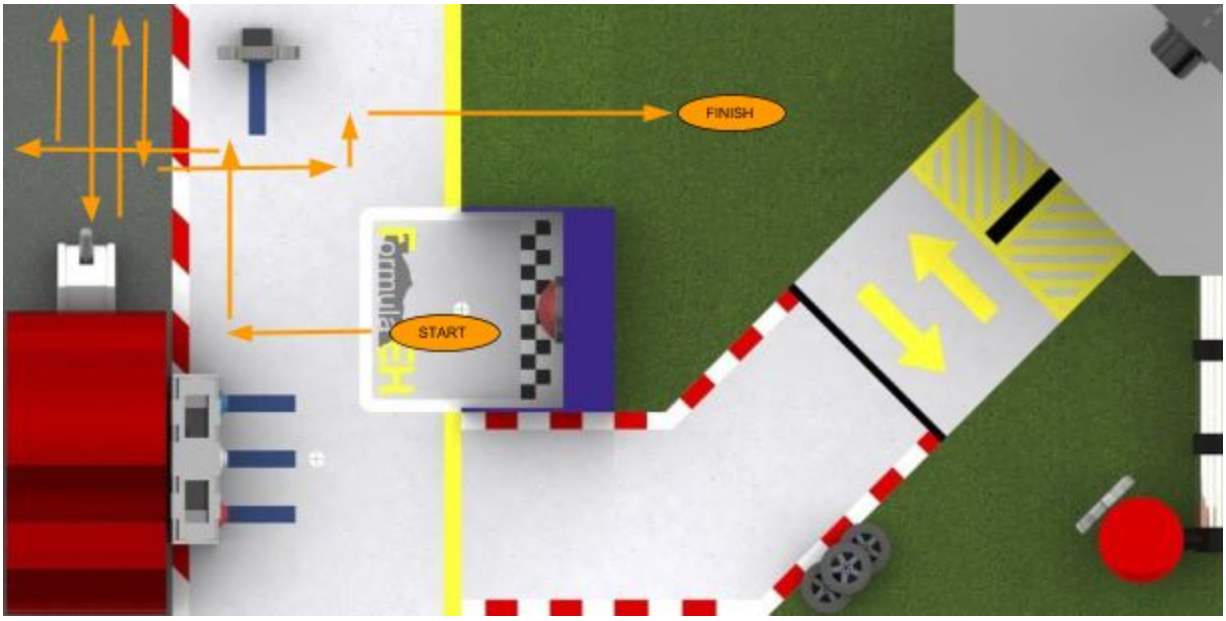


Figure F1: Initial course strategy for Performance Test 1.

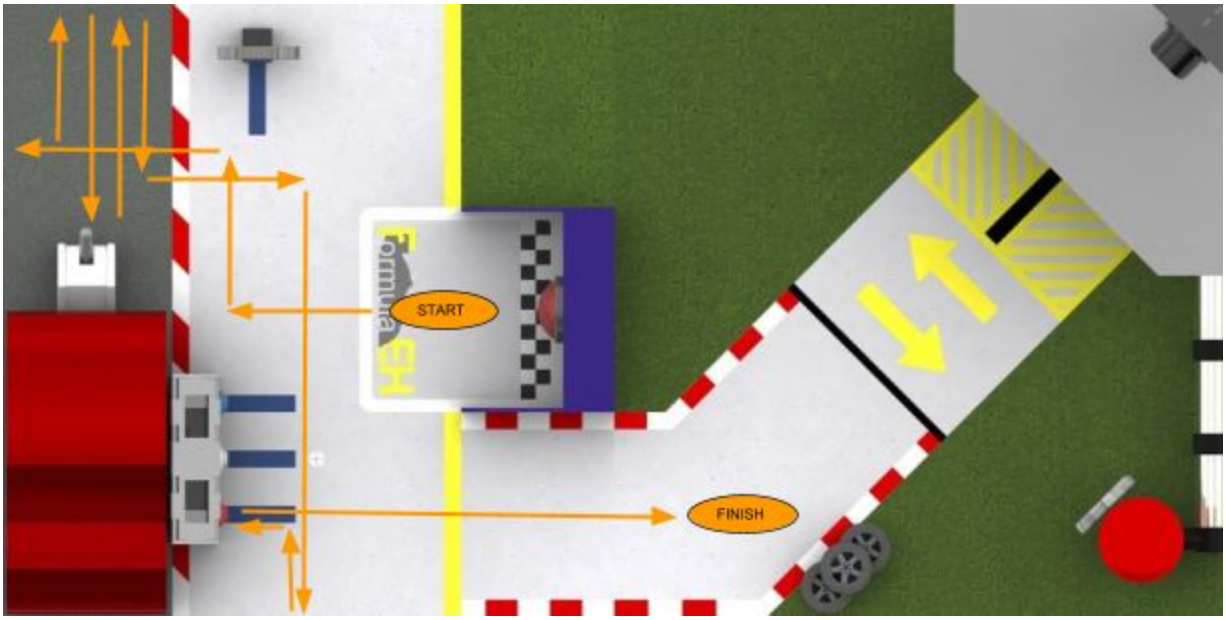


Figure F2: Second course strategy for Performance Test 1.

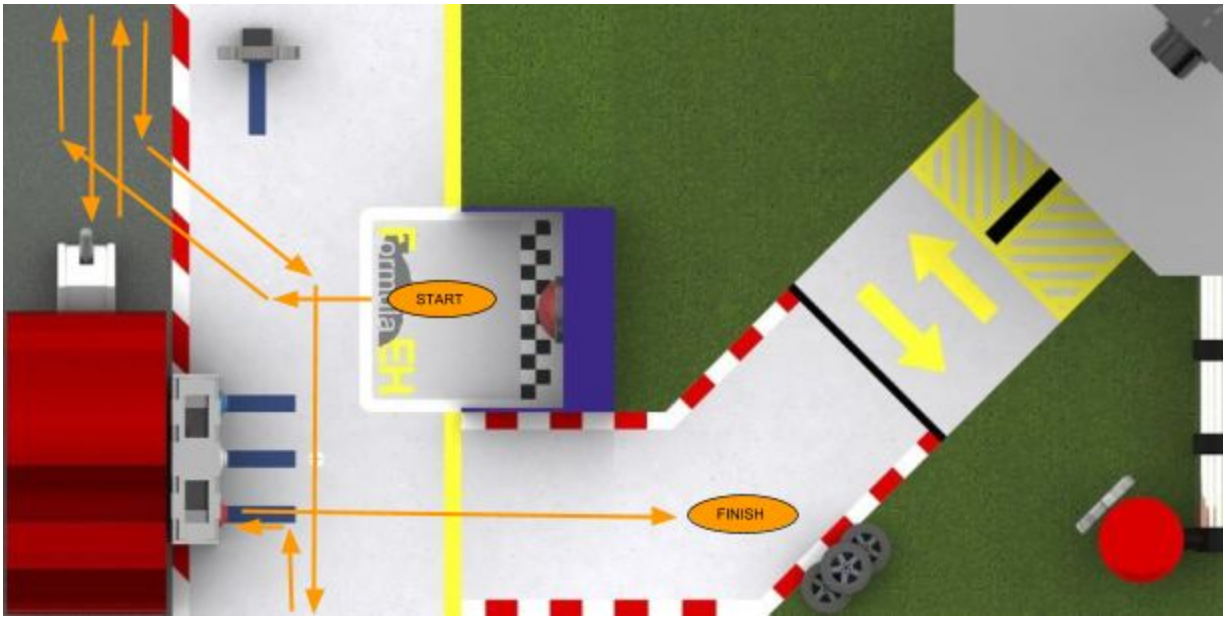


Figure F3: Final course strategy for Performance Test 1.

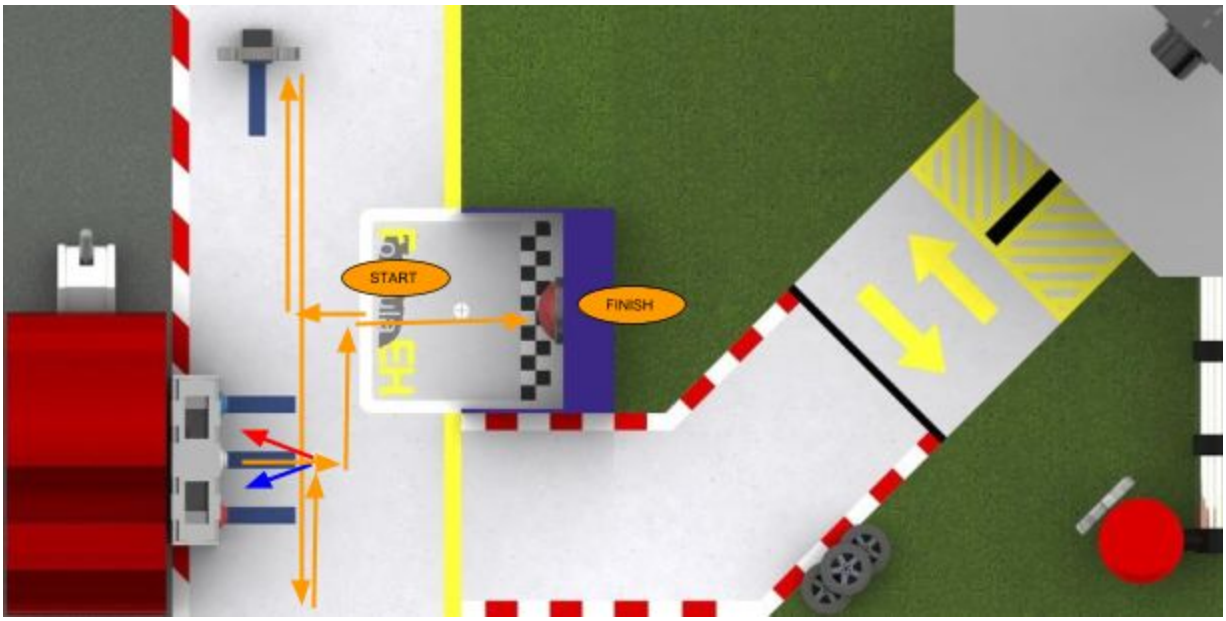


Figure F4: Course strategy for Performance Test 2.

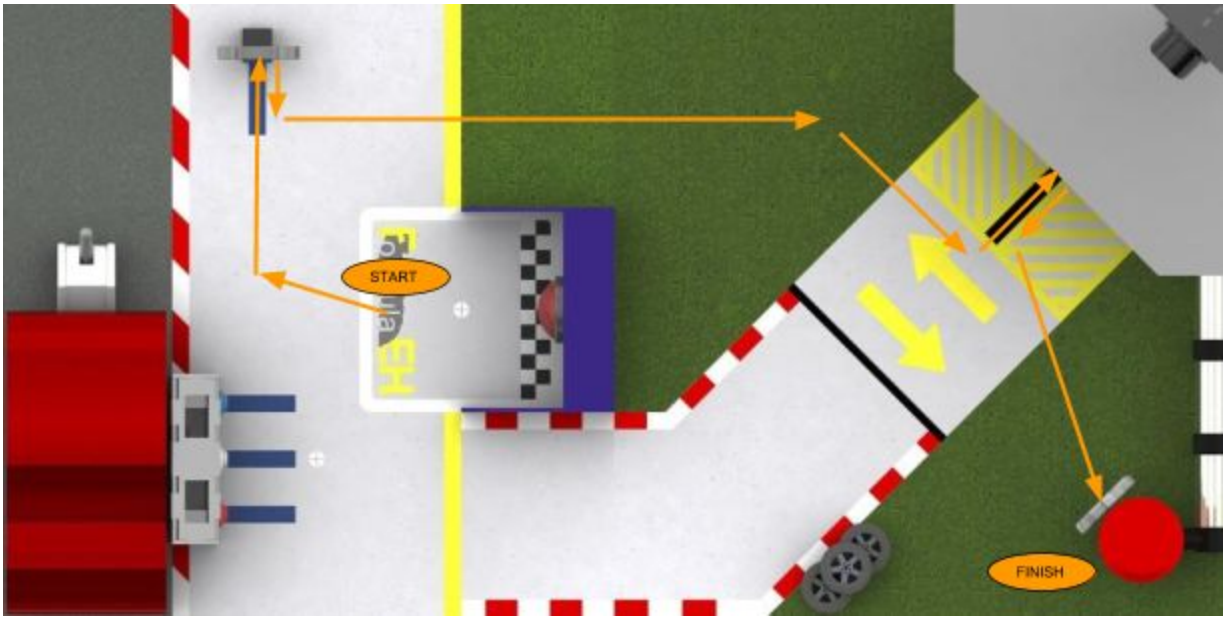


Figure F5: Course strategy for Performance Test 3.

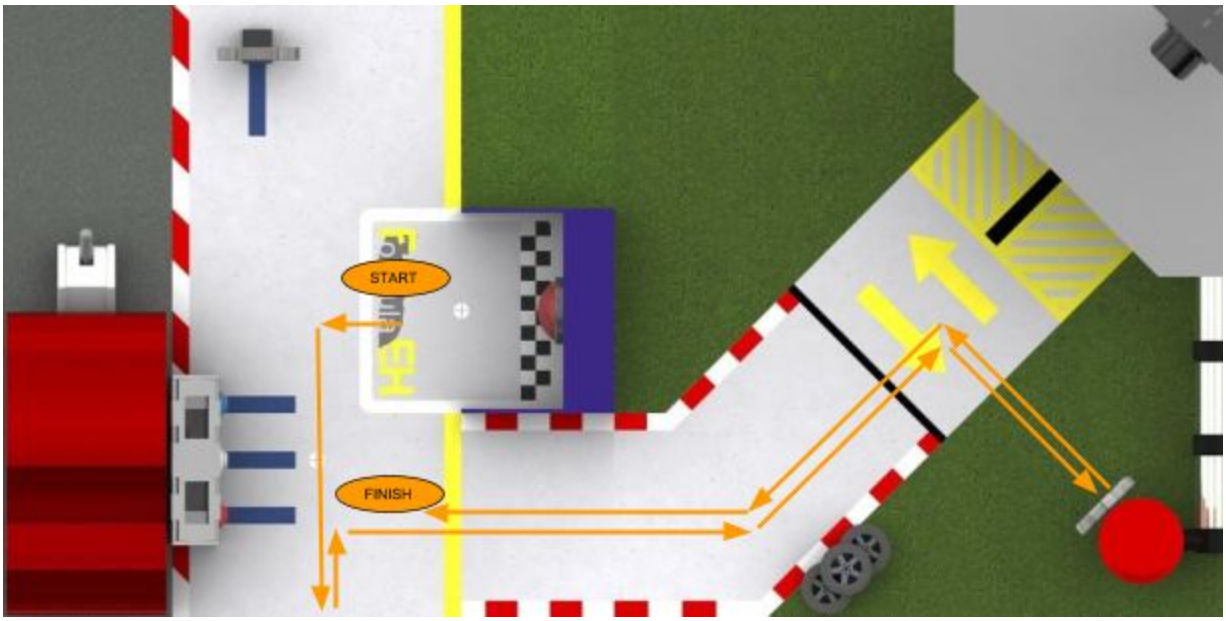


Figure F6: Initial course strategy for Performance Test 4.

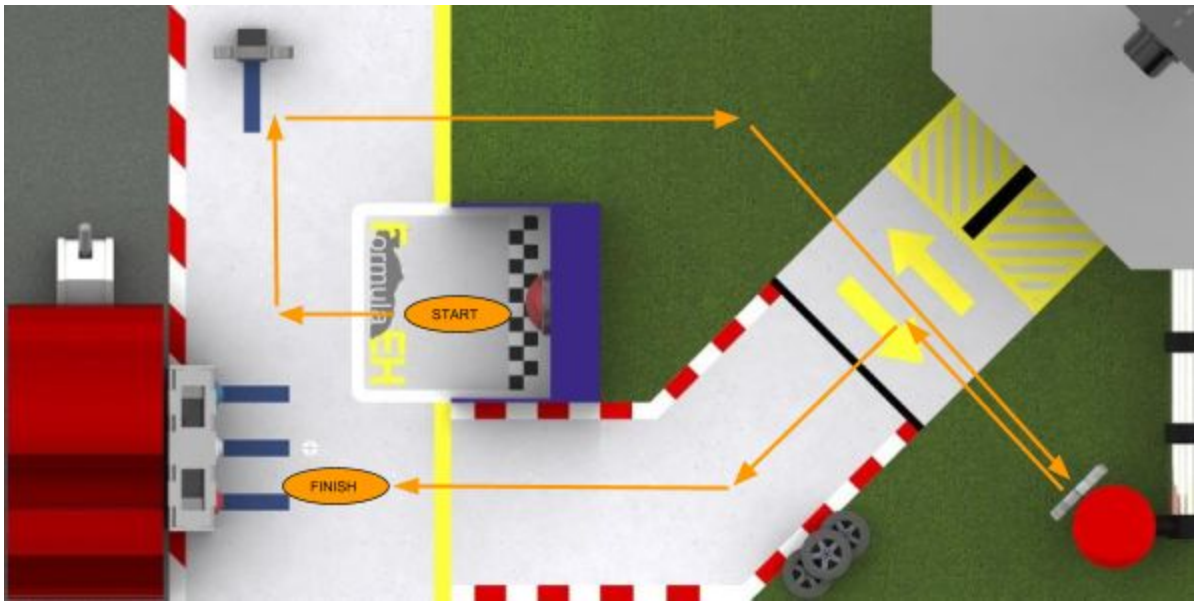


Figure F7: Second course strategy for Performance Test 4.

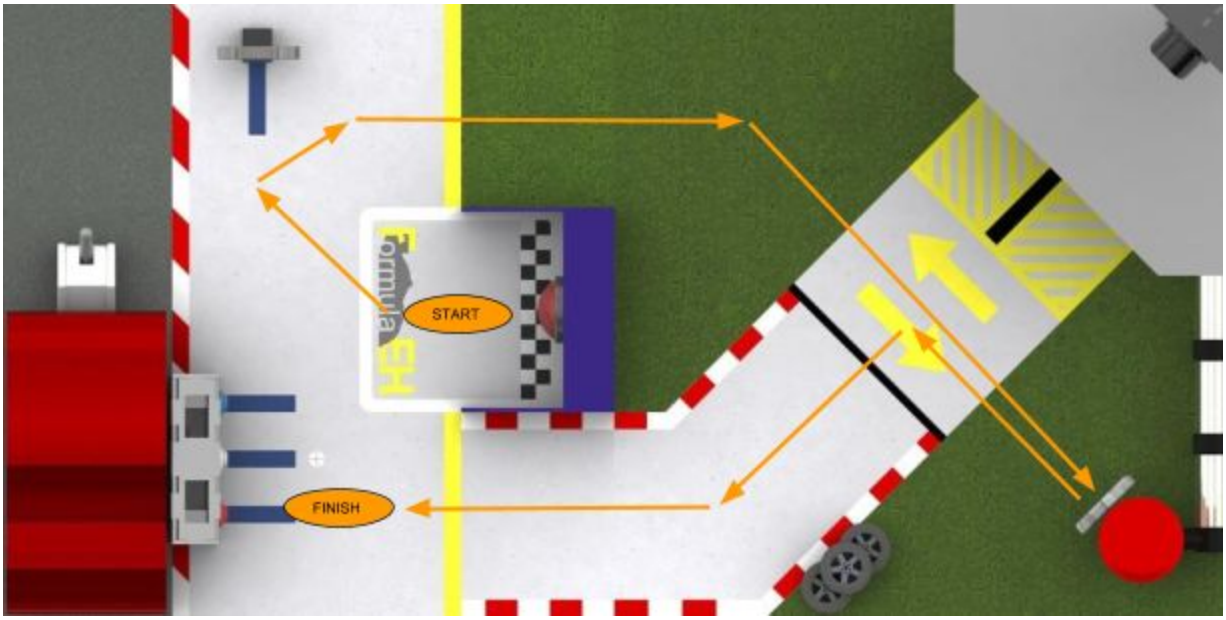


Figure F8: Third course strategy for Performance Test 4.

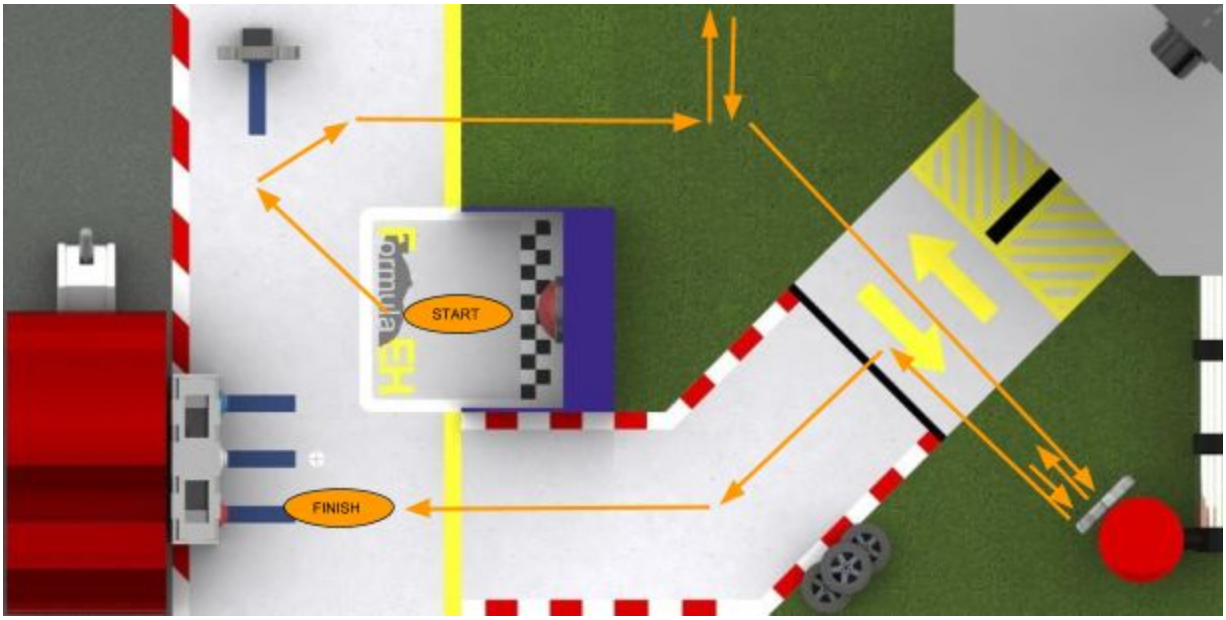


Figure F9: Fourth course strategy for Performance Test 4.

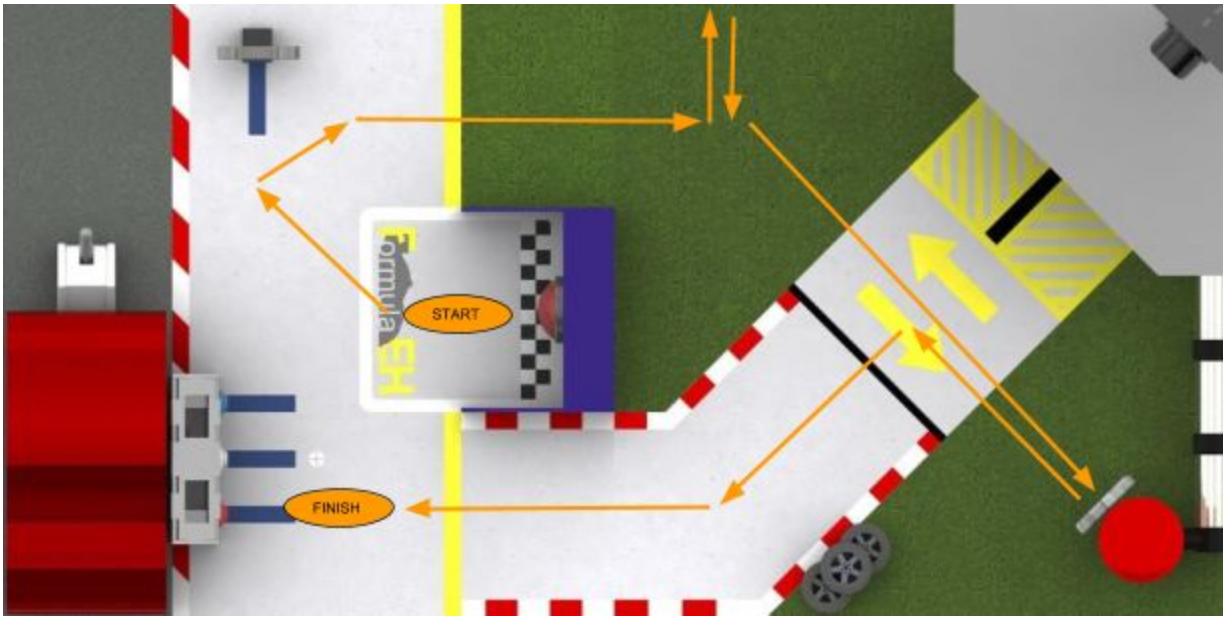


Figure F10: Final course strategy for Performance Test 4.

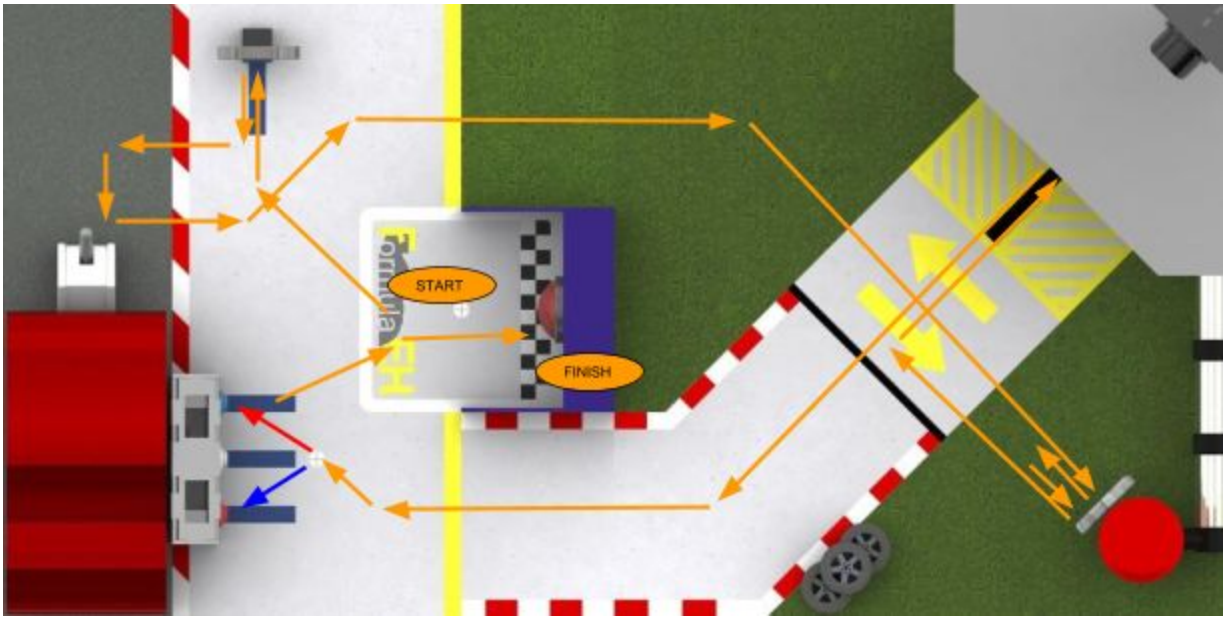


Figure F11: Initial course strategy for the individual competition.

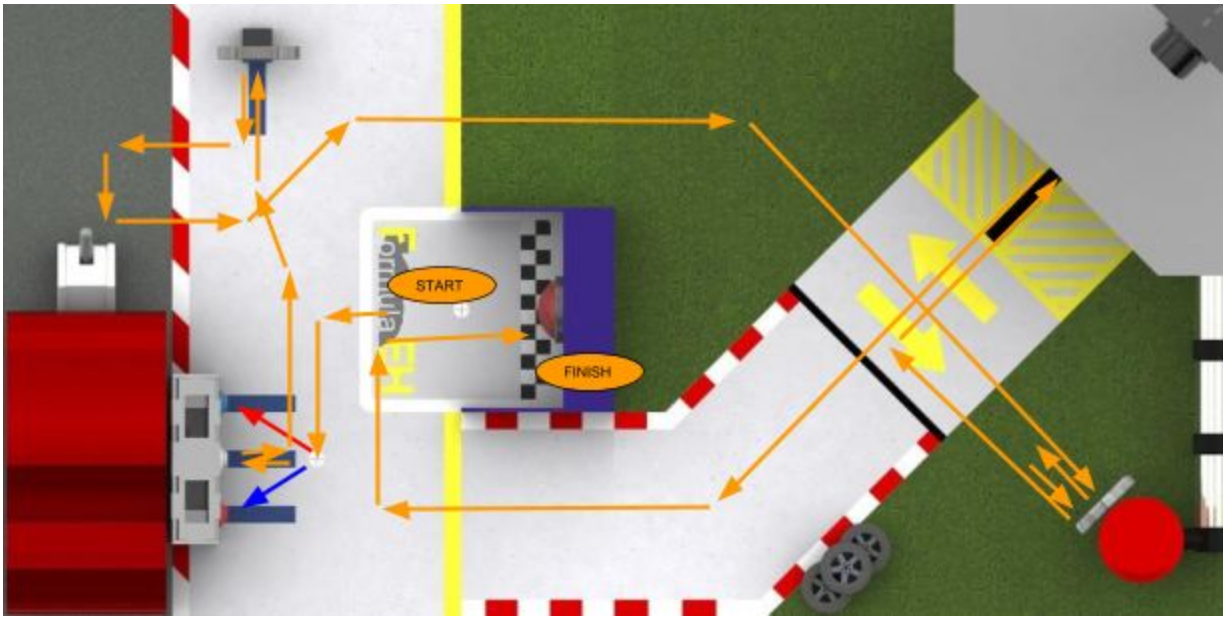


Figure F12: Final course strategy for the individual competition.

APPENDIX G

Electrical Systems

Table G1: Electrical connections table for GPIO ports

Port ID	Wire ID	In-Code Name (Object Name)	Type of Sensor	Purpose	Additional Notes
P0_0					
P0_1	P0_1	lightSensor	CdS Cell	Detects the start light. Identifies button board light	White, yellow, orange wire
P0_2					
P0_3					
P0_4					
P0_5					
P0_6					
P0_7					
P1_0	P1_0	leftEncoder	Igwan encoder	Counts revolutions of left motor shaft	Blue, green, brown wire
P1_1	P1_1	rightEncoder	Igwan encoder	Counts revolutions of right motor shaft	Blue, green, brown wire
P1_2					
P1_3					
P1_4					
P1_5					
P1_6					
P1_7					
P2_0					
P2_1					
P2_2					
P2_3					
P2_4					
P2_5					
P2_6					
P2_7					
P3_0					
P3_1					
P3_2					
P3_3					
P3_4					
P3_5					
P3_6					

P3_7					

Table G2: Electrical connections table for Servo ports

Port ID	Wire ID	Variable Name	Type of Servo	Purpose	Additional Notes
Servo 0	S0	fuelServo	Futaba	Turn fuel crank	red, white, black wire
Servo 1					
Servo 2					
Servo 3					
Servo 4					
Servo 5					
Servo 6					
Servo 7	S1	wrenchServo	Futaba	Pick up wrench	red, white, black wire

Table G3: Electrical connections table for Motor ports

Port ID	Wire ID	Variable Name	Type of Motor	Purpose	Additional Notes
Motor0	M0	rightMotor	Igwan	Drive for right wheel	M0
Motor1	M1	leftMotor	Igwan	Drive for left wheel	M1
Motor2					

Motor3					

APPENDIX H

Code

```
//Includes
#include <FEHLCD.h>
#include <FEHIO.h>
#include <FEHUtility.h>
#include <FEHMotor.h>
#include <FEHServo.h>
#include <FEHRPS.h>

//A ton of defines

//Power
#define POWER 9.0

//Movement Percentages
#define HIGH_FORWARD 50
#define MID_FORWARD 30
#define LOW_FORWARD 20
#define FAST 100

//Turning Percentages (-20/20)
#define INSIDE_TURN_POWER -12
#define OUTSIDE_TURN_POWER 12
#define FAST_INSIDE -50
#define FAST_OUTSIDE 50


//Internal adjustments
#define MOVE_ADJUST2 3
#define TURN_ADJUST 145 //Increase decreases turn angle     317    55.6
#define MOVE_ADJUST 4.6 //Increase increases distance

//Times
#define HALF_SECOND 500
#define FULL_SECOND 1000

//Wrench Servo: 40 degrees is straight up
#define WRENCH_SERVO_MIN 508
#define WRENCH_SERVO_MAX 2061
#define FUEL_SERVO_MIN 500
#define FUEL_SERVO_MAX 2268

//Starting position
#define START_X 16.7
```

```
#define START_Y 28.6
#define START_HEADING 270.0

#define TIMEOUT 10



//Figure out connections
FEHMotor rightMotor(FEHMotor::Motor0,POWER);
FEHMotor leftMotor(FEHMotor::Motor1,POWER);
FEHServo wrenchServo(FEHServo::Servo7);
FEHServo fuelServo(FEHServo::Servo0);
DigitalEncoder rightEncoder(FEHIO::P1_1);
DigitalEncoder leftEncoder(FEHIO::P1_0);
AnalogInputPin lightSensor(FEHIO::P0_1);


int startUp();
void checkMotors();
void showInfo();
void turnRight(float angle);
void moveForward(int percent, float distance);
void readyStart();
float abs(float value);
void runCourse(int test);
bool isBlue();
void pressButton(bool isBlue);
bool ask(char question[]);
void setFuel();
void turnFuel();
void mainToWrench();
void wrenchToJack();
bool isClockwise;
void turnTo(float angle);
void moveToX(float x);
void moveToY(float y);



float xCalibrate;
float yCalibrate;
float hCalibrate;
```

```
bool testing = true;


//270 points towards -x
//360 points towards +y
//x is jack direction

int main(void)
{


    int course = startUp();

    if(course != -1) {
        LCD.WriteLine("Ok, I'll run that course!");
        Sleep(FULL_SECOND);
        LCD.WriteLine("Wish me luck!");
        Sleep(FULL_SECOND);

        runCourse(course);

        if(ask("Did I do it right???")) {
            LCD.WriteLine("Yay I did it!");
            Sleep(HALF_SECOND);
            LCD.WriteLine("Was that an official test?");
            Sleep(HALF_SECOND);
            if(ask("Was it?!")) {
                LCD.WriteLine("Yes!  Full points!");
            } else {
                LCD.WriteLine("Lets make it official then");
                Sleep(HALF_SECOND);
                LCD.WriteLine("I got this!");
                Sleep(HALF_SECOND);
                LCD.WriteLine(" ");
                LCD.WriteLine(":)");
            }
        } else {
            LCD.WriteRC("Aw",0,0);
            Sleep(HALF_SECOND);
            LCD.WriteRC(".", 0, 2);
            Sleep(HALF_SECOND);
            LCD.WriteRC(".", 0, 3);
```

```
            Sleep(HALF_SECOND);
            LCD.WriteRC(".", 0, 4);
            Sleep(HALF_SECOND);
            LCD.WriteRC("I'll do better next time,",1,0);
            LCD.WriteRC("promise!",2,0);
        }

    } else {
        LCD.WriteLine("Guess I'm done then!");
        LCD.WriteLine("Good night!");
    }


    return 0;
}

//Runs the specified course
void runCourse(int test) {



    readyStart();

    switch (test) {
    case 0:     //MAIN TEST

        setFuel();

        moveForward(HIGH_FORWARD, 10);
        moveToY(22.5);

        turnRight(-75);
        turnTo(358);

        moveForward(HIGH_FORWARD, 17);
        moveToX(25.5);


        pressButton(isBlue());

        moveForward(HIGH_FORWARD, 13);
        //moveToX(16);
```

```
turnRight(-35);
//turnTo(240);



//THE WRENCH


wrenchServo.SetDegree(140);         // Set the servos

moveForward(50, 10);
moveToY(18.5);
showInfo();
Sleep(FULL_SECOND);                 // Line up with wrench

turnRight(30);
turnTo(180);
showInfo();
Sleep(FULL_SECOND);                 // Turn towards wrench

moveForward(LOW_FORWARD, 17);
//moveToX(7.8);
showInfo();
                // Approach wrench

wrenchServo.SetDegree(40);
Sleep(HALF_SECOND);

moveForward(-50, 6);
//moveToX(10);
showInfo();
                // Back off from wrench

turnRight(-66);
//turnTo(255);
showInfo();
               // Turn to approach crank

moveForward(50, 12);
moveToY(11);
showInfo();
                // Approach crank area
```

```
turnRight(-92);
//turnTo(355);
showInfo();
              // Turn to crank

moveForward(50, 10);
//moveToX(8.5);
showInfo();
              // Push the crank

moveForward(-50, 5);
//moveToX(6);
showInfo();
             // Back off

turnRight(85);
//turnTo(272);
showInfo();
            // Turn to ramp

moveForward(-50, 15);
moveToY(21);
showInfo();
              // Move to ramp

turnRight(-42);
turnTo(315);
showInfo();
             // Adjust on ramp

moveForward(-50, 10);
moveToX(4);
showInfo();
              // Move to ramp

turnRight(35);
turnTo(274);
showInfo();
             // Adjust on ramp

bool deadzone;
deadzone = (RPS.IsDeadzoneActive() != 2);
```

```
// UP TOP

moveForward(-1 * HIGH_FORWARD, 52);
if(!deadzone) {
   moveToY(48);
}

turnRight(43);
if(!deadzone) {
   turnTo(226);
}

moveForward(-1 * HIGH_FORWARD, 20);
if(!deadzone) {
   turnTo(226);
}
moveForward(-1 * HIGH_FORWARD, 20);
if(!deadzone) {
   turnTo(226);
}
moveForward(-1 * HIGH_FORWARD, 10);
turnFuel();                    // FUEL STUFF

moveForward(HIGH_FORWARD, 17);
if(!deadzone) {
   moveToX(17);
}

turnRight(80);
moveForward(HIGH_FORWARD, 25);
wrenchServo.SetDegree(150);        // Place wrench in garage
Sleep(FULL_SECOND);
moveForward(-1 * FAST, 45);        // Go to the ramp again
wrenchServo.SetDegree(40);
turnRight(47);                  // Turn to ramp
moveForward(-1 * FAST, 17);        // Down we go
turnRight(105);
moveForward(-1 * HIGH_FORWARD, 20);
turnRight(105);
moveForward(-1 * HIGH_FORWARD, 30);
```

```
    /*moveForward(-1 * FAST, 44);         // UP THE RAMP
    turnRight(90);                      // LINE UP WITH FUEL
    moveForward(MID_FORWARD, 10);
    moveForward(FAST, 10);
    moveForward(-1 * FAST, 10);
    turnRight(-39);

     // GET TO FUEL
    moveForward(-1 * HIGH_FORWARD, 50);
    turnFuel();                         // TURN THE FUEL
    moveForward(HIGH_FORWARD, 19);      // BACK TO CENTER
    turnRight(70);                      // LINE UP WITH GARAGE
    moveForward(HIGH_FORWARD, 40);      // INTO THE GARAGE

    wrenchServo.SetDegree(150);         // Place wrench in garage
    Sleep(FULL_SECOND);
    moveForward(-1 * FAST, 41);         // Go to the ramp again
    wrenchServo.SetDegree(40);
    turnRight(40);                      // Turn to ramp
    moveForward(-1 * FAST, 18);         // Down we go


    //25.9, 20.6
    moveToY(28);
    turnRight(45);
    turnTo(45);
    moveToY(20.6);
    turnRight(-45);
    turnTo(90);*/


    break;

case 1:     //POSITION BACKWARDS - CHECK 1
    moveForward(-1 * LOW_FORWARD, 8);   // Move out of start
    turnRight(32);
    moveForward(-1 * LOW_FORWARD, 18);  //Move to jack area
    turnRight(42);
    moveForward(-1 * MID_FORWARD, 8);
    moveForward(LOW_FORWARD, 8);        // LEVER
    moveForward(-1 * MID_FORWARD, 10);
```

```
    moveForward(LOW_FORWARD, 3);
    turnRight(-45);
    moveForward(LOW_FORWARD, 15);       // Back to center
    turnRight(45);
    moveForward(MID_FORWARD, 20);       // Over to the buttons
    moveForward(-1 * LOW_FORWARD, 4);
    turnRight(90);
    moveForward(LOW_FORWARD, 13);       // Nudge those buttons
    moveForward(-100, 50);              // And away we go
    break;

case 2:      // CHECK 2

    moveForward(LOW_FORWARD, 6);
    turnRight(85);
    moveForward(LOW_FORWARD, 15);       // Poke the Wrench
    moveForward(-1 * MID_FORWARD, 30);
    moveForward(LOW_FORWARD, 4);        // Ready for buttons
    turnRight(-90);
    moveForward(LOW_FORWARD, 1.5);
    pressButton(isBlue());              // Presses the button
    turnRight(90);
    moveForward(-1 * MID_FORWARD, 10);
    moveForward(LOW_FORWARD, 12);       // Back to the center
    turnRight(90);
    moveForward(LOW_FORWARD, 10);
    moveForward(HIGH_FORWARD, 10);      // Ram the finish button
    break;

case 3:     // CHECK 3
    turnRight(30);
    moveForward(LOW_FORWARD, 18.75);
    turnRight(65);
    wrenchServo.SetDegree(141);
    Sleep(HALF_SECOND);
    moveForward(LOW_FORWARD, 18);       // Approaches wrench
    wrenchServo.SetDegree(40);
    Sleep(HALF_SECOND);
    moveForward(-1 * LOW_FORWARD, 10);   // Backs away with wrench
    turnRight(-135);
    moveForward(-1 * LOW_FORWARD, 17);
    turnRight(35);
    moveForward(-1 * HIGH_FORWARD, 60); // Up the ramp
```

```
    turnRight(45);
    moveForward(-1 * LOW_FORWARD, 10);
    turnRight(95);
    moveForward(LOW_FORWARD, 20);
    wrenchServo.SetDegree(150);         // Place wrench in garage
    Sleep(FULL_SECOND);
    moveForward(-1 * LOW_FORWARD, 20);
    turnRight(-90);
    moveForward(-1 * LOW_FORWARD, 30);
    break;

case 4:     // CHECK 4
    setFuel();



    /*moveForward(LOW_FORWARD, 6);
    turnRight(90);
    moveForward(MID_FORWARD, 30);       // Right self before ramp
    moveForward(-1 * LOW_FORWARD, 6);
    turnRight(-90);

    moveForward(-1 * HIGH_FORWARD, 40); // Up the ramp we go



    turnRight(50);
    moveForward(-1 * LOW_FORWARD, 60);  */



    turnRight(30);
    moveForward(LOW_FORWARD, 18);
    turnRight(-75);
    moveForward(-1 * LOW_FORWARD, 17);
    turnRight(42);
    moveForward(-1 * HIGH_FORWARD, 46); // Up the ramp

    turnRight(110); //straighten on wall
    moveForward(MID_FORWARD, 25);
    leftMotor.SetPercent(HIGH_FORWARD);
    Sleep(FULL_SECOND);
    leftMotor.Stop();
```

```
        moveForward(-1 * LOW_FORWARD, 12);

        turnRight(-61);
        moveForward(-1 * LOW_FORWARD, 50);

        turnFuel();                     // Turn the fuel crank
        moveForward(LOW_FORWARD, 18);
        turnRight(-104);
        moveForward(LOW_FORWARD, 31);
        turnRight(49);                  //And right back down
        moveForward(LOW_FORWARD, 30);

        /*
        moveForward(LOW_FORWARD, 6);
        turnRight(-90);
        moveForward(MID_FORWARD, 20);
        moveForward(-1 * LOW_FORWARD, 4);
        turnRight(90);
        moveForward(HIGH_FORWARD, 100);
        */

    }

    LCD.Clear( FEHLCD::Black );
}

//Run some startup stuff
int startUp() {

    //Check screen
    LCD.Clear( FEHLCD::Black );
    LCD.SetFontColor( FEHLCD::White );
    LCD.WriteLine("Dan, up and running!");
    Sleep(FULL_SECOND);

    wrenchServo.SetMin(WRENCH_SERVO_MIN);
    wrenchServo.SetMax(WRENCH_SERVO_MAX);
    wrenchServo.SetDegree(30);

    fuelServo.SetMin(FUEL_SERVO_MIN);
    fuelServo.SetMax(FUEL_SERVO_MAX);
    fuelServo.SetDegree(90);
```

```
    //Decide what checks to run
    if(ask("Should I test my motors?")) {
        checkMotors();
    }

    RPS.InitializeTouchMenu();

    //Decide what course Dan should run
    int course;
    if (ask("The main test?")) {
        course = 0;
    } else if (ask("Should I run PT1?")) {
        course = 1;
    } else if (ask("How about PT2?")){
        course = 2;
    } else if (ask("PT3 then?")) {
        course = 3;
    } else if (ask("Time for PT4?")) {
        course = 4;
    } else {
        course = -1;
    }

    return course;
}

//Checks to see if the motors are functional
void checkMotors() {

    //Motor check

    LCD.WriteRC("Initiating Motor Check ", 1, 0);
    Sleep(HALF_SECOND);
    LCD.WriteRC(".", 1, 22);
    Sleep(HALF_SECOND);
    LCD.WriteRC(".", 1, 23);
    Sleep(HALF_SECOND);
    LCD.WriteRC(".", 1, 24);
    Sleep(HALF_SECOND);

    LCD.Clear( FEHLCD::Black );

    LCD.WriteLine("Moving Wrench Servo!");
```

```
wrenchServo.SetDegree(100);
Sleep(HALF_SECOND);
wrenchServo.SetDegree(50);
Sleep(FULL_SECOND);

LCD.WriteLine("Moving Fuel Servo!");
fuelServo.SetDegree(100);
Sleep(HALF_SECOND);
fuelServo.SetDegree(50);
Sleep(FULL_SECOND);

LCD.WriteLine("Ok!  Testing Forward!");
rightMotor.SetPercent(LOW_FORWARD);
leftMotor.SetPercent(LOW_FORWARD);
Sleep(FULL_SECOND);
rightMotor.Stop();
leftMotor.Stop();
Sleep(HALF_SECOND);

LCD.WriteLine("Now testing backwards!");
rightMotor.SetPercent(-1 * LOW_FORWARD);
leftMotor.SetPercent(-1 * LOW_FORWARD);
Sleep(FULL_SECOND);
rightMotor.Stop();
leftMotor.Stop();
Sleep(HALF_SECOND);

LCD.WriteLine("Alright, going right now!");
leftMotor.SetPercent(MID_FORWARD);
Sleep(FULL_SECOND);
leftMotor.Stop();
Sleep(HALF_SECOND);

LCD.WriteLine("And finally, left!");
rightMotor.SetPercent(MID_FORWARD);
Sleep(FULL_SECOND);
rightMotor.Stop();
Sleep(HALF_SECOND);
rightMotor.Stop();

LCD.Clear( FEHLCD::Black );

return;
```

```
}

//Display the current information from the robot
void showInfo() {

    if(testing) {
    LCD.WriteRC(lightSensor.Value(), 0, 0);
    LCD.WriteRC("X: ", 1, 0);
    LCD.WriteRC(RPS.X(), 1, 3);
    LCD.WriteRC("Y: ", 1, 10);
    LCD.WriteRC(RPS.Y(), 1, 13);
    LCD.WriteRC("Heading: ", 2, 0);
    LCD.WriteRC(RPS.Heading(), 2, 9);
    LCD.WriteRC("Fuel Type (1, Octane): ", 3, 0);
    //LCD.WriteRC(RPS.FuelType(), 3, 23);
    LCD.WriteRC("Deadzone Active: ", 4, 0);
    //LCD.WriteRC(RPS.IsDeadzoneActive(), 4, 17);
    } else {
        LCD.SetFontColor(FEHLCD::White);
        LCD.DrawCircle(159, 119, 59);
        LCD.SetFontColor(FEHLCD::Black);
    }



    return;

}

//Turns to a certain angle.  Negative = left
void turnRight(float angle) {

    //Figure out what direction to turn, set motors
    if(angle < 0) {
        LCD.WriteLine("I'm turning left!");

        rightMotor.SetPercent(OUTSIDE_TURN_POWER);
        leftMotor.SetPercent(INSIDE_TURN_POWER);

        Sleep(100);

        rightMotor.SetPercent(FAST_OUTSIDE);
```

```
        leftMotor.SetPercent(FAST_INSIDE);
    } else {
        LCD.WriteLine("Turning right!");
        rightMotor.SetPercent(FAST_INSIDE);
        leftMotor.SetPercent(FAST_OUTSIDE);
    }

    //Turn based on the angle desired
    Sleep(abs(angle) / TURN_ADJUST);
    rightMotor.Stop();
    leftMotor.Stop();

    return;
}

//Moves forward a certain distance (inch) at the given speed (percent). Negative PERCENT =
backwards
void moveForward(int percent, float distance)
{

  if(percent > 0) {
    rightMotor.SetPercent(12);
    leftMotor.SetPercent(12);
  } else {
     rightMotor.SetPercent(-12);
     leftMotor.SetPercent(-12);
  }


  Sleep(100);

  //Set both motors to desired percent
  rightMotor.SetPercent(percent);
  leftMotor.SetPercent(percent);

  //Sleep until the distance is reached
  Sleep(MOVE_ADJUST2 * distance / abs(percent) - 0.1);

  rightMotor.Stop();
  leftMotor.Stop();

  return;
}
```

```
//Looks for starting light
void readyStart() {

   //Makes sure the robot is in starting position
   /*while((START_X - 0.01 > RPS.X() || RPS.X() > START_X + 0.1) ||
         (START_Y - 0.01 > RPS.Y() || RPS.Y() > START_Y + 0.1) ||
         (START_HEADING - 0.5 > RPS.Heading() || RPS.Heading() > START_HEADING + 0.5))
{
      showInfo();

      //Show Desired Values
      LCD.WriteRC("Move me to...", 6, 0);
      LCD.WriteRC("X: ", 7, 0);
      LCD.WriteRC(START_X, 7, 3);
      LCD.WriteRC("Y: ", 7, 10);
      LCD.WriteRC(START_Y, 7, 13);
      LCD.WriteRC("Heading: ", 8, 0);
      LCD.WriteRC(START_HEADING, 8, 9);


      Sleep(100);
      LCD.Clear( FEHLCD::Black );
   }*/

   LCD.WriteLine("I'm in position, ready to start!");
   Sleep(FULL_SECOND);
   LCD.Clear( FEHLCD::Black );

   float timeStart = TimeNow();

   //Display light values as Dan waits for the light to turn on
   while(lightSensor.Value() > 0.9 && timeStart + 30 > TimeNow()) {
      showInfo();
      Sleep(100);
      LCD.Clear( FEHLCD::Black );
   }
   LCD.WriteLine("GO!");


   xCalibrate = START_X - RPS.X();
   yCalibrate = START_Y - RPS.Y();
   hCalibrate = START_HEADING - RPS.Heading();
```

```
    return;
}

//Checks light color
bool isBlue() {

    bool isBlue = false;

    //If the light has a value less than 0.8, it's blue
    if(lightSensor.Value() > 0.8) {
        isBlue = true;
    }

    return isBlue;
}

//Movement to press correct button
void pressButton(bool isBlue) {


    //After reading the color, make corresponding movements
    if(isBlue) {
        LCD.WriteLine("BLUE!");

        //Orient
        moveForward(MID_FORWARD, 1);
        turnRight(90);
        turnTo(270);
        //moveForward(MID_FORWARD,15);
        moveForward(100, 15);
        moveForward(-1 * LOW_FORWARD, 3);
        rightMotor.SetPercent(25);
        Sleep(250);
        rightMotor.Stop();
        Sleep(1.0);
        rightMotor.SetPercent(100);
        Sleep(6.0);
        rightMotor.Stop();

        //Revert
        moveForward(-1 * MID_FORWARD, 7);
        turnRight(100);
```

```
        turnTo(180);
        moveForward(MID_FORWARD, 7);


    } else {
        LCD.WriteLine("RED!");

        //Orient
        moveForward(-1 * MID_FORWARD, 6);
        turnRight(90);
        turnTo(270);
        //moveForward(MID_FORWARD,15);
        moveForward(100, 15);
        moveForward(-1 * LOW_FORWARD, 3);
        leftMotor.SetPercent(25);
        Sleep(250);
        leftMotor.Stop();
        Sleep(1.0);
        leftMotor.SetPercent(100);
        Sleep(6.0);
        leftMotor.Stop();

        //Revert
        moveForward(-1 * MID_FORWARD, 7);
        turnRight(60);
        turnTo(180);

    }


}

//If it's negative, make it positive
float abs(float value) {

    //If negative, multiply by -1
    if(value < 0) {
        value *= -1;
    }

    return value;
}
```

```
//Ask the user a yes/no question
bool ask(char question[]) {

    bool answer = true;

    //Ask the question on screen
    LCD.WriteRC(question, 3, 0);
    LCD.WriteAt("Yes", 70, 130);
    LCD.WriteAt("No", 190, 130);

    float x, y;

    //Wait until the user presses the screen
    while(!LCD.Touch(&x,&y));

    //Take user input, false = no
    if(x > 130) {
        answer = false;
    }
    Sleep(100);

    LCD.Clear( FEHLCD::Black );

    return answer;
}

//Sets servo angle based on fuel type
void setFuel() {
    if(RPS.FuelType() == 1) {
        // ROTATE CLOCKWISE SETUP
        fuelServo.SetDegree(0);
        isClockwise = true;
    } else {
        // ROTATE COUNTERCLOCKWISE SETUP
        fuelServo.SetDegree(180);
        isClockwise = false;
    }
}

//Turns crank based on fuel type
void turnFuel() {

    if(isClockwise) {
```

```
        fuelServo.SetDegree(180);
       moveForward(-1 * LOW_FORWARD,20);
        moveForward(LOW_FORWARD, 10);
        fuelServo.SetDegree(160);
        moveForward(-1 * LOW_FORWARD, 15);
        fuelServo.SetDegree(180);
    } else {
        fuelServo.SetDegree(0);
        moveForward(-1 * LOW_FORWARD,20);
        moveForward(LOW_FORWARD, 10);
        fuelServo.SetDegree(30);
        moveForward(-1 * LOW_FORWARD, 15);
        fuelServo.SetDegree(0);
    }



    //moveForward(-1 * MID_FORWARD, 10);//was 35
    Sleep(2.0);
}


//Main path move to wrench
void mainToWrench() {

    wrenchServo.SetDegree(110);

    rightMotor.SetPercent(23);
    leftMotor.SetPercent(80);
    Sleep(0.8);

    rightMotor.SetPercent(100);
    leftMotor.SetPercent(100);
    Sleep(0.3);

    wrenchServo.SetDegree(40);

    rightMotor.Stop();
    leftMotor.Stop();

    Sleep(FULL_SECOND);
}
```

```
//Main path move to jack
void wrenchToJack() {

   rightMotor.SetPercent(-100);
   leftMotor.SetPercent(-100);
   Sleep(0.05);

   rightMotor.SetPercent(40);
   Sleep(0.3);

   rightMotor.Stop();
   leftMotor.Stop();
}

//Turn to a direction using RPS
void turnTo(float angle) {


   int check = 0;
   bool stop = false;
   float start = TimeNow();

   while ((RPS.Heading() + hCalibrate  > angle + 1 || RPS.Heading() + hCalibrate < angle - 1)
&& !stop && (start + TIMEOUT > TimeNow())) {
      showInfo();
      if (RPS.Heading() > angle) {



         if ((RPS.Heading() + hCalibrate - angle) > (angle + 360 - RPS.Heading() + hCalibrate)) {
            // TURN LEFT

            if (check == -1) {
               stop = true; //count++
            }

            check = 1;

            rightMotor.SetPercent(OUTSIDE_TURN_POWER);
            leftMotor.SetPercent(INSIDE_TURN_POWER);
         } else {
            // TURN RIGHT
```

```
        if (check == 1) {
            stop = true; //count++
        }

        check = -1;


        rightMotor.SetPercent(INSIDE_TURN_POWER);
        leftMotor.SetPercent(OUTSIDE_TURN_POWER);
    }

} else {



    if ((angle - RPS.Heading() + hCalibrate) > (RPS.Heading() + hCalibrate + 360 - angle)) {
        // TURN RIGHT

        if (check == 1) {
            stop = true; //count++
        }

        check = -1;

        rightMotor.SetPercent(INSIDE_TURN_POWER);
        leftMotor.SetPercent(OUTSIDE_TURN_POWER);
    } else {
        //TURN LEFT

        if (check == -1) {
            stop = true; //count++
        }

        check = 1;

        rightMotor.SetPercent(OUTSIDE_TURN_POWER);
        leftMotor.SetPercent(INSIDE_TURN_POWER);
    }

}

Sleep(100);
```

```
        rightMotor.Stop();
        leftMotor.Stop();

        Sleep(100);

    }

    rightMotor.Stop();
    leftMotor.Stop();
}

//Move to an x coord using RPS
void moveToX(float x) {

    int direction = 1;
    int check = 0;
    bool stop = false; //int count = 0;
    float start = TimeNow();

    if(RPS.Heading() > 90 && RPS.Heading() < 270) {
        //Facing the negative x direction
       direction = -1;

    }

    while( (RPS.X() + xCalibrate < x - 0.3 || RPS.X() + xCalibrate > x + 0.3 ) && !stop && (start +
TIMEOUT > TimeNow()))  { //&&count < 3

        if( RPS.X() < x) {
            if (check == -1) {
                stop = true; //count++
            }

            check = 1;
            rightMotor.SetPercent(direction * 12);
            leftMotor.SetPercent(direction * 12);

        } else {
            if (check == 1) {
                stop = true; //count++
            }
```

```
            check = -1;
            rightMotor.SetPercent(direction * -12);
            leftMotor.SetPercent(direction * -12);

        }

        Sleep(50);

        rightMotor.Stop();
        leftMotor.Stop();

        Sleep(100);

    }
}

//Move to an y coord using RPS
void moveToY(float y) {

    int direction = 1;
    int check = 0;
    bool stop = false;
    float start = TimeNow();

    if(RPS.Heading() < 360 && RPS.Heading() > 180) {
        //Facing the negative y direction
       direction = -1;

    }

    while( (RPS.Y() + yCalibrate < y - 0.3 || RPS.Y() + yCalibrate > y + 0.3)  && !stop && (start +
TIMEOUT > TimeNow()))  {


        if( RPS.Y() < y) {

            if (check == -1) {
                stop = true; //count++
            }

            check = 1;

            rightMotor.SetPercent(direction * 12);
```

```
            leftMotor.SetPercent(direction * 12);

        } else {

            if (check == 1) {
                stop = true; //count++
            }

            check = -1;

            rightMotor.SetPercent(direction * -12);
            leftMotor.SetPercent(direction * -12);

        }

        Sleep(50);

        rightMotor.Stop();
        leftMotor.Stop();

        Sleep(100);

    }


}
```

APPENDIX I

Budget

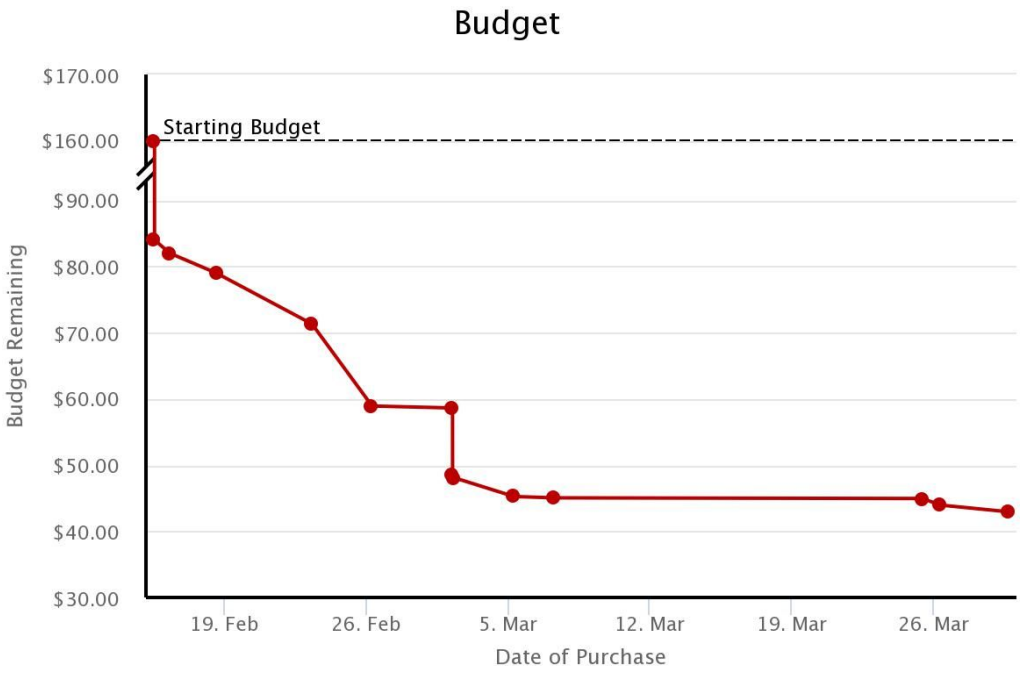
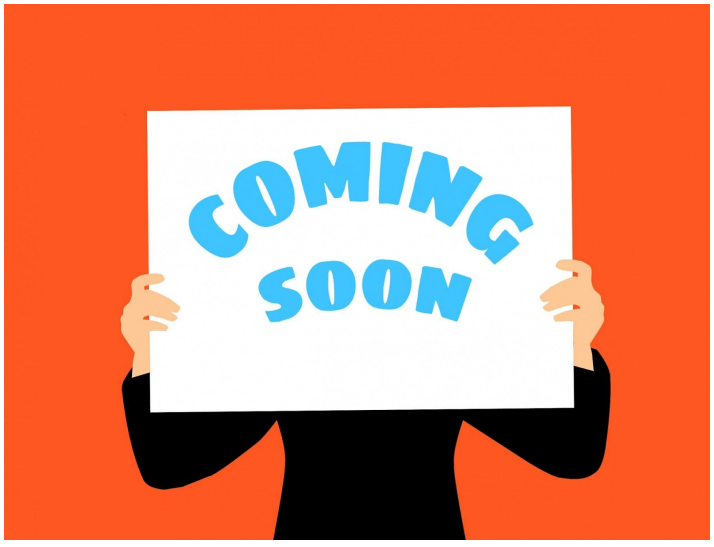


Figure I1: Graph of budget remaining over time [4].

Table I1: Log of purchases.



APPENDIX J

Schedule

Table J1: Design schedule.

#	DAL Number	Task	Scheduled Start	Scheduled Finish	Actual Start	Actual Finish	Due Date	Colby	Jack	Justin	Kelly	Status	Estimated Time (hrs)	Actual Time (hrs)
1	R01	**Individual Brainstorming**	1/26/2018	1/30/2018	1/26/2018	1/30/2018	1/31/2018					100%	2	2
2	R02	**Working Agreement**	1/31/2018	2/1/2018	1/31/2018	2/1/2018	2/2/2018					100%	0.5	0.5
3		**Team Crest**	1/31/2018	2/2/2018	1/31/2018	1/31/2018	2/2/2018					100%	0.5	0.5
4	R03	**Group Brainstorming**	1/31/2018	2/6/2018	2/2/2018	2/6/2018	2/7/2018					100%	1.5	2.2
4A		List Ideas	2/2/2018	2/2/2018	2/2/2018	2/2/2018	2/7/2018					100%	0.25	0.5
4B		Idea Screening	2/2/2018	2/6/2018	2/2/2018	2/6/2018	2/7/2018					100%	0.25	0.75
4C		Concept Scoring	2/6/2018	2/6/2018	2/6/2018	2/6/2018	2/7/2018					100%	0.25	0.25
4D		Descriptions and Sketches	2/6/2018	2/6/2018	2/6/2018	2/6/2018	2/7/2018					100%	0.5	0.4
4E		Strategies	2/6/2018	2/6/2018	2/5/2018	2/6/2018	2/7/2018					100%	0.25	0.3
5	R04	**Design Schedule**	2/2/2018	2/5/2018	2/3/2018	2/7/2018	2/9/2018					100%	1	3
	R05	**Mockup / Solid Model**	2/2/2018	2/11/2018	2/7/2018	2/11/2018	2/12/2018					100%	3	3
6		Physical Model	2/2/2018	2/11/2018	2/7/2018	2/8/2018	2/12/2018					100%	2	1.5
7		SolidWorks Model	2/2/2018	2/11/2018	2/11/2018	2/11/2018	2/12/2018					100%	1	1.5
8	R06	**Drivetrain Analysis**	2/9/2018	2/13/2018	2/12/2018	2/12/2018	2/14/2018					100%	0.75	0.9
8A		Part I: Speed	2/9/2018	2/13/2018	2/12/2018	2/12/2018	2/14/2018					100%	0.25	0.3
8B		Part II: Torque	2/9/2018	2/13/2018	2/12/2018	2/12/2018	2/14/2018					100%	0.25	0.5
8C		Part III: Motor Selection	2/9/2018	2/13/2018	2/12/2018	2/12/2018	2/14/2018					100%	0.25	0.1
9	R07	**Exploration 1 Report (Individual)**	2/9/2018	2/15/2018	2/9/2018	2/15/2018	2/16/2018					100%	3	1
10	R08	**Agendas and Notes**	2/2/2018	2/18/2018	2/2/2018	2/18/2018	2/19/2018					100%	0.55	0.45
10A		Keep Agenda	2/2/2018	2/18/2018	2/2/2018	2/18/2018	2/19/2018					100%	0.25	0.1
10B		Take Notes	2/2/2018	2/18/2018	2/2/2018	2/18/2018	2/19/2018					100%	0.25	0.25
10C		Upload to Project Portfolio	2/2/2018	2/18/2018	2/6/2018	2/18/2018	2/19/2018					100%	0.05	0.1
11	R09	**Final Report Outline**	2/19/2018	2/20/2018	2/19/2018	2/20/2018	2/21/2018					100%	3	0.5
12	R10-1	**Report for Peer Feedback**	2/19/2018	2/25/2018	2/23/2018	2/26/2018	2/26/2018					100%	1.05	7.05
12A		Write Intro & Prelim. Concepts	2/19/2018	2/25/2018	2/23/2018	2/26/2018	2/26/2018					100%	1	7
12B		Print 4 Copies and Rubric	2/19/2018	2/25/2018	2/26/2018	2/26/2018	2/26/2018					100%	0.05	0.05
13	R10-2	**Peer Review**	2/26/2018	3/4/2018	2/26/2018	3/5/2018	3/5/2018					100%	1.25	0.9
13A		Peer Feedback Form	2/26/2018	3/4/2018	2/26/2018	3/2/2018	3/5/2018					100%	0.5	0.25
13B		Group Discussion	2/26/2018	3/4/2018	3/2/2018	3/2/2018	3/5/2018					100%	0.25	0.15
13C		Summary	2/26/2018	3/4/2018	3/2/2018	3/5/2018	3/5/2018					100%	0.5	0.5
		Performance Test 1	2/14/2018	2/22/2018	2/14/2018	2/23/2018	2/23/2018					100%	8	9
14		Navigation to Car Jack	2/14/2018	2/22/2018	2/14/2018	2/23/2018	2/23/2018					100%	4	7
15		Car Jack Flip	2/14/2018	2/22/2018	2/14/2018	2/23/2018	2/23/2018					100%	2	1
16		Move to Top Level	2/14/2018	2/22/2018	2/15/2018	2/23/2018	2/23/2018					100%	2	1
17	R11	**Exploration 2 Report (Individual)**	2/16/2018	2/22/2018	2/16/2018	2/22/2018	2/23/2018					100%	4	1.5
18	R12	**Code Representation**	2/21/2018	2/25/2018	2/23/2018	2/25/2018	2/26/2018					100%	13	8.5
18A		Write Code	2/21/2018	2/25/2018	2/10/2018	2/25/2018	2/26/2018					100%	10	7
18B		Represent Code Logic	2/21/2018	2/25/2018	2/23/2018	2/25/2018	2/26/2018					100%	2	1
18C		Describe Code and Logic	2/21/2018	2/25/2018	2/25/2018	2/25/2018	2/26/2018					100%	1	0.5
19	R13	**Budget and Testing Log**	2/10/2018	2/17/2018	2/10/2018	2/17/2018	2/18/2018					100%	2.25	3
19A		Budget Summary	2/10/2018	2/17/2018	2/10/2018	2/17/2018	2/18/2018					100%	1	1
19B		Time Summary	2/10/2018	2/17/2018	2/17/2018	2/17/2018	2/18/2018					100%	1	1
19C		Sample Page of Testing Log	2/10/2018	2/17/2018	2/17/2018	2/17/2018	2/18/2018					100%	0.25	1
		Performance Test 2	2/22/2018	3/1/2018	2/22/2018	2/23/2018	3/2/2018					100%	6	0.75
20		Navigate to Button Panel	2/22/2018	3/1/2018	2/22/2018	2/23/2018	3/2/2018					100%	2	0.25
21		Read Light	2/22/2018	3/1/2018	2/22/2018	2/23/2018	3/2/2018					100%	2	0.25
22		Push Correct Button	2/22/2018	3/1/2018	2/22/2018	2/23/2018	3/2/2018					100%	2	0.25
23	R14	**Exploration 3 Report (Individual)**	2/28/2018	3/6/2018	2/28/2018	3/7/2018	3/7/2018					100%	4	1.5
		Performance Test 3	3/2/2018	3/8/2018	2/28/2018	3/2/2018	3/9/2018					100%	7	1.75
24		Navigate to Wrench	3/2/2018	3/8/2018	2/28/2018	3/2/2018	3/9/2018					100%	2	0.25
25		Remove Wrench from Holder	3/2/2018	3/8/2018	2/28/2018	3/2/2018	3/9/2018					100%	2	0.5
26		Deposit Wrench in Garage	3/2/2018	3/8/2018	2/28/2018	3/2/2018	3/9/2018					100%	3	1
27	R15	**Final Report First Draft**	3/5/2018	3/20/2018	2/19/2018	3/20/2018	3/21/2018					100%	5	10
		Performance Test 4	3/9/2018	3/22/2018	3/2/2018	3/23/2018	3/23/2018					100%	5	12
28		Navigate to Fuel Pump	3/9/2018	3/22/2018	3/2/2018	3/23/2018	3/23/2018					100%	2	8
29		Turn Crank 180 Degrees	3/9/2018	3/22/2018	3/2/2018	3/23/2018	3/23/2018					100%	3	4
30	R16	**Electrical Systems**	3/7/2018	3/25/2018	3/21/2018	3/21/2018	3/26/2018					100%	4	0.5
30A		Diagram	3/7/2018	3/25/2018	3/21/2018	3/21/2018	3/26/2018					100%	2	0.25
30B		Tables	3/7/2018	3/25/2018	3/21/2018	3/21/2018	3/26/2018					100%	2	0.25
31	R17	**Final Report Second Draft**	3/21/2018	4/1/2018	3/20/2018		4/2/2018					70%	5	8
32	R18	**Isometric for Display**	3/30/2018	4/3/2018	3/17/2018	3/28/2018	4/4/2018					100%	3.25	6.15
32A		CAD Drawing	3/30/2018	4/3/2018	3/17/2018	3/28/2018	4/4/2018					100%	3	6
32B		Qt Creator Installation	3/30/2018	4/3/2018	12/5/2017	12/5/2017	4/4/2018					100%	0.25	0.15
33	R19	**Working Drawing Set**	3/23/2018	4/15/2018			4/16/2018					0%	14	0
33A		Non-exploded Assembly	3/23/2018	4/15/2018			4/16/2018					0%	2	
33B		Exploded Assembly	3/23/2018	4/15/2018			4/16/2018					0%	3	
33C		Subassemblies	3/23/2018	4/15/2018			4/16/2018					0%	2	
33D		Bill of Materials	3/23/2018	4/15/2018			4/16/2018					0%	1	
33E		Detail Drawings	3/23/2018	4/15/2018			4/16/2018					0%	6	
34	R20	**Oral Report**	3/21/2018	4/17/2018			4/18/2018					0%	2	
35	R21	**Final Written Report**	2/19/2018	4/22/2018			4/23/2018					0%	7	
36	R22	**Project Portfolio**	1/31/2018	4/22/2018			4/23/2018					0%	1.85	0.05
36A		Navigation Menu	1/31/2018	4/22/2018			4/23/2018					0%	0.25	
36B		Organizational Pages	1/31/2018	4/22/2018			4/23/2018					0%	0.25	
36C		Team Contact Info	1/31/2018	4/22/2018	2/6/2018	2/6/2018	4/23/2018					100%	0.1	0.05
36D		Design Schedule	1/31/2018	4/22/2018	2/19/2018	2/19/2018	4/23/2018					100%	0.1	
36E		Budget	1/31/2018	4/22/2018	2/19/2018	2/19/2018	4/23/2018					100%	0.1	
36F		Agendas and Notes	1/31/2018	4/22/2018	2/6/2018		4/23/2018					5%	0.25	
36G		Sketches, Models, Drawings	1/31/2018	4/22/2018			4/23/2018					0%	0.25	
36H		Electrical Systems	1/31/2018	4/22/2018			4/23/2018					0%	0.1	
36I		Code	1/31/2018	4/22/2018			4/23/2018					0%	0.1	
36J		Code Representation	1/31/2018	4/22/2018			4/23/2018					0%	0.1	
36K		Testing Logs	1/31/2018	4/22/2018	2/21/2018		4/23/2018					0%	0.1	
36L		Testing Results	1/31/2018	4/22/2018			4/23/2018					0%	0.1	
36M		References	1/31/2018	4/22/2018			4/23/2018					0%	0.05	

Created:	2/3/2018
Last Updated:	4/1/2018 2:06 PM Kelly Meaden
Primary	
Secondary	

	Sunday				Monday				Tuesday				Wednesday				Thursday				Friday				Saturday			
	JB	JM	KM	CW	JB	JM	KM	CW	JB	JM	KM	CW	JB	JM	KM	CW	JB	JM	KM	CW	JB	JM	KM	CW	JB	JM	KM	CW
12:00 AM																												
12:30 AM																												
1:00 AM																												
1:30 AM																												
2:00 AM																												
2:30 AM																												
3:00 AM																												
3:30 AM																												
4:00 AM																												
4:30 AM																												
5:00 AM																												
5:30 AM																												
6:00 AM																												
6:30 AM																												
7:00 AM																												
7:30 AM																												
8:00 AM																												
8:30 AM																												
9:00 AM																												
9:30 AM																												
10:00 AM																												
10:30 AM																												
11:00 AM																					brainstorming							
11:30 AM																												
12:00 PM																												
12:30 PM																												
1:00 PM																												
1:30 PM																												
2:00 PM																												
2:30 PM																												
3:00 PM																												
3:30 PM																												
4:00 PM																												
4:30 PM																												
5:00 PM																												
5:30 PM																												
6:00 PM																												
6:30 PM																												
7:00 PM																												
7:30 PM																												
8:00 PM																												
8:30 PM																												
9:00 PM																												
9:30 PM																												
10:00 PM																												
10:30 PM																												
11:00 PM																												
11:30 PM																												

Category	Total	JB	JM	KM	CW
Documentation	2.5	0.5	2	0	0
Project Management	16	3.5	5	5	2.5
Coding	0	0	0	0	0
Testing	0	0	0	0	0
CAD	0	0	0	0	0
Building	0	0	0	0	0
Other	7	2.5	1.5	1.5	1.5
Total Hours:	25.5	6.5	8.5	6.5	4

Table J2: Weekly timesheet for Jan. 27 - Feb. 3.

Table J3: Testing log.