

ME 588 - Mechatronics

Autonomous Whack-A-Mole Robot Project

Final Report

May 2nd, 2022

Team 13

Collin Feustel

Wendy Kim

Joel Kong

Norawish Lohitnavy

Arijeet Nath

Table of Contents

Abstract	3
Introduction	4
Design Details.....	5
Robot Behavior.....	5
Robot Chassis	5
Mole Whacker	7
Driving Mechanism and Power Distribution	9
Sensor Inputs.....	9
User Inputs.....	11
Outputs and Display	12
Results	13
Discussion	15
Conclusion.....	16
Appendix I – Design Ideation Diagrams	17
Appendix II – FSM State Transition Diagram	19
Appendix III – Arduino Code Examples.....	20
Appendix IV – Arduino Mega Pin Designation	21
Appendix V – Robot Components and Sensor Selection Sheet	23
Appendix VI – Completed Component Budget and Sourcing Plan.....	25
Appendix VII – Subassembly Wiring Diagrams.....	26
Appendix VIII – Manufactured Part Drawings.....	28
Appendix IX – Microsoft Project Team Schedule	35

Abstract

As the capstone project for ME 588 (Mechatronics), the goal of this project was to apply all previously acquired mechatronics knowledge towards the development of an autonomous robot to complete a game of “whack-a-mole”. The project team was given a set of engineering requirements and constraints to limit the overall size and cost of the robot, as well as implement the desired “whack-a-mole” behavior of autonomously driving around a game field and placing five foam blocks onto five tiles of a desired color. Key functional requirements of the robot included having an input and output to indicate color selection, an input and output to indicate the start of a new game, the inclusion of an emergency stop button, the inclusion of a unique visual indicator, and two unique sensing modalities. The final design and CAD model of the team’s robot can be seen in Figure 1. The robot utilized two IR line sensors, an RGB color sensor, and two DC motor encoders to achieve the desired autonomous behavior. Team 13’s robot was successfully and accurately able to complete the “whack-a-mole” game in approximately 55 seconds and won second place in the course-wide robotics competition.

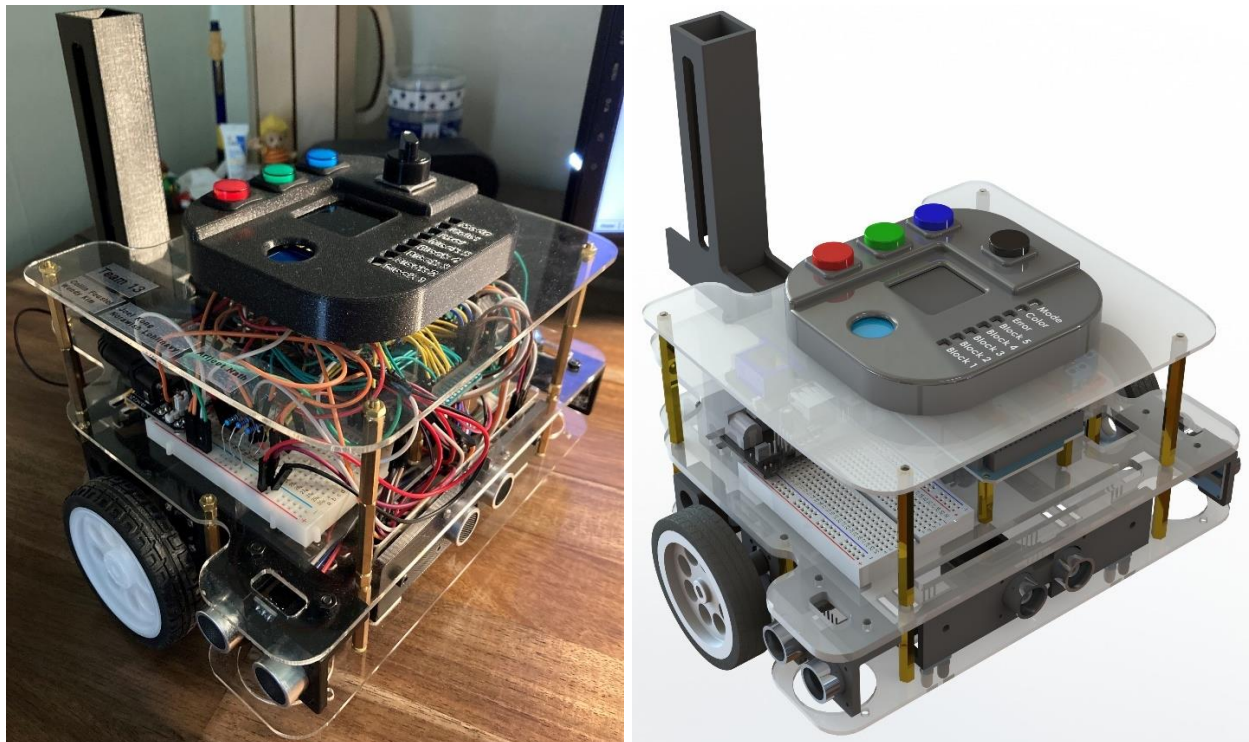


Figure 1: Team 13's completed whack-a-mole robot (left) and matching final CAD render (right)

Introduction

The goal of the project was to design, manufacture, assemble, and control an autonomous whack-a-mole robot that accepts user inputs for a desired color and navigates across a game field to drop foam cubes onto tiles of a specified color. The 8ft x 8ft game field is comprised of 16 squares – five blue tiles, five red tiles, five yellow tiles, and a single white starting tile. The design of the mole whacker had to drop five foam cubes within the 2ft x 2ft area of each colored tile. The robot was required to stop moving after two minutes regardless of whether the robot was able to drop all the cubes.

To achieve the project objective, the team went through several brainstorming sessions to develop possible solutions for each of the desired robot tasks. By the end of the brainstorming phase, the team decided to rely primarily on two DC motor encoders to drive the robot through a predetermined path. The predetermined path starts at the white starting tile and goes around the game field while only driving over each tile once. The predetermined path can be seen in Figure A.I.1. As the robot follows the path, the color of each tile is scanned, and moles are dropped onto the desired tile color. It was decided that the foam cubes would be used for the moles (instead of foam balls) due to their modularity for storage and controllability for dropping. To achieve the autonomous motion of the robot within the game field, two IR line sensors, an RGB color sensor, and two DC motor encoders were used in a complementary fashion. For the user input, a capacitive touch sensor, three colored LED buttons, and a simple latching switch were implemented onto a custom 3D-printed shroud on the top of the robot. The mole whacker subsystem has a piston mechanism that was driven by a 9g servo motor to drop all the cubes. A three-layer chassis was designed using three acrylic sheets to organize all components, and the chassis was driven by two DC motors. To mount all electronic components to the chassis of the robot, several mounting components were designed and 3D-printed.

Subsystem calibration tests were conducted prior to the final assembly of the robot to ensure that all components were working as expected. A finite state machine (FSM) was then developed to govern the behavior of the robot. A total of five states were used to account for all potential actions that the robot might take. Arduino code created while calibrating each subsystem was then implemented in the final FSM. An Arduino Mega was chosen to act as the microcontroller due to the large number of digital and analog pins available for the multitude of inputs and outputs that the robot required.

Design Details

Robot Behavior

Before designing the chassis of the robot, the behavior of the robot was first determined by combining subsystem concepts from each team member. The behavior of the robot consists of the movement of the robot, the selection of different sensors, the method of dropping the foam cubes, and various user inputs and outputs. Two IR line sensors, one RGB sensor, and two DC motors with encoders were selected to aid in controlling the path of the robot; three colored LED buttons and a capacitive touch sensor were used to receive user inputs; two DC motors and one servo motor were used to actuate the robot; an LCD screen, an RGBW strip, and three LED buttons were used as outputs. The complete electronic component selection process is illustrated in the Excel sheet in Appendix V.

As for robot operation, a state transition diagram was developed as seen in Appendix II. The transition diagram acted as a template for the creation of the final Arduino FSM program. To initiate the operation of the robot, the user selects one of the three colors on the field using one of the three colored LED buttons (State 0 to State 1). The user can then press the capacitive touch sensor to start the whack-a-mole game (State 1 to State 2). For the motion of the robot, a predetermined path was chosen. As shown in Appendix I, Figure A.I.1, the robot drives over each of the colored tiles with the minimum number of turns while only driving over each tile once. Velocity and position PID controllers were implemented using the two motor encoders to ensure the robot drives along the predetermined path in a controlled and predictable manner. The two IR line sensors were used to detect the black electrical tape on the floor of the game field to indicate when the robot had entered a new colored tile and when to stop the motion of the robot (State 2 to State 3). The use of motor encoders and IR line sensors acted as a relative location system for the robot. As the robot drives into a new tile, all motion is stopped, and the color sensor is used to detect the color of the current tile. If the color of the tile matches the selected color, the servo motor is activated to drive the mole whacker to drop a mole (State 3 to State 4). Otherwise, the robot continues to the next tile (State 3 to State 2). The game ends after two minutes have elapsed or if the robot has successfully completed its predetermined path.

Robot Chassis

To organize all electronic components, the robot was designed to have three layers. The three-layer design offers modularity and allows for easy repair if wiring must be changed, or if a

hardware failure occurs. The bottom baseplate of the robot includes all components needed for driving and all sensor inputs; the second layer acts as a central location for the Arduino Mega microcontroller, all wiring, and the mole whacker; the top layer contains all components needed for display and user inputs. Sensor and motor mounts were designed to attach all components to the layers. The designs for the baseplates and the mounts were done in SolidWorks and Fusion 360 and the individual components can be seen in the engineering drawings in Appendix VIII. Once the mounts were 3D-printed and tolerances were adjusted, an M3 hardware mounting scheme was designed to attach all mounts to the main robot chassis. Cutouts were included in each of the baseplates to aid in wire management. The three baseplates were laser-cut from a piece of 0.125in acrylic and assembled into the main chassis of the robot. The layers were separated using M3 standoffs of various lengths to provide enough room for wiring and for the electronic components. Several shrouds were designed and 3D-printed to streamline the operation of the robot: a lens shroud was designed to fit over the RGB sensor to direct LED light and color-reading in a controlled manner to avoid detecting extraneous colors, and a user input/output shroud was design to organize and display the status of the robot.

The layout of the bottom layer was crucial to the overall stability of the robot. All heavy components were placed on this bottom layer to lower the robot's center of mass. This was done to reduce the risk of tipping whenever the robot was accelerating, stopping, or turning. For the wheelbase, a 3-wheel structure was chosen to maintain the balance of the robot. Two 65mm wheels were used as the main driving wheels and a single idler wheel was placed at the back of the chassis. The driving wheels were designed to be as close to the center of the robot as possible to achieve neutral robot steering. The bottom layer of the robot can be seen in Figure 2. For the sensor locations, two IR sensors were placed at the front of the underside of the robot, while the RGB sensor was placed slightly behind the IR sensors. The relative positioning of the IR line sensors and the RGB sensor can be seen in Figure 3 and complements the designed robot behavior which was previously mentioned (reading black lines of arena prior to reading tile color). The two batteries were placed at the back of the chassis to provide a good source of counterweight and further improve the stability of the robot.

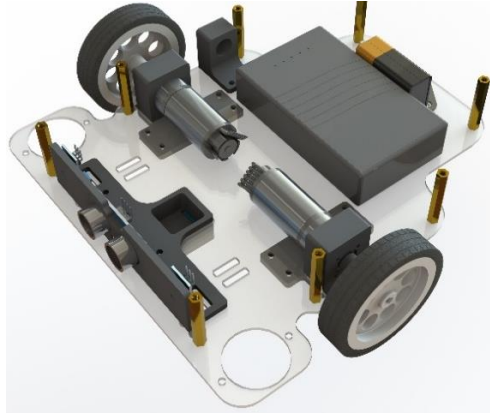


Figure 2: First layer and baseplate of robot – both batteries mounted at the back of the robot, DC motors and wheels mounted to baseplate, all input sensors at front of robot, added charging port at back of robot, single caster wheel at back

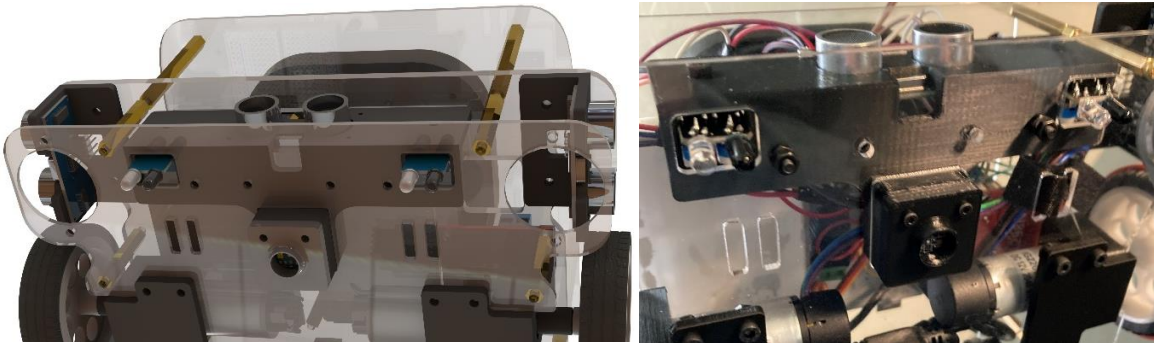


Figure 3: Underside of robot - two IR line sensors at front of robot with a single RGB color sensor/shroud at the robot center

Mole Whacker

An important requirement of the robot was to be able to drop the provided foam moles inside the bounded areas of the colored tiles. To satisfy this requirement, a mole whacker subsystem was designed. Figure A.I.2 in Appendix I shows the design iteration process for the mole whacker and Figure 4 shows the final CAD model for the mole whacker subsystem.

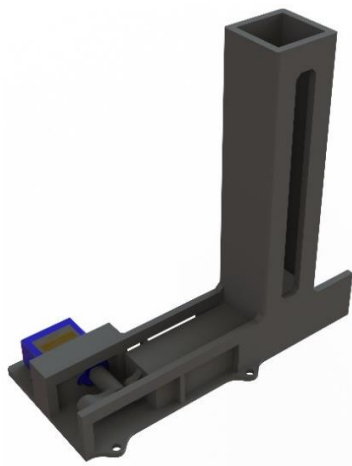


Figure 4: Mole whacker subsystem – 9g servo motor used to drive the motion of the piston and linkages, all components 3D printed using PLA

The 3D-printed mole whacker is comprised of three main components: a vertically placed cube cartridge, a pusher, and the crank motion linkage bars. To limit the length of the pusher as well as to increase the drop accuracy, it was desired that each cube has the same pre-drop location. The vertical placement of the cartridge allows for the cubes to fall into the same dropping position. Compared to a horizontal placement method, the vertical placement method utilized in the team's robot provides more stability and accuracy. The cube cartridge was designed to hold six cubes – The five foam cubes and one 3D-printed cube to serve as a weight to push down the foam cubes. At the bottom of the mole whacker there are two side walls to further constrain the path of the cubes. The pusher was designed to be slightly shorter than the height of the cubes, which ensures that the next cube falls into the pre-dropping location as the current cube is being dropped. As the pusher retracts, the next cube falls into the dropping position without interference. Two fins were attached to the sides of the pusher to fit inside of a guide in the cartridge to constrain the movement of the pusher. The mole whacker features a crank motion linkage system that converts the rotational motion from the servo into the linear motion of the pusher. Three stages of the mole whacker operation can be seen in Figure 5.

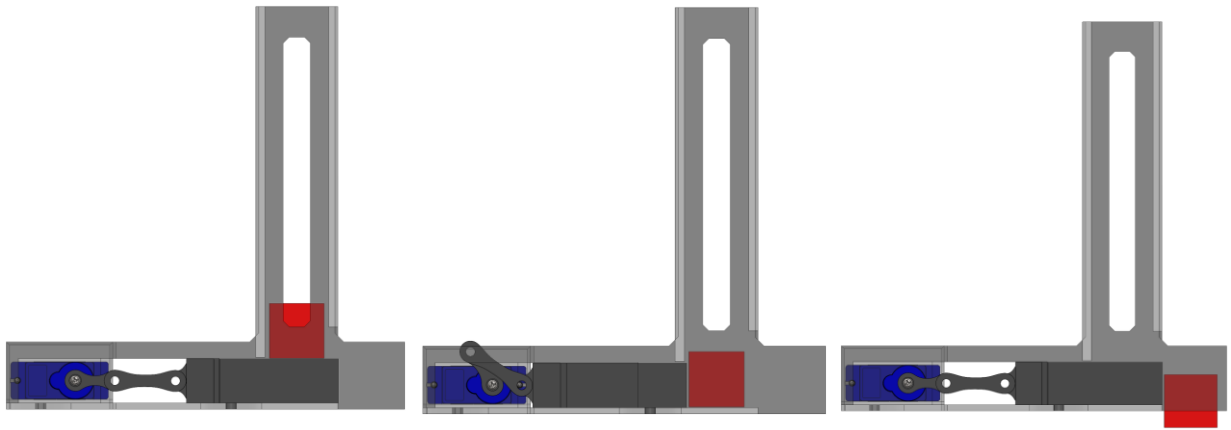


Figure 5: Operation of mole whacker subsystem – leftmost image shows starting position of mole whacker with a loaded cube, middle image shows the pulled-back piston, the rightmost image shows the final position of the mole whacker and the dropped cube

The crank system is driven by a 9g servo motor, which has a range of motion limited to 0-180°. As seen in Figure A.III.1 from Appendix III, for() loops were used to control the servo motor to retract 180° and extend 180° whenever it was desired for a mole to be placed. A while() loop timer was used to ensure that the servo had enough time to drop the cube before the motion of the robot was continued.

Driving Mechanism and Power Distribution

An L298N dual H-bridge was used to control the forward and backward motion of the two DC motors. The motor driver board contained all necessary flyback diodes and capacitors needed to protect the 290-006 DC motors. Thus, additional circuit calculations were not needed. Both DC motors had built-in Hall-Effect encoders with a resolution of 230 counts per revolution which were used to control the velocity and position of the robot within the play field. Four digital outputs from the Arduino Mega were used to act as inputs to the H-bridge to control the direction of each motor, while four hardware interrupt pins were used for the Hall-Effect encoders. Two PWM pins were used for the two enable pins of the L298N controller and the PWM signal of each was varied to control motor speeds. Velocity and position PID controllers were developed using the motor encoder readings as inputs. The velocity controller was used for straight-path driving because it was desired to change the speed of the robot to improve consistency and/or reduce the amount of time required to finish the game. The position controller was used for turning to allow for more precise 90° turns. The Arduino code implemented for these controllers is shown in Appendix III, Figure A.III.2, which depicts the proportional, integral, and derivative gains for each controller, as well as how errors are determined. If it were desired to change the velocity of the motors, targetVel (Line 283) would be changed. If it were desired to change the position of the motors, enc (Line 288) would be changed.

For power, two separate batteries were used. A 12V 3000mA battery was used to power all sensing, input, and output components while a rechargeable 9V battery was used to power the Arduino Mega to avoid any potential noise. While the L298N motor driver was powered using the 12V battery, a breadboard power adapter was used to step-down the 12V provided by the 3000mA battery to 3.3V and 5V power lines on the mini breadboard to power the LEDs and onboard sensors. A common ground was established between all power sources and the Arduino Mega microcontroller. The rechargeable 9V battery in Figure 2 was fastened to the back of the 12V battery using Velcro to optimize the process of swapping out and recharging the battery.

Sensor Inputs

- *Infrared Barrier Sensors*

The Infrared Barrier Sensor by RobotDyn was used to sense the boundaries of the tiles in the game field. The IR sensor consists of an IR emitter and an IR receiver. There are two onboard potentiometers which can be used to vary the frequency of the light pulse emitted and to control

the threshold distance for detecting obstacles by the receiver. The emitter emits signals at a certain frequency. When this signal hits an obstacle, it bounces back, which is then detected by the IR receiver. The sensor output returns a logic low when the receiver detects an obstacle. Since it is used to detect the black lines that mark the boundaries of the colored tiles, the sensor will return a logic high since the black tape will absorb all IR light from the pulse sent out by the emitter. The sensor consists of a 5V pin (Vcc), one ground pin (GND), one output pin (OUT), and an enable pin (TX IN). The enable pin is used, as the name suggests, to enable the IR sensor. This means that if the sensor is enabled, it will be constantly detecting obstacles. For most sensors sending a logic high signal to the ENB pin enables the sensor, but in this case, sending a logic low signal is necessary to enable the sensor.

- *RGB Color Sensor*

The colors of the tiles in the game field are red, blue, and yellow. The TCS3472 sensor was chosen as the RGB sensor due to the dynamic range of color that can be detected and its IR blocking filter which allows color values to be measured with increased precision. This sensor returns the digital values of the detected RGB colors and clear light. It uses an I2C protocol for data communication with the Arduino, which is preferred when establishing low-speed communications over short distances. To achieve this, the sensor uses two lines for sending and receiving data, namely a serial clock pin (SCL) and a serial data pin (SDA). For this, the SCL and SDA line of the sensor must be connected to the corresponding lines of the Arduino mega microcontroller, which in this case are the pins D21 and D20 respectively. When the clock line changes to high, one bit of data is transferred by the Arduino to the sensor over the SDA line. The sensor executes this request and transfers data back to the Arduino over the same SDA line. The SCL line allows data transfer only from the Arduino to the sensor; hence it is an input-only line, while the SDA line is bi-directional, which is why it is an I/O type line. In addition to the I2C lines, the sensor has one voltage pin (VDD), one ground pin (GND), one interrupt pin (INT), and a No Connect (NC) pin. The interrupt pin is set to low to turn on the onboard LED. Since the sensor is extremely sensitive, a lens shroud was designed to prevent ambient light from being detected by the sensor. Threshold values were defined to detect the red and blue colored tiles. The yellow tile was detected indirectly - if the red and blue threshold were not met, the detected tile would be read as yellow. Once a color is detected by the color sensor, it is set as a flag within the FSM of the robot.

User Inputs

- *LED Momentary Button Color Inputs*

Three colored momentary push buttons were used as inputs for selecting the desired color. Each push button has a built-in LED which was used to indicate the selected color. Each of the buttons had four connections – two connections were used for control logic while the other two connections were used to power the LED. For logic control, the first lead was connected directly to the 3.3V breadboard power line. The second logic control lead was connected to a pull-down resistor and to an input pin on the Arduino. The LED of each button was grounded and sent a 5V control logic signal from the Arduino. The complete LED button wiring, and all current-limiting resistor values, can be seen in Figure A.VII.1 and Figure A.VII.2 from Appendix VII. Figure 6 shows the three LED buttons mounted to the input shroud while Figure 7 shows manual operation of the LED buttons.

- *Capacitive Touch Sensor*

After the desired color has been selected using the color input button, the AT42QT1010 capacitive touch button was used to send the robot into game mode. The working principle is simple – the chip monitors the conductive area for touch. Once touch is detected, the chip sends a logical high to its output, otherwise the output is kept low. The four pins of the chip were LED, VDD, GND and OUT. VDD was the power supply for the button and was given 5V from the breadboard adapter. GND was the ground pin, and OUT was the output pin of the button. The LED pin controlled the onboard LED of the button and was soldered to the OUT pin. As a result, when a touch was detected, the onboard LED lit up and the robot was switched into game mode.

- *Emergency Stop Switch*

It was initially desired to use a momentary push button to terminate the Arduino FSM and to act as the emergency stop for the robot. Testing and calibration of the velocity and position PID controllers required the use of a latching emergency stop switch to cut power from the L298N motor driver. The momentary button seen in all CAD renders and wiring diagrams was replaced on the physical robot with a latching switch. Figure 1 shows the actual emergency stop switch that was used to cut the 12V power to the motors.

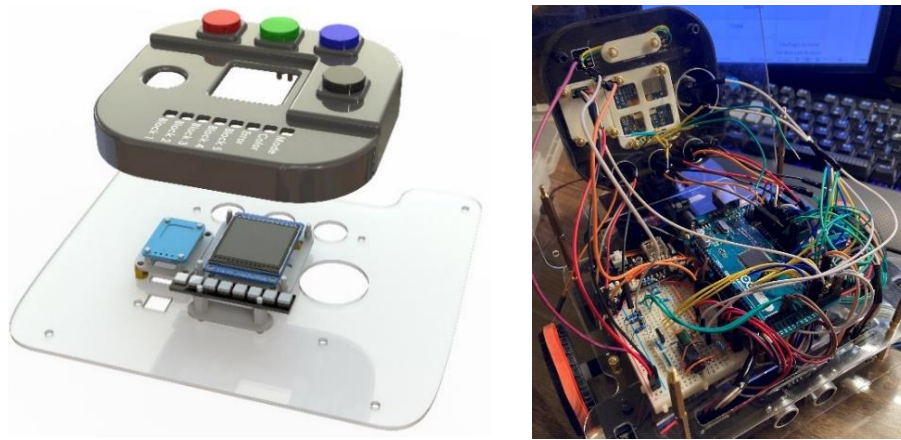


Figure 6: User inputs and outputs custom shroud (left) and wiring (right) – all robot inputs and display outputs were contained within a 3D-printed shroud. The momentary capacitive touch button, Neopixel RGBW LED stick, and 1.54in LCD screen were all mounted to the top baseplate using 3D-printed brackets while the LED buttons and emergency stop switch were mounted directly to the custom shroud

Outputs and Display

- *LED Momentary Button Color Outputs*

The three momentary push buttons that were used as a user input were also used as an output to display the selected color. Each button had an internal LED and when it was pressed, it lit up to display the selected color as shown in Figure 7. Since the same push buttons were used as both user input and output, no additional wiring was necessary. This LED remained lit throughout the game until the robot reached the final tile, or until the time ran out. The user was allowed to change the desired color while the robot was not in game mode. However, once the game started, color selection was not available anymore and pressing the buttons could not change the color of the LED. The order of operation for the LED momentary buttons can be seen in the completed FSM.

- *LCD Screen*

An Adafruit 1.54in TFT LCD display was used to fulfill the additional output functionality for the robot. The LCD screen utilizes 4-wire SPI communication to display BMP images and was powered from the 5V power line on the breadboard. An onboard micro-SD card was used to store the required BMP images that were displayed on the screen. The main function of the LCD screen was to map the colored tiles of the play arena as the robot completed the whack-a-mole game. Figure 7 shows manual operation of the LCD screen as the colored LED buttons were used as inputs to create an example arena layout. Figure 7 also shows the resolution of the screen

and the added functionality of displaying the Purdue ME logo. All required wiring connections for the LCD screen can be seen in Appendix VII, Figure A.VII.2.

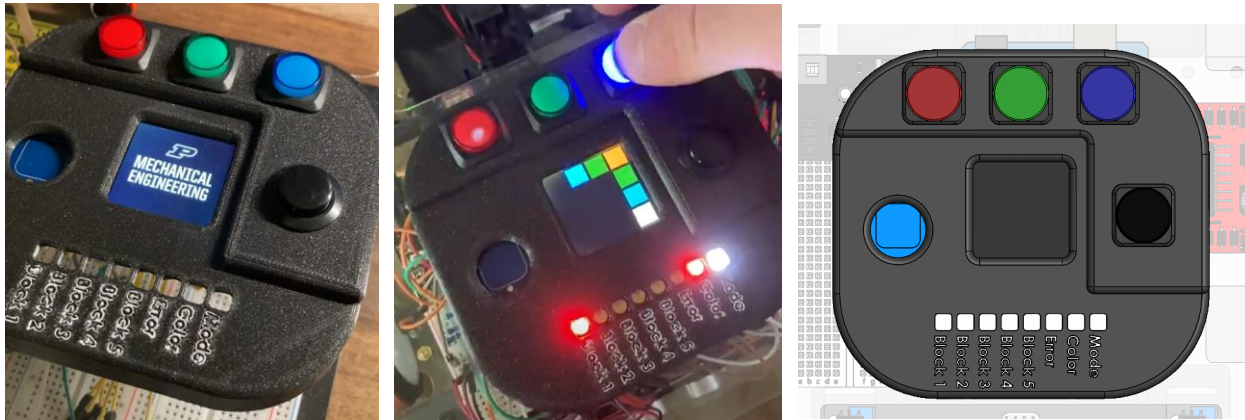


Figure 7: Performance of LCD screen, LED momentary push buttons, and RGBW NeoPixel Stick (left and middle) with CAD for shroud (right)– testing of all components was completed off the main robot chassis

- **RGBW LED Strip**

A NeoPixel stick with 8 LEDs was used to indicate color selection, game mode, and the number of blocks successfully dropped. When the desired color was selected, the LED strip displayed the color on the second LED. The first LED was used to indicate that the robot had successfully entered game mode. Each time the robot placed a block onto a tile, the LED strip would turn on an additional LED starting from the far left of the strip. The pixel color of each LED was also adjustable by red, green, and blue intensity values in the 0-255 range. This feature allowed the LED to show a variety of colors - such as yellow light which can be displayed by adjusting red and green brightness values.

The wiring of the NeoPixel strip only had three pins, 5V, GND, and DIN. The 5V pin was connected to the 5V of the breadboard adapter, GND was connected to the ground pin and DIN was connected to a digital pin of the Arduino. A resistor was added between the Arduino digital pin and the LED strip DIN pin to prevent LEDs from being damaged. The wiring diagram and selected resistor can be found in Appendix VII, Figure A.VII.2.

Results

After extensive testing of the mole whacker, it was concluded that the servo motor was working smoothly. However, several issues related to the 3D-printed pieces were discovered during the testing. The first issue being the linkage bars not connecting securely to the servo motor. Since there are no sensors to ensure that the cubes have been dropped, the linkage system must be rigidly attached to the motor to ensure that the rotation cycle of the servo is completed. The

calibration of the motion also depends on the placement of the linkage bars. To tackle this issue, a new linkage part was designed with a more rigid connection with the servo motor, and less degrees of freedom were given to the linkage system. Another issue observed during calibration of the mole whacker was the clearance of the pusher guide not being large enough. Since the 9g servo motor does not provide a large amount of torque, the pusher tended to get stuck. A new pusher was designed with a slightly lower height, and the pusher guide was also sanded down to decrease friction between the two components. The modifications proved to be effective, as the mole whacker worked reliably in all future testing and during the final robotic competition.

While the ultrasonic sensors can be seen in the CAD model for the robot and on the physical robot, the sensors do not serve any purpose in the current robot design. In the original design of the robot behavior, the ultrasonic sensors were implemented to detect the distances between the robot and the front and side walls and to identify the robot location. However, after testing the robot on the actual game map, it was determined that the ultrasonic sensors would not perform as expected; due to the inconsistent wall conditions, the ultrasonic sensors were unable to yield accurate results. On top of that, the motor encoders were enough to accurately locate the robot and ensure that the robot was successfully able to complete the whack-a-mole game. As a result, the ultrasonic sensors were not implemented in the final design of the robot, and they were left attached only in the rare chance that an error-correcting state was needed.

The initial design of the robot had two 65mm wheels and 3 idlers wheel - with two idler wheels in front of the drive wheels and one in the center back. After the robot was fully assembled, it was discovered that the CAD model for the idler wheels did not match the physical ones, resulting in an unstable contact between the drive wheels and the ground. As a temporary solution, rubber bands were initially stretched onto the wheels to increase the diameter of the drive wheels as well as to increase the clearance of the wheelbase. While the rubber bands proved to be an effective temporary solution, several issues were discovered during calibration testing; the rubber bands would constantly misalign the wheels and sometimes detach from the wheels, resulting in an unpredictable robot driving behavior. Additionally, it was noted that due to the insufficient wheel clearance, the front idler wheels and the IR line sensors would tend to interfere with bumps in the arena floor and cause the robot to deviate from its intended path. To solve all previously mentioned wheel clearance issues, the front two idler wheels were removed along with the rubber

bands. The IR sensors were also slightly bent backwards to create more clearance in the front. The new design was tested, and it displayed great ability to traverse over the map while maintaining its longitudinal balance and accurate readings.

Towards the end of the project, the team was actively preparing for the competition. While the robot was able to achieve all tasks listed on the checklist, it did not complete them in an efficient manner. The initial testing showed that the robot would take around 91 seconds to complete the whole map. Although it satisfied the requirement of 2 minutes, the team determined that it was insufficient for the competition. Therefore, the code was modified to remove all unnecessary delays that were included for color reading and “mole-whacking”. The motor speed was also increased incrementally to ensure the robot would not tip forward and have large overshoots when braking. After multiple modifications, the robot was able to complete the whack-a-mole game in 55 seconds.

The competition was the ultimate challenge for the robot. In other words, the competition provided the most accurate reference for the robot’s performance. The robot made it to the final round of the whack-a-mole tournament after three successful games. For all three games, the robot’s performance was consistent; the robot completed all three games in 55 seconds, and it was able to accurately drop all five cubes on the tiles of the desired colors. The mole whacker did not encounter any jamming issues during the games and was able to smoothly drop all the cubes. While the rearrangement of the game field resulted in large bumps on several of the tiles, the robot’s modified design was able to traverse across the tiles with ease while maintaining a forward motion and accurate turning. As the robot slowly stopped inside the middle of the starting white tile, it completed all the tasks flawlessly, putting a perfect end to the competition and to the project. A video showing the robot completing the whack-a-mole game in 55 seconds can be found using this link: <https://youtu.be/sy63tyc4v4k>.

Discussion

The robot successfully completed all the major functions outlined in the final check-off document. During the competition, the robot navigated the game field and met the required objectives. Two significant problems the team encountered while testing were not having the robot traveling on a straight path and not being able to make consistent 90° turns. This was due to a combination of factors including the two DC motors not being physically identical, wheel slippage

on the surface of the game field, and less charge on the battery which resulted in insufficient torque being given to the wheels. Attempting to make a 90° turn would result in significant overshoot or undershoot. To counter these issues, the team implemented a positional PID controller to ensure accurate 90° turns and a velocity PID controller to ensure that the robot drove in a straight path. The tuning of the PID constants was done through trial and error. The constants were initialized at zero, and each constant was incremented slowly to alleviate any jitters, disturbances, or overshoot.

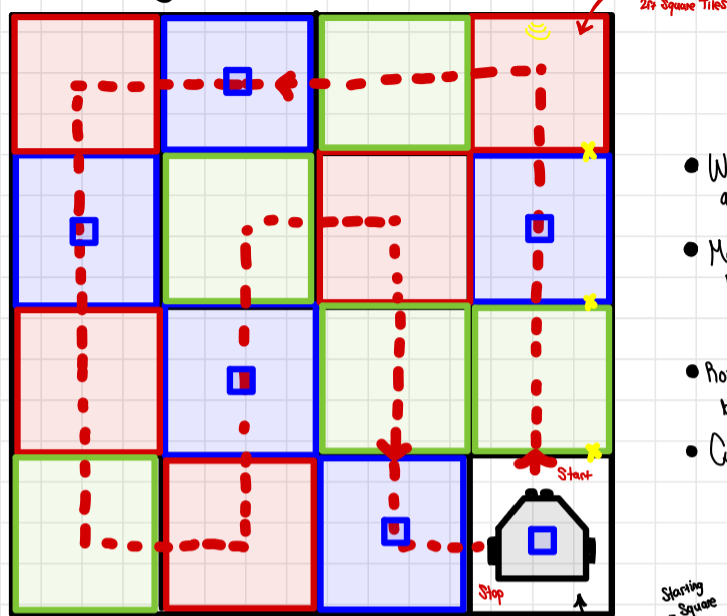
For future work, there is room for improvement in making the robot achieve its objectives faster. For example, currently, the robot comes to a complete stop on every square tile to detect its color. Going forward, the robot could be programmed to detect the color of the square tiles as it moves across each tile and comes to a stop once it detects the desired color. It might also be desired to include an error-correcting state. Currently, the robot moves on a predetermined path; if it is thrown off course, it will not find its path and complete the game. An error correcting state could implement gyroscopes and ultrasonic sensors to help the robot better locate itself and correct its path to finish the game if any extraneous disturbances occur on the path of the robot.

Conclusion

The goal of the project was to apply all previously acquired mechatronics knowledge towards the development of an autonomous robot to complete a game of whack-a-mole. Given a set of engineering requirements and constraints to limit the overall size and cost of the robot, as well as a set of game rules to guide the behavior of autonomously driving the robot, the team worked to design and implement a complex mechatronics system. The mechanical design of the robot was initially developed to facilitate the movement of the robot, the actuation of the mole whacker, and the containment of all electronic components. The electronic design of the robot was conducted in a parallel manner with the mechanical design as components were carefully selected based on compatible communication protocols, operation, and power requirements. The software design of the robot followed the mechanical and electrical work, and it took up a disproportionate amount of project time to develop and to tune. At the completion of the project, Team 13's completely original robot was successfully and accurately able to complete the whack-a-mole game in approximately 55 seconds and won second place in the course-wide robotics competition.

Appendix I – Design Ideation Diagrams

Sensing Path:



- Want to limit number of "moves" and turns that the robot has to make
- Maybe want to use the ultrasonic as much as possible
- Robot's route needs to route it back to white square in 2 minutes
- Current path assumes sensing capabilities are sound and work well

Sensing Scheme:

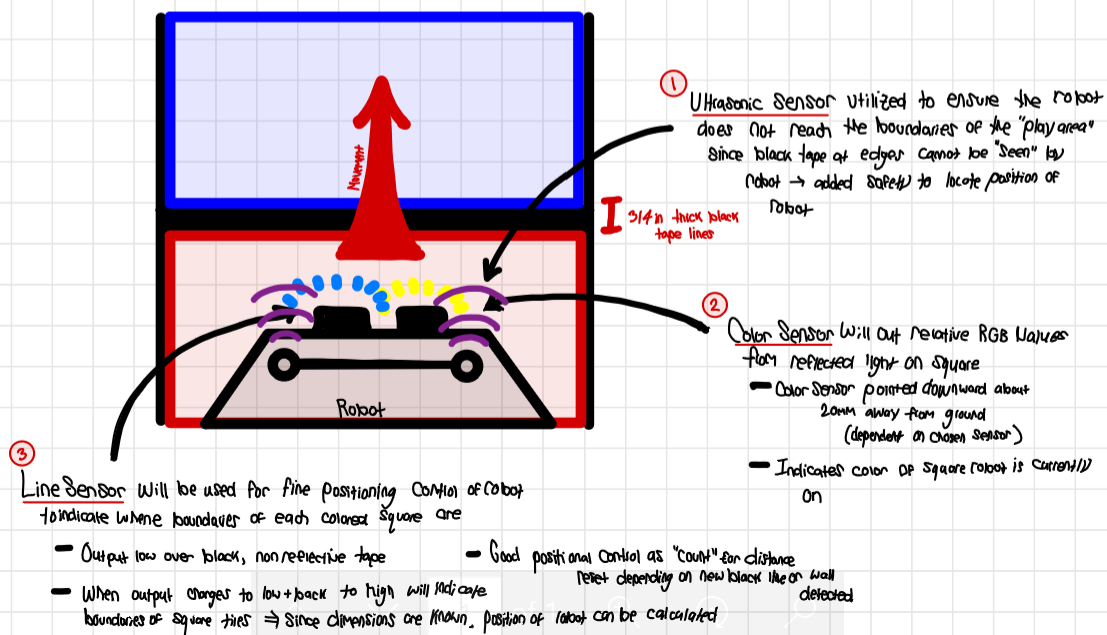


Figure A.I.1: Predetermined robot path and sensing scheme

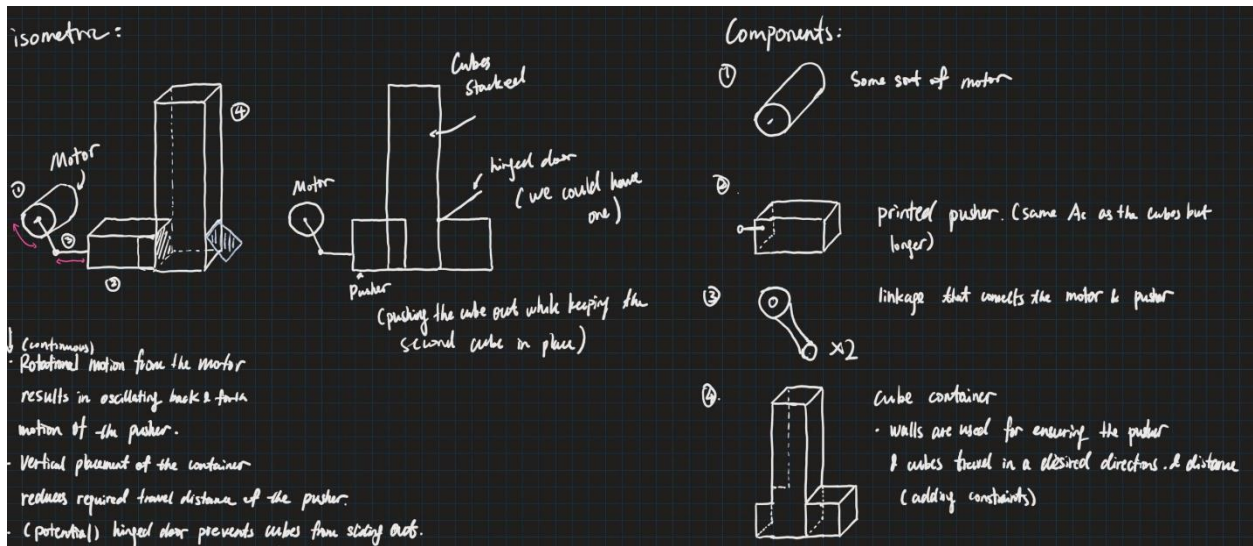


Figure A.1.2: Ideation for Mole Whacker Design - specific parts were listed, and the design remained unchanged throughout the duration of the project

Appendix II – FSM State Transition Diagram

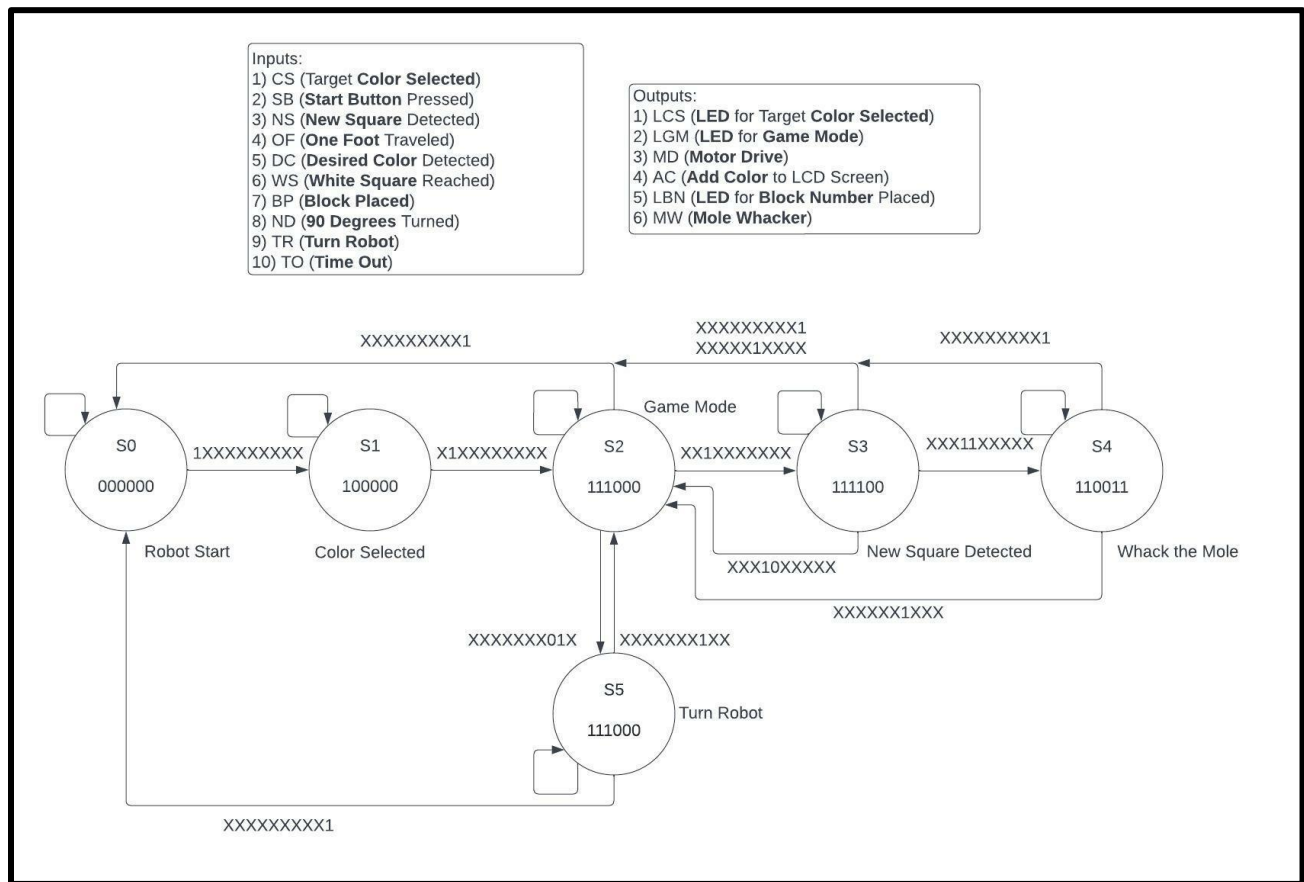


Figure A.II.1 FSM State Transition Diagram

Appendix III – Arduino Code Examples

```
634 // Mole whacking function
635 void whack_mole() {
636     int angle = 0;                                // Servo motor angle
637
638     for(angle = 0; angle < 180; angle += 5) {      // Move piston back
639         mole_whacker.write(angle);
640         delay(5);
641     }
642     delay(300);
643
644     for(angle = 180; angle > 0; angle -= 5) {      // Move piston forward
645         mole_whacker.write(angle);
646         delay(5);
647     }
648     delay(300);
649 }
```

Figure A.III.1 Sample Code for Mole Whacking Functionality

```
278 // Evaluate the control signal
279 if (state == S2 || state == S3 || state == S4) { // For velocity control
280     Kp[k] = 10;
281     Ki[k] = 1.85;
282     Kd[k] = 0;
283     e[k] = targetVel[k] - vfilt[k];
284 } else if (state == S5) { // For position control
285     Kp[k] = 3.75;
286     Ki[k] = 0.5;
287     Kd[k] = 0.4;
288     e[k] = enc[k] - pos[k];
289 }
290 e_int[k] = e_int[k] + (e[k] * deltaT);
291 e_der[k] = (e[k] - e_prev[k])/deltaT;
292 e_prev[k] = e[k];
293 motor_pwm[k] = (Kp[k] * e[k]) + (Ki[k] * e_int[k]) + (Kd[k] * e_der[k]);
294
295 // Signal the motor
296 setMotor(motor_pwm[k], pwm[k], in1[k], in2[k]);
```

Figure A.III.2 Sample Code for Velocity & Position PID Control – error calculation variables were reused between the two controllers when it was desired to switch between driving and turning

Appendix IV – Arduino Mega Pin Designation

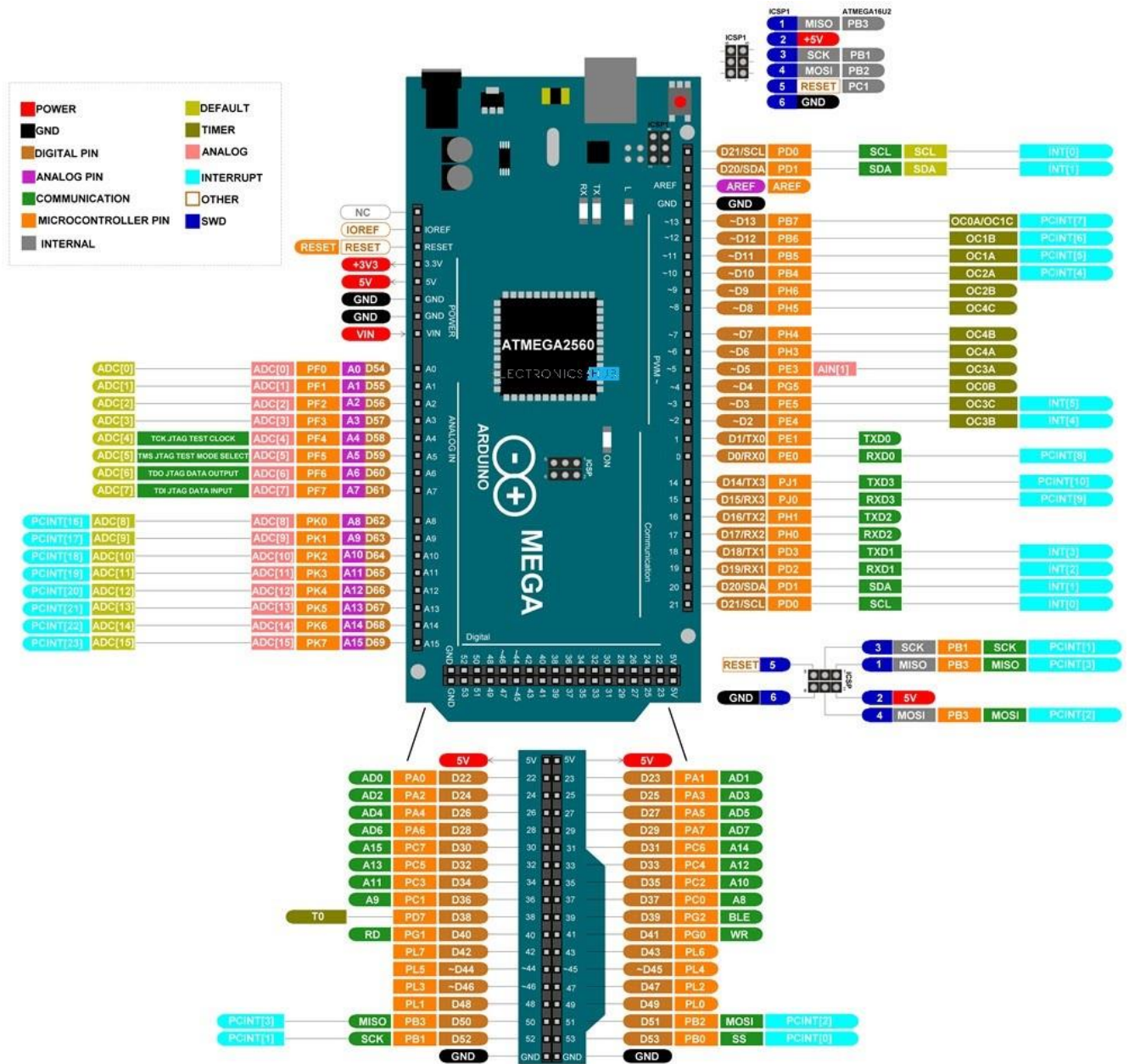







Figure A.IV.1: Arduino Mega Pin Layout




Pin #	Pin Name	Additional Pin Function	Pin Usage
5	GND		Common Ground
26	D53		CCS Signal for LCD Screen
27	D52		SCK Signal for LCD Screen
28	D51		SI Signal for LCD Screen
29	D50		SO Signal for LCD Screen
30	D49		TCS Signal for LCD Screen
32	D47		RST Signal for LCD Screen
34	~D45		DC Signal for LCD Screen
36	D43		Vin/Logic Control for Red LED Button
38	D41		Vin/Logic Control for Green LED Button
40	D39		Vin/Logic Control for Blue LED Button
42	D37		ENB Signal for IR Left Line Sensor
43	D36		Input 3 (L298N)
44	D35		ENB Signal for IR Right Line Sensor
45	D34		Input 4 (L298N)
46	D33		Capacitive Touch Input
47	D32		Input 1 (L298N)
48	D31		RGB LED Stick Output
49	D30		Input 2 (L298N)
51	D28		Blue LED Button Input
53	D26		Green LED Button Input
54	D25		IR Line Sensor Left Input
55	D24		Red LED Button Input
56	D23		IR Line Sensor Right Input
60	D21/SCL	SCL	SCL Pin for RGB Sensor
61	D20/SDA	SDA	SDA Pin for RGB Sensor
62	D19/RX1	RXD1	Left Motor Hall A Interrupt
63	D18/TX1	TXD1	Right Motor Hall B Interrupt
70	~D2	PWM	Hall B Left Motor
71	~D3	PWM	Hall A Right Motor
72	~D4	PWM	ENA for L298N
73	~D5	PWM	ENB for L298N
76	~D8	PWM	ECHO Pin Side Ultrasonic
77	~D9	PWM	TRIG Pin Side Ultrasonic
78	~D10	PWM	Output signal of servomotor for mole whacker
79	~D11	PWM	ECHO Pin Front Ultrasonic
80	~D12	PWM	TRIG Pin Front Ultrasonic

Figure A.IV.2: Pin Assignments for the Robot

Appendix V – Robot Components and Sensor Selection Sheet

RGB Sensors													
Purchase?	Sensor Name	Communication Protocol	Analog or Digital?	LEDs for Illumination?	Lens/Light-shield?	Operating Distance	Input Voltage	Online Retailer	Unit Cost	Unit Cost	Purchase Link	Purchase Link	Notes
No	TCS320/TCS3200	TTL	Variable Frequency	Yes	No		2.7-5.5V	Mouser Electronics, Amazon	\$ 7.90	\$ 8.99	com/ProductDetail/RoboBot/SEN010-TCS300-PhotoPosition-Module-Sep8070558?qs=Ah3D%3D&mgch=1&gclid=CwKCA9hV-17898&creative=9325	https://www.youtube.com/watch?v=MwDANEtIPY8ab_ch	*Tutorial found online to calibrate sensor: *Widely available on Amazon with a shroud that can be easily printed
No	ISL29125	I2C	Digital	No	No		3.3V	Digi-Key	\$ 8.50		sealectronics/SEN-17829/66737647cm	https://www.youtube.com/watch?v=MwDANEtIPY8ab_ch	Tutorial found online to calibrate sensor: *Widely available on Amazon with a shroud that can be easily printed
Yes	TCS34725	I2C	Digital	Yes	No		2.7-3.6V	Digi-Key, Adafruit	\$ 7.95	\$ 7.95	https://www.adafruit.com/product/1334	https://www.adafruit.com/product/1334	Used in Lab 6
No	Flora TCS34725	I2C	Digital	Yes	No		2.7-3.6V	Adafruit	\$ 7.95		https://www.adafruit.com/product/1356		Same as above but in a circular format
Maybe	AS7262	I2C	Digital	Yes	Yes		3.3V	Adafruit	\$ 19.95		https://www.adafruit.com/product/3779		*As opposed to most sensors, this sensor has 6 color bins *Onboard voltage regulator for 3.3-5V (power with 5V) *Powerful onboard LED and Shroud for Sensor



IR Line Sensors															
Purchase?	Sensor Name	Communication Protocols	Analog or Digital?	LED for Illumination?	Lens/Light-shield?	Operating Range	Input Voltage	Online Retailer	Unit Cost	Number of Units	Purchase Link	Purchase Link	Notes		
No	DAKQI IR Tracking Sensor Module TCR5000	HIGH/LOW	Digital	NA	NA	0.2-15mm 2.5mm optimal	3.3-5V	Amazon	\$ 5.59	X5	com/DAKQI-IR-Tracking-Sensor-Module-TCR5000-Reflective-Photoelectric/dp/B00KT09GQ1/ref=sr_1_10		*Multi-turn potentiometer to adjust sensitivity of each IR sensor *Typical IR sensor on Amazon and comes in many variations with included wiring		
No	Parallax QTI Sensor	HIGH/LOW or Varying Shades of Gray	Digital or Analog	NA	NA	NA	5V	Digi-Key	\$ 9.30	X1	https://www.digikey.com/en/products/detail/parallax-inc/255-17409/1774455				
Yes	Infrared Barrier Sensor - RobotDyn	HIGH/LOW	Digital	NA	NA	NA	3.3-5V	Owned	\$ 3.99	X2	com/RobotDyn-Infrared-Obstacle-Avoidance-Barrier/dp/B0738933-5Dc	https://www.robotdyn.com/ProductsDetail/Infrared-Obstacle-Avoidance-Barrier/dp/B0738933-5Dc	*Provided by TA's from in-lab equipment		

Input/Output Devices												
Purchase?	Component Name	Input/Output Type	Communication Protocols	Input Voltage	Online Retailer	Unit Cost	Cost/Number of Units	Purchase Link	Purchase Link	Purchase Link	Purchase Link	Notes
No	Adafruit Round Rectangle Color TFT Display	Other Output	SPI	3.3-5V	Adafruit	\$ 17.50	\$ 7.95	https://www.adafruit.com/product/7393	https://www.adafruit.com/product/7393			*1.47" Diagonal TFT full-color display *Requires separate SD Card if images are to be stored *Display minimap of robot position on playfield? *Should be easy to use with Arduino library
Yes	Illuminated Momentary Button	Color Input	Digital	3V	Adafruit	\$ 1.95	X3	https://www.adafruit.com/product/1477	https://www.adafruit.com/product/1446	https://www.adafruit.com/product/1438		*16mm diameter *LEDs built-in *Available as Color Output *Available as Emergency Stop Output
No	SparkFun DEV-15083 RGB Rotary Encoder	Color Input	I2C	3.3V	Digi-Key, Mouser Electronics	\$ 22.95	\$ 19.95	https://www.digikey.com/en/products/detail/sealectronics/DEV-15083/174681246	https://www.mouser.com/ProductDetail/sealectronics/DEV-15083/174681246?qs=Ah3D%3D&mgch=1&gclid=CwKCA9hV-17898&creative=9325			*24 ticks per rotation *RGB LED controlled via PWM *Alternatively could use a standard encoder with a color wheel attached *Available as Color Output
Yes	NeoPixel Stick - Cool White	Color State Output	I2C	5V	Adafruit	\$ 7.95		https://www.adafruit.com/product/2869#tech-spec-details				*Since 4 cubes will be provided, two lights will turn on per dropped block *Color can be changed depending on color selection
No	Mini LED Matrix	Other Output	I2C	5V	Adafruit	\$ 11.95		https://www.adafruit.com/product/959#technical-details				*Can act as a 7-segment display to display the number of dropped cubes *Could also display position of robot on small grid? *Comes in multiple colors
No	Metal Ball Tactile Button	Round Start Input	Digital	Rated to 12V or 50mA	Adafruit	\$ 5.95	X10	https://www.adafruit.com/product/7347				*Higher quality button than a standard button *Pack of 10 buttons
No	STEMMA Wired Tactile Push Button	Round Start Input	Digital	3-5V	Adafruit	\$ 7.50	X5	https://www.adafruit.com/product/4431				*Comes with 5 multicolor buttons - black button used to start round *Long wire harness included *Available as Color Input
Yes	Panel Mount Momentary Pushbutton-Black	Round Start Input	Digital	3-5V	Adafruit	\$ 0.95		https://www.adafruit.com/product/1305				*Very simple cheap button *16mm diameter
No	Illuminated Toggle Switch With Red Cover	Emergency Stop Input	Digital	Rated to 3-12V or 2A	Adafruit	\$ 3.95		https://www.adafruit.com/product/3318#tech-spec-details				*Toggle switch with light on end to indicate state of switch *Somewhat large *Cover is removable
Yes	Standalone Momentary Capacitive Touch Sensor - AT42QT1012	Round Start Input	Digital	2-5V	Adafruit	\$ 5.95		https://www.adafruit.com/product/1374				*Pin goes high or low depending on touch *Available as Emergency Stop Input (if toggle used)
Yes	Adafruit 1.54" Wide Angle TFT LCD Display	Other Output	SPI	3.3-5V	Adafruit	\$ 24.95		https://www.adafruit.com/product/7387				*1.5" Diagonal TFT full-color display *Requires separate SD Card if images are to be stored *Display minimap of robot position on playfield? *Should be easy to use with Arduino library
No	Massive Arcade Button with LED - 300mm Red	Emergency Stop Input	Digital	5V-12V	Adafruit	\$ 9.95		adafruit				



Appendix VI – Completed Component Budget and Sourcing Plan

Completed Orders Fulfilled by Department									
Item Description	How will the item be used for the project?	Sourcing Plan	Vendor	Unit Cost	Quantity	Shipping Cost	Total Item Cost	Purchase Link	Estimated Purchase date
RGB Color Sensor with IR filter and White LED - TCS34725	Sense current square color	Purchase	Adafruit	\$ 7.95	1	\$ 13.23	\$21.18	https://www.adafruit.com/product/1334	6-Mar-2022
16mm Illuminated Pushbutton - BLUE (momentary)	Double as both BLUE color input and output	Purchase	Adafruit	\$ 1.95	1	\$ -	\$1.95	https://www.adafruit.com/product/4373	6-Mar-2022
16mm Illuminated Pushbutton - GREEN (momentary)	Double as both GREEN color input and output	Purchase	Adafruit	\$ 1.50	1	\$ -	\$1.50	https://www.adafruit.com/product/1345	6-Mar-2022
16mm Illuminated Pushbutton - RED (momentary)	Double as both RED color input and output	Purchase	Adafruit	\$ 1.50	1	\$ -	\$1.50	https://www.adafruit.com/product/1439	6-Mar-2022
NeoPixel Stick - 8x 5050 RGBW LEDs - Cool White	1 LED for Target Color output, 1 LED for Game Mode State output, 5 LEDs for Placed Cubes output	Purchase	Adafruit	\$ 7.95	1	\$ -	\$7.95	https://www.adafruit.com/product/2968	6-Mar-2022
Momentary Capacitive Touch Sensor - AT42QT1010	Digital momentary input button to indicate round start input	Purchase	Adafruit	\$ 5.95	1	\$ -	\$5.95	https://www.adafruit.com/product/1374	6-Mar-2022
16mm Panel Mount Momentary Pushbutton - Black	Digital momentary input button to indicate round start input (backup)	Purchase	Adafruit	\$ 0.95	1	\$ -	\$0.95	https://www.adafruit.com/product/473	6-Mar-2022
30mm Arcade Button - Red	Emergency Stop	Purchase	Adafruit	\$ 5.95	1	\$ -	\$5.95	https://www.adafruit.com/product/473	6-Mar-2022
Adafruit 1.54" 240x240 Wide Angle TFT LCD Display with MicroSD - ST7789	Extra output device for mapping tile layout of arena and indicate placed cubes	Purchase	Adafruit	\$ 24.95	1	\$ -	\$24.95	https://www.adafruit.com/product/3787	6-Mar-2022
8GB Blank SD/MicroSD Memory Card	Needed with LCD screen to load full color bitmaps	Purchase	Adafruit	\$ 9.95	1	\$ -	\$9.95	https://www.adafruit.com/product/1294	6-Mar-2022
2-Way Barrel Jack Splitter	Splits power from portable battery to breadboard and wheelbase	Purchase	Adafruit	\$ 2.95	1	\$ -	\$2.95	https://www.adafruit.com/product/1302	6-Mar-2022
Half-Size Breadboard with Mounting Holes	Needed for powering sensor components and various connections	Purchase	Adafruit	\$ 5.00	1	\$ -	\$5.00	https://www.adafruit.com/product/4530	6-Mar-2022
9V Battery Clip with 5.5mm/2.1mm Plug	Power Arduino Mega separately	Purchase	Adafruit	\$ 2.95	1	\$ -	\$2.95	https://www.adafruit.com/product/80	6-Mar-2022
Wheels	To Drive	Purchase	Adafruit	\$ 1.50	2	\$ -	\$3.00	https://www.adafruit.com/product/3763	6-Mar-2022
TOTAL:							\$95.73		

Completed Orders Fulfilled Independently									
Item Description	How will the item be used for the project?	Sourcing Plan	Vendor	Item Cost	Quantity	Shipping Cost	Total Item Cost	Purchase Link	Purchase date
240 Piece M3 Male Female Hex Brass Spacer Standoff Screw Nut Threaded Pillars	Used to assemble the main chassis of the robot to provide space for all electronic components	Purchase	Amazon	\$ 14.99	1	\$ -	\$14.99	https://www.amazon.com/gp/product/B000000000	6-Mar-2022
20 Piece M3 x 30mm Male to Female Thread Brass Hex Standoff Spacer Pillars	Used to assemble the main chassis of the robot to provide space for all electronic components	Purchase	Amazon	\$ 8.99	1	\$ -	\$8.99	https://www.amazon.com/gp/product/B000000000	6-Mar-2022
Rechargeable 9V Lithium Batteries, 5400mWh USB (2 Pack)	Used to power the Arduino Mega	Purchase	Amazon	\$ 14.39	1	\$ -	\$14.39	https://www.amazon.com/gp/product/B000000000	6-Mar-2022
120 Piece Breadboard Jumper Wires 10cm 15cm 20cm 30cm 40cm 50cm 100cm Wire Length	To connect all electronic components	Purchase	Amazon	\$ 6.89	1	\$ -	\$6.89	https://www.amazon.com/gp/product/B000000000	6-Mar-2022
Acrylic Plastic Sheet (1/8" Thick)	Laser-cut using ME machine shop laser cutter to form the three layered main chassis of the robot	Modify	Home Depot	\$ 20.00	1	\$ -	\$ 20.00		6-Mar-2022
Electrical Tape	To protect soldered connections on robot	Modify	CVS	\$ 7.00	1	\$ -	\$ 7.00		6-Mar-2022
TOTAL:							\$72.26		
GRAND TOTAL:							\$167.99		

Figure A.VI.1: Final Budget for Team 13's Robot – component orders fulfilled by the ME department totaled around \$95 while component orders fulfilled independently totaled around \$73. Overall, the total cost of the robot was around \$168

Appendix VII – Subassembly Wiring Diagrams

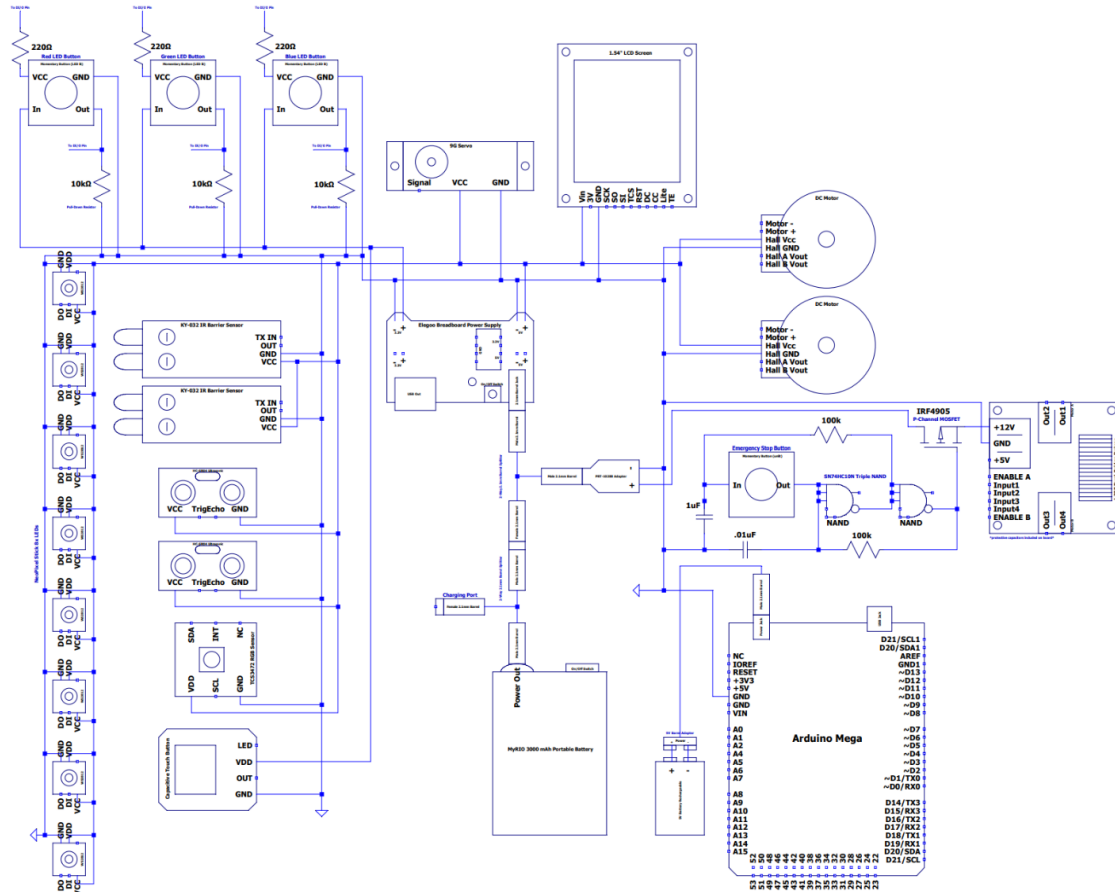


Figure A.VII.1: Power Distribution Subsystem wiring diagram (created in LTSpice)

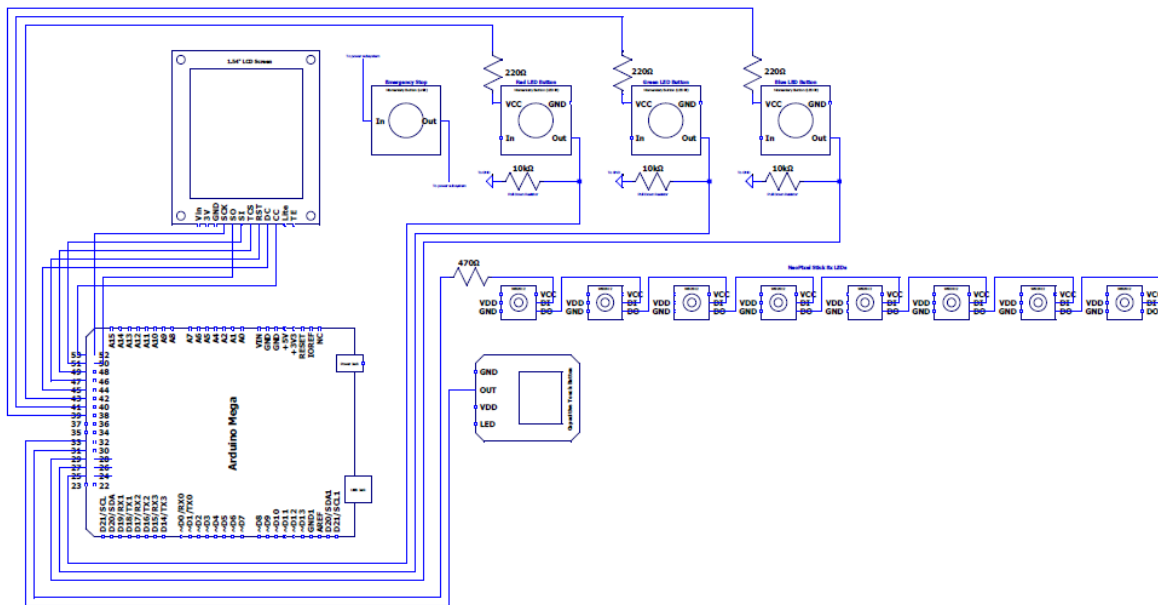


Figure A.VII.2: User Inputs and Output Subsystem wiring diagram (created in LTSpice)

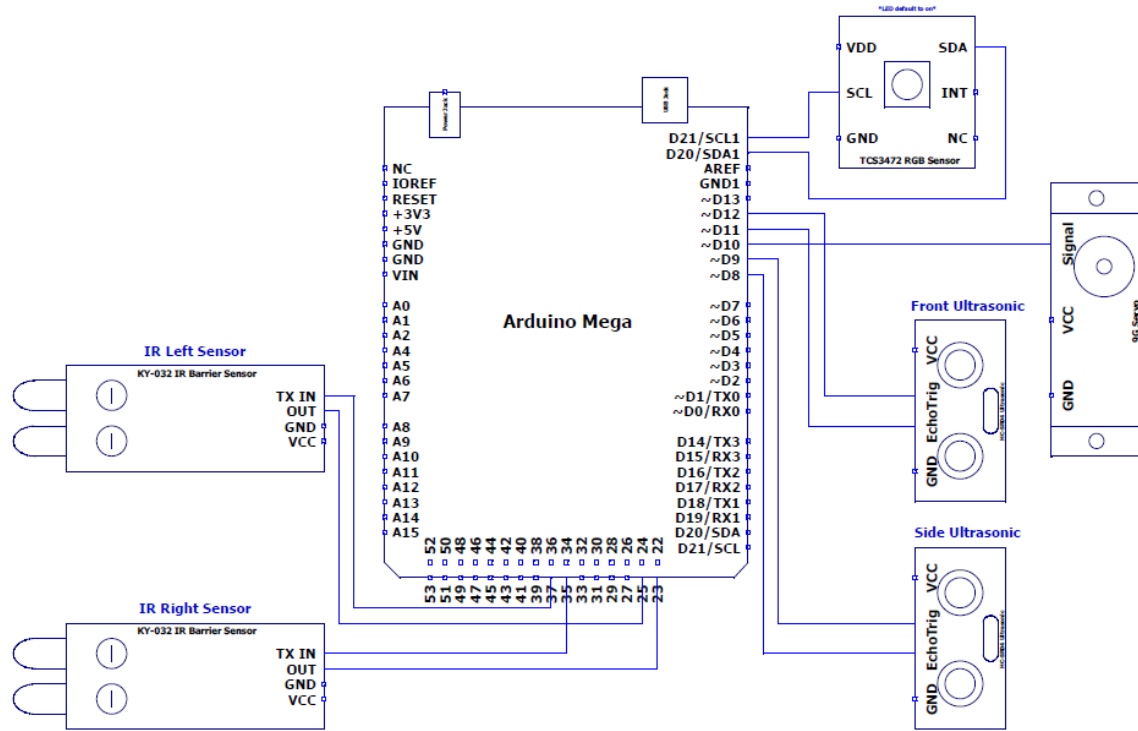


Figure A.VII.3: Mole Distribution Subsystem wiring diagram (created in LTSpice)

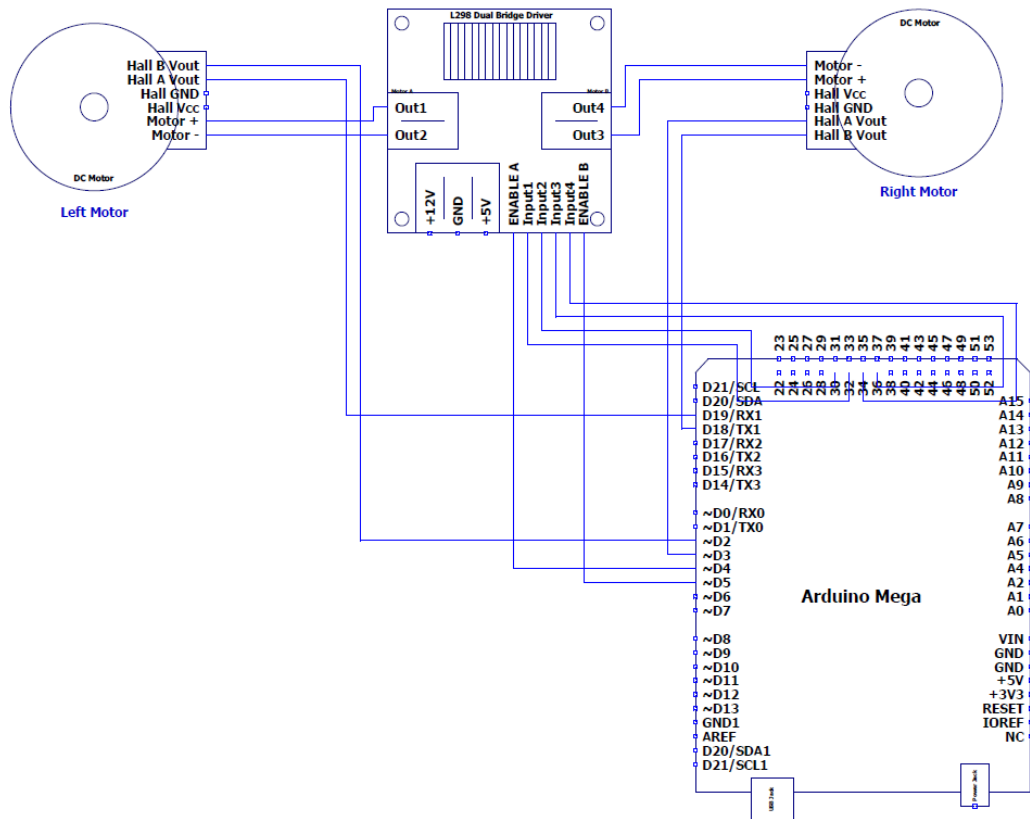
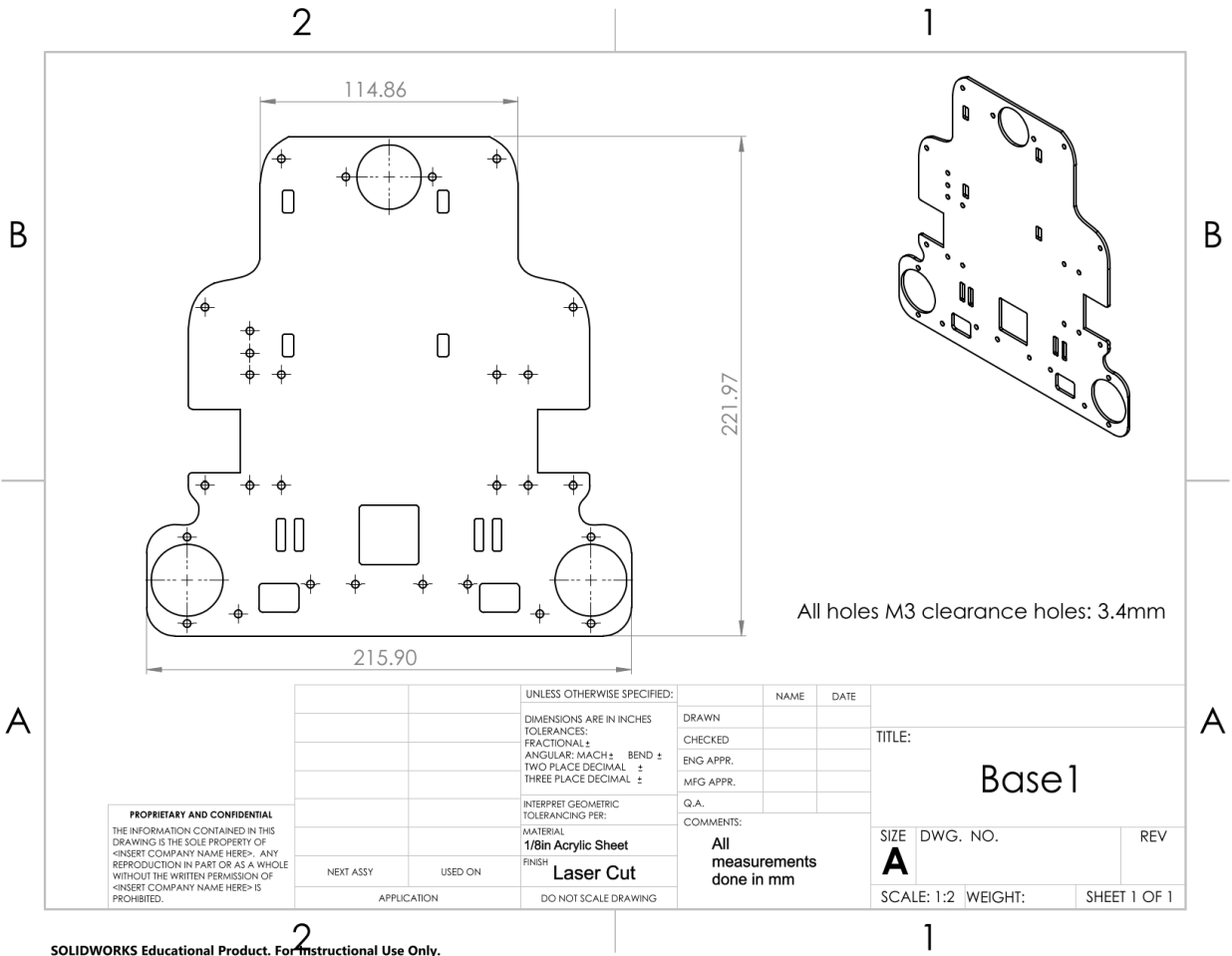
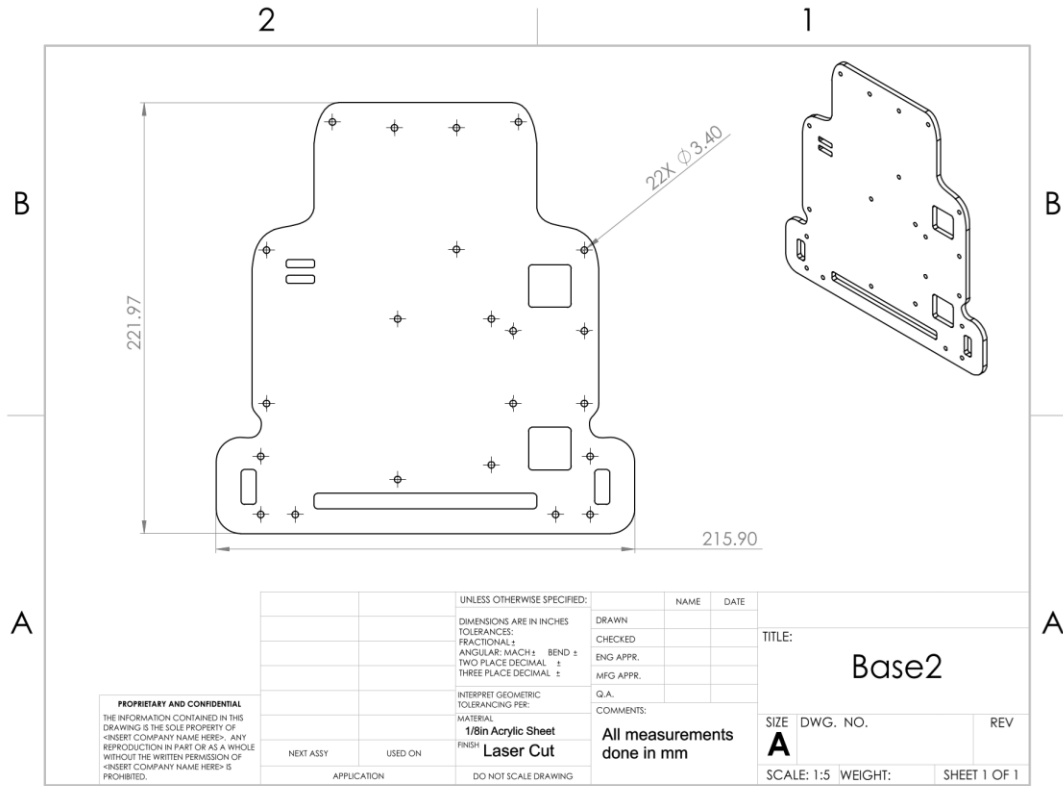


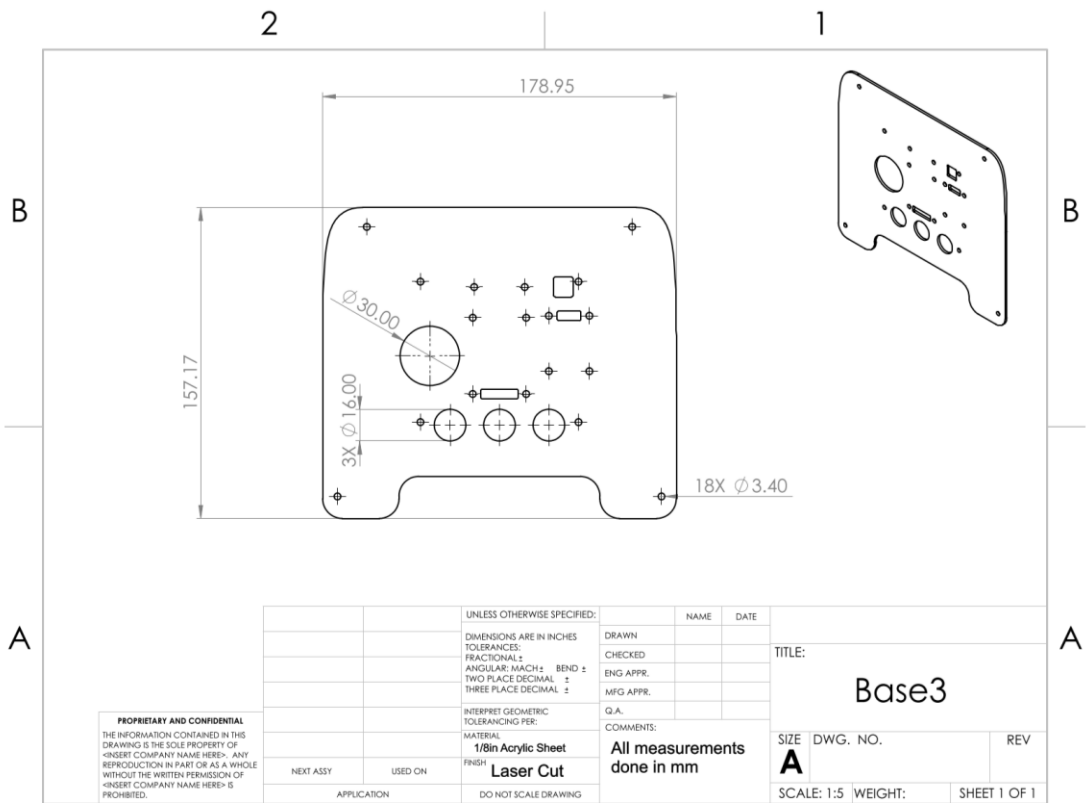
Figure A.VII.4: Drive Mechanism Subsystem wiring diagram (created in LTSpice)

Appendix VIII – Manufactured Part Drawings

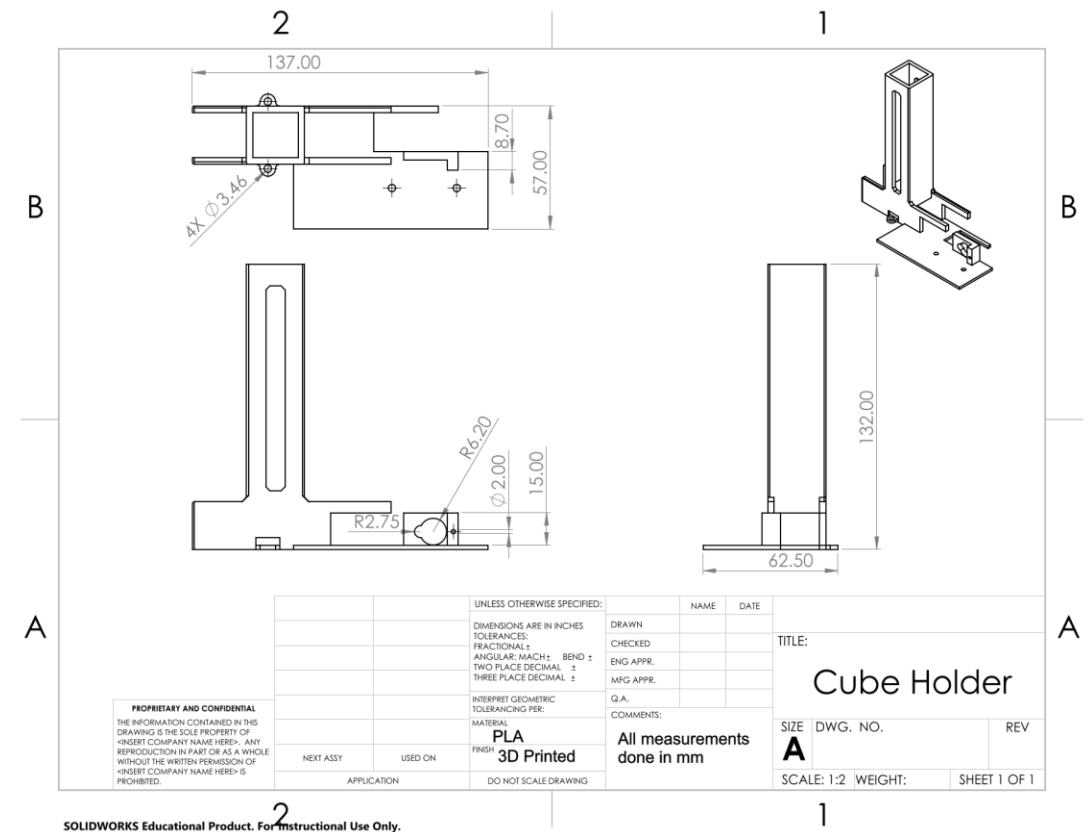
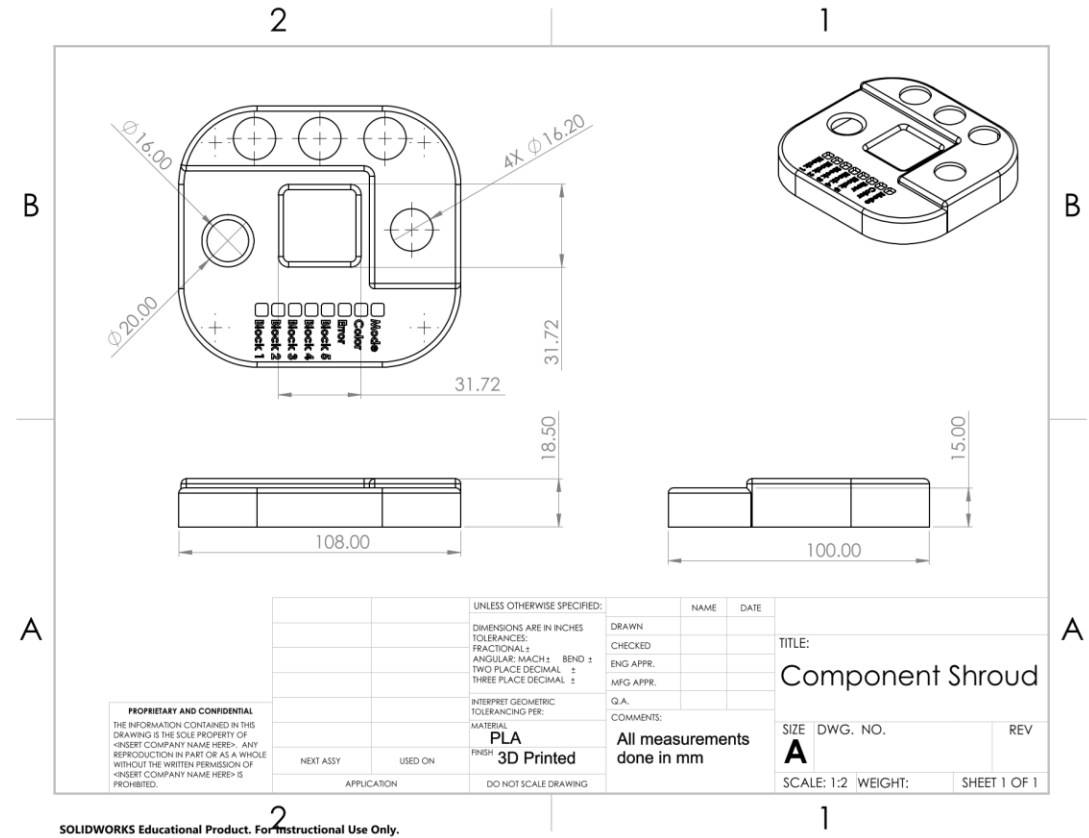


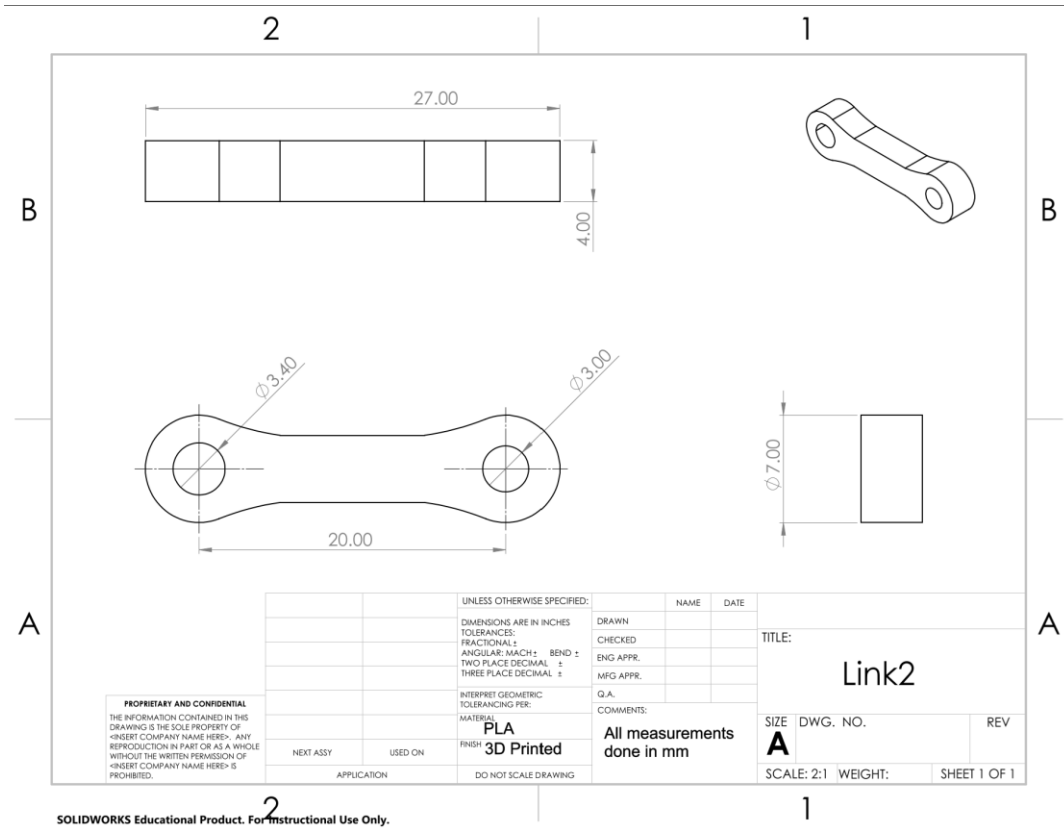
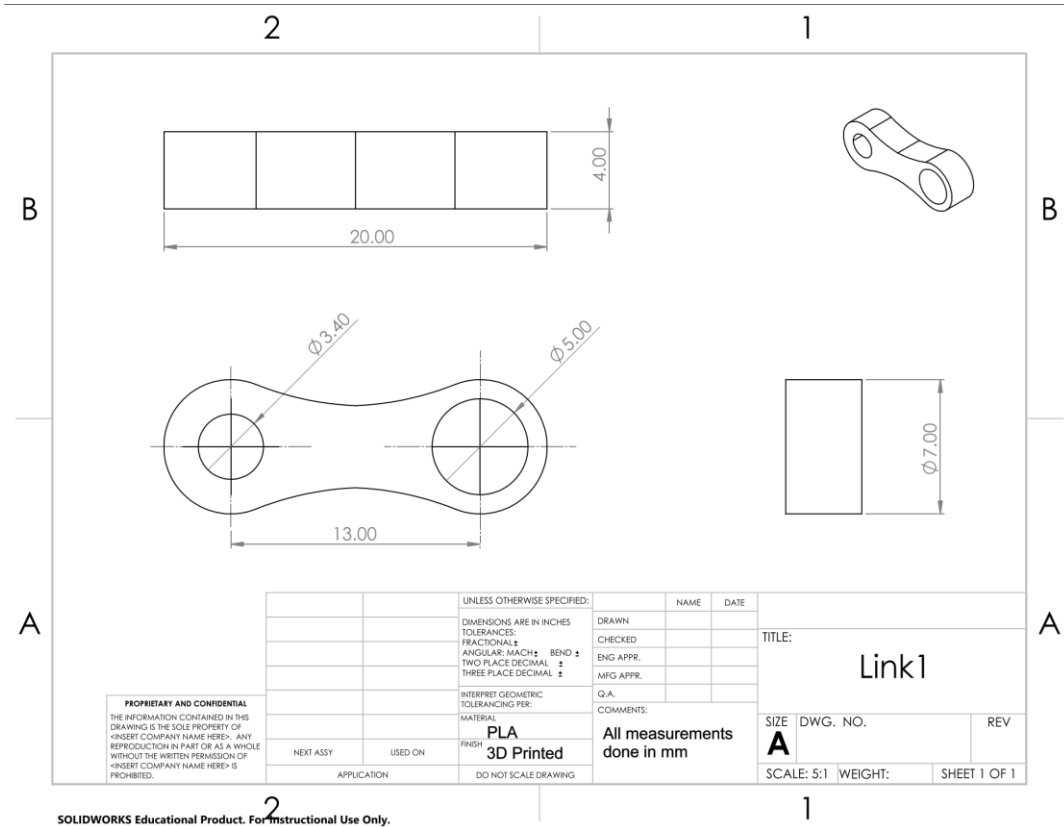


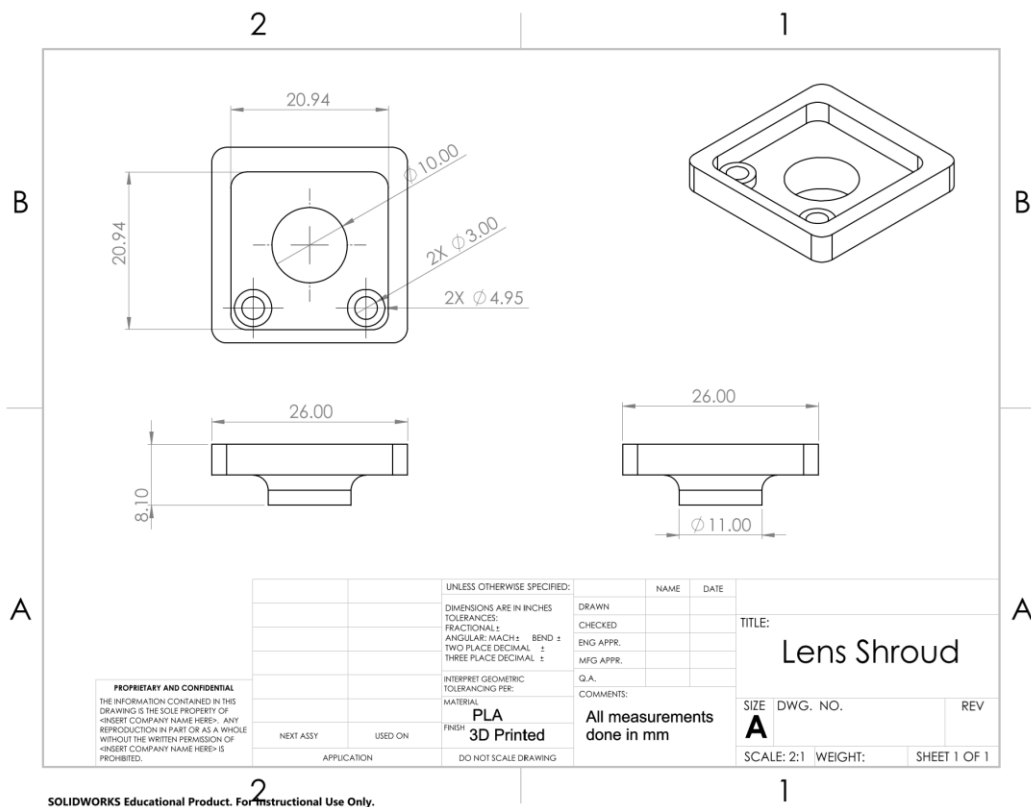
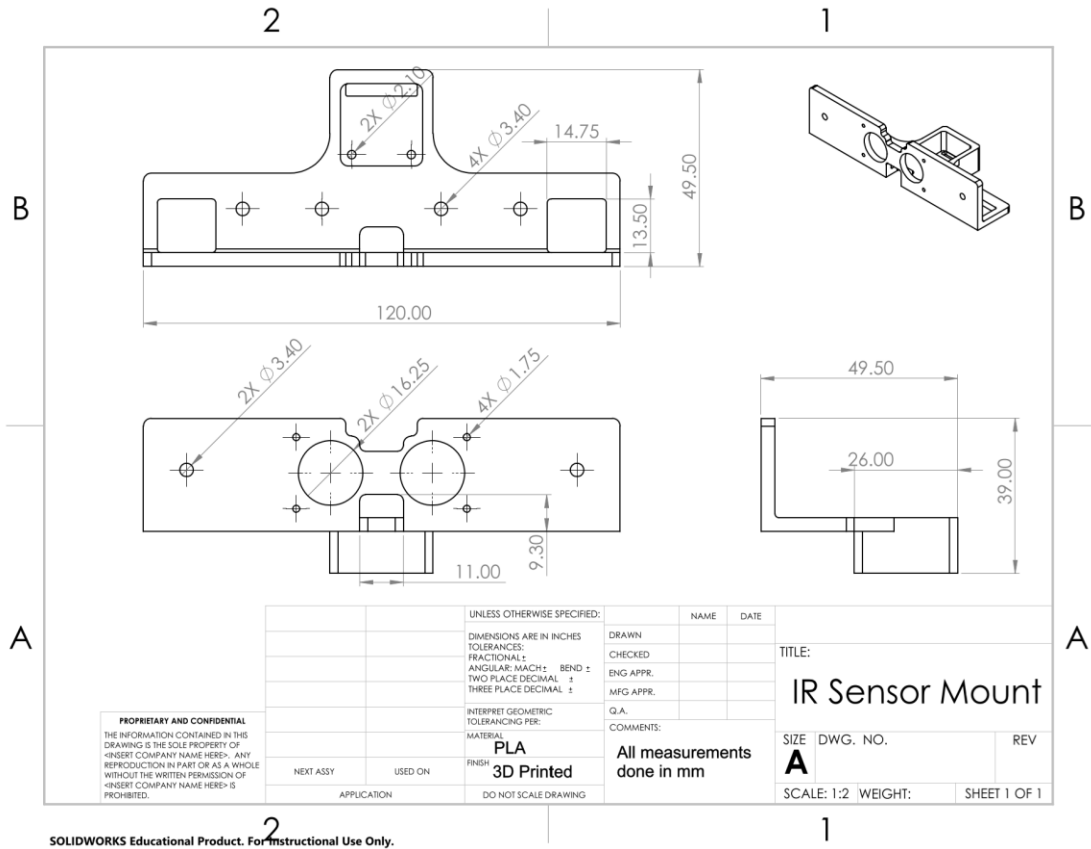
SOLIDWORKS Educational Product. For Instructional Use Only.

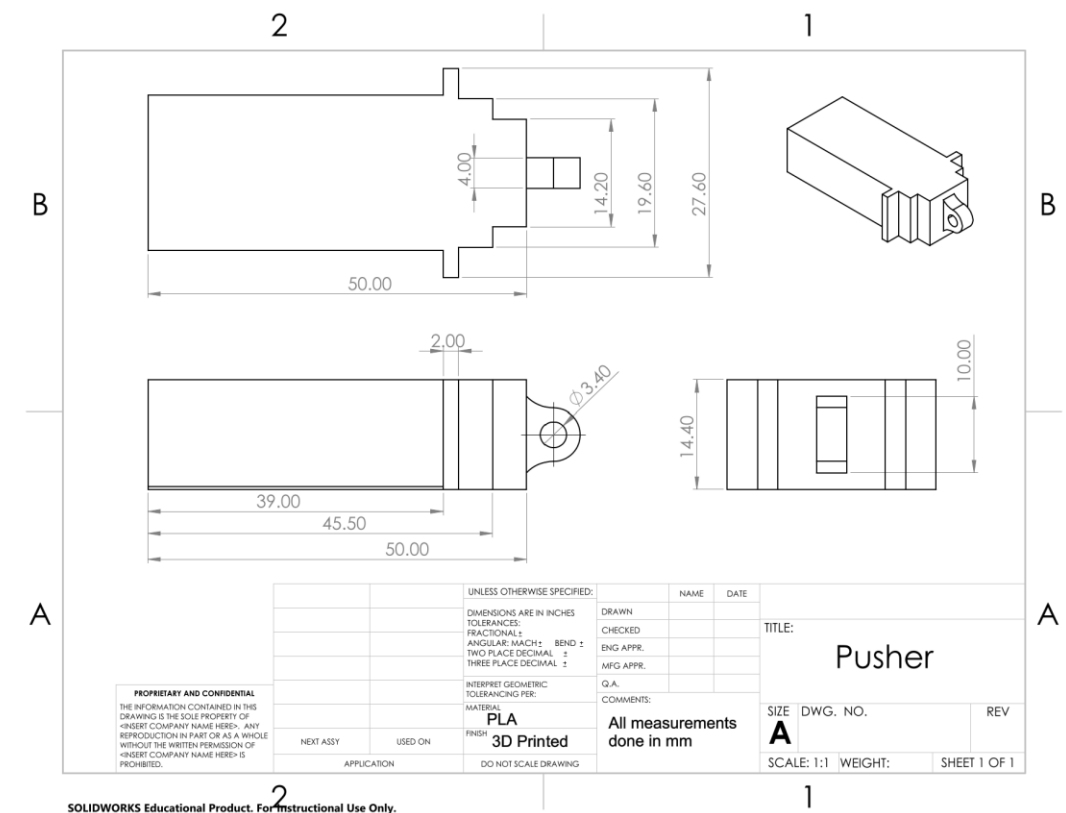
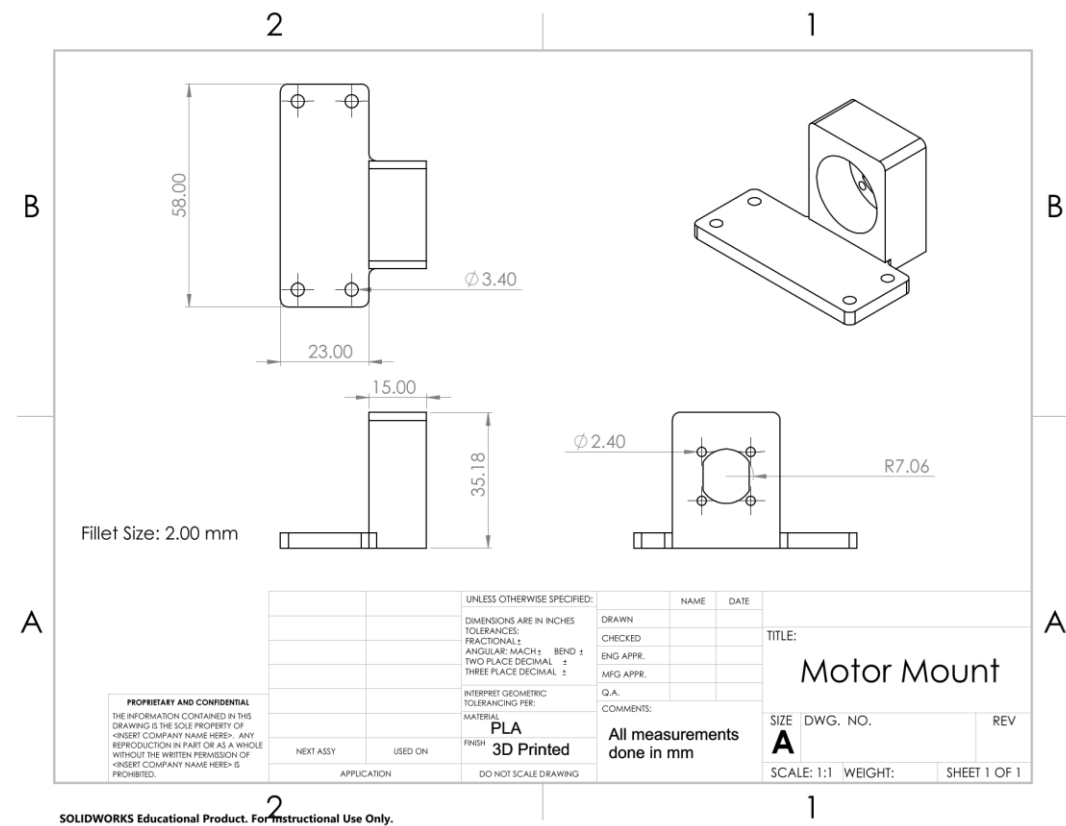


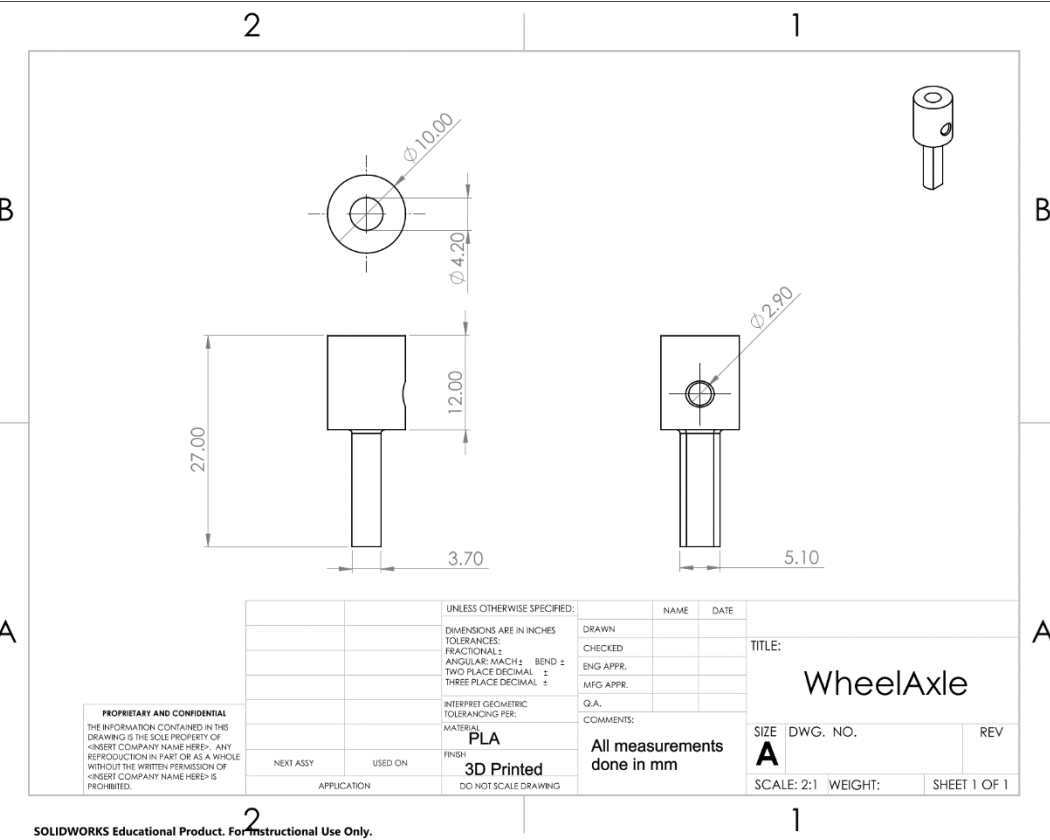
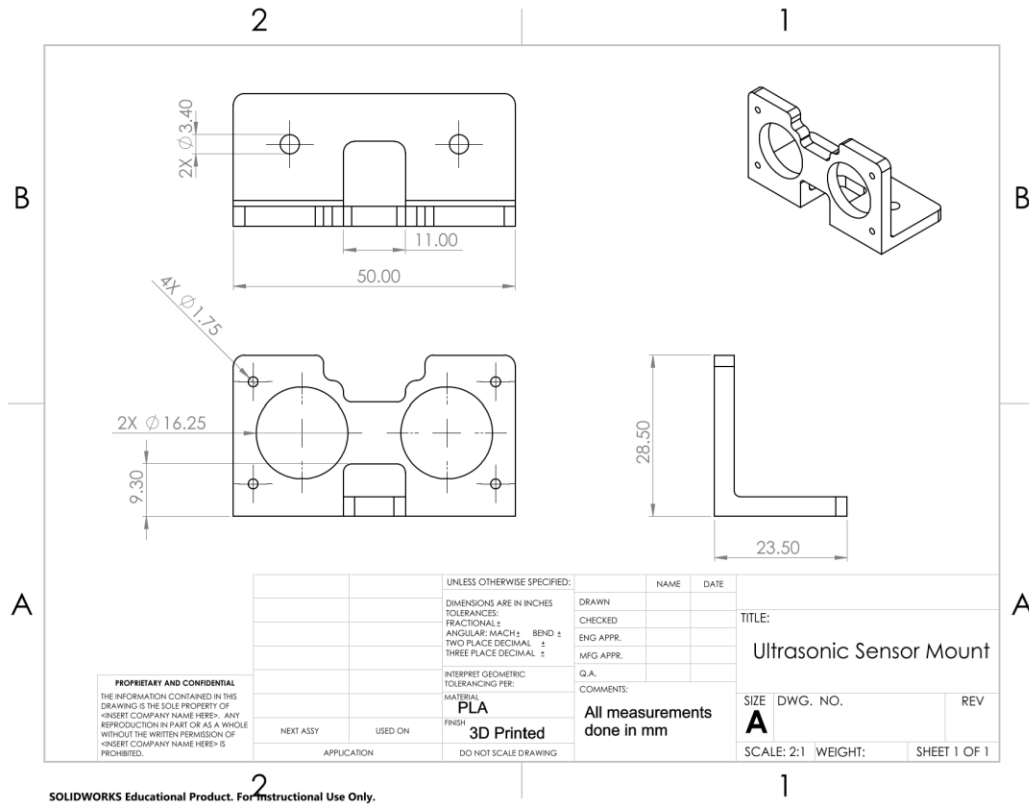
SOLIDWORKS Educational Product. For Instructional Use Only.











Appendix IX – Microsoft Project Team Schedule

