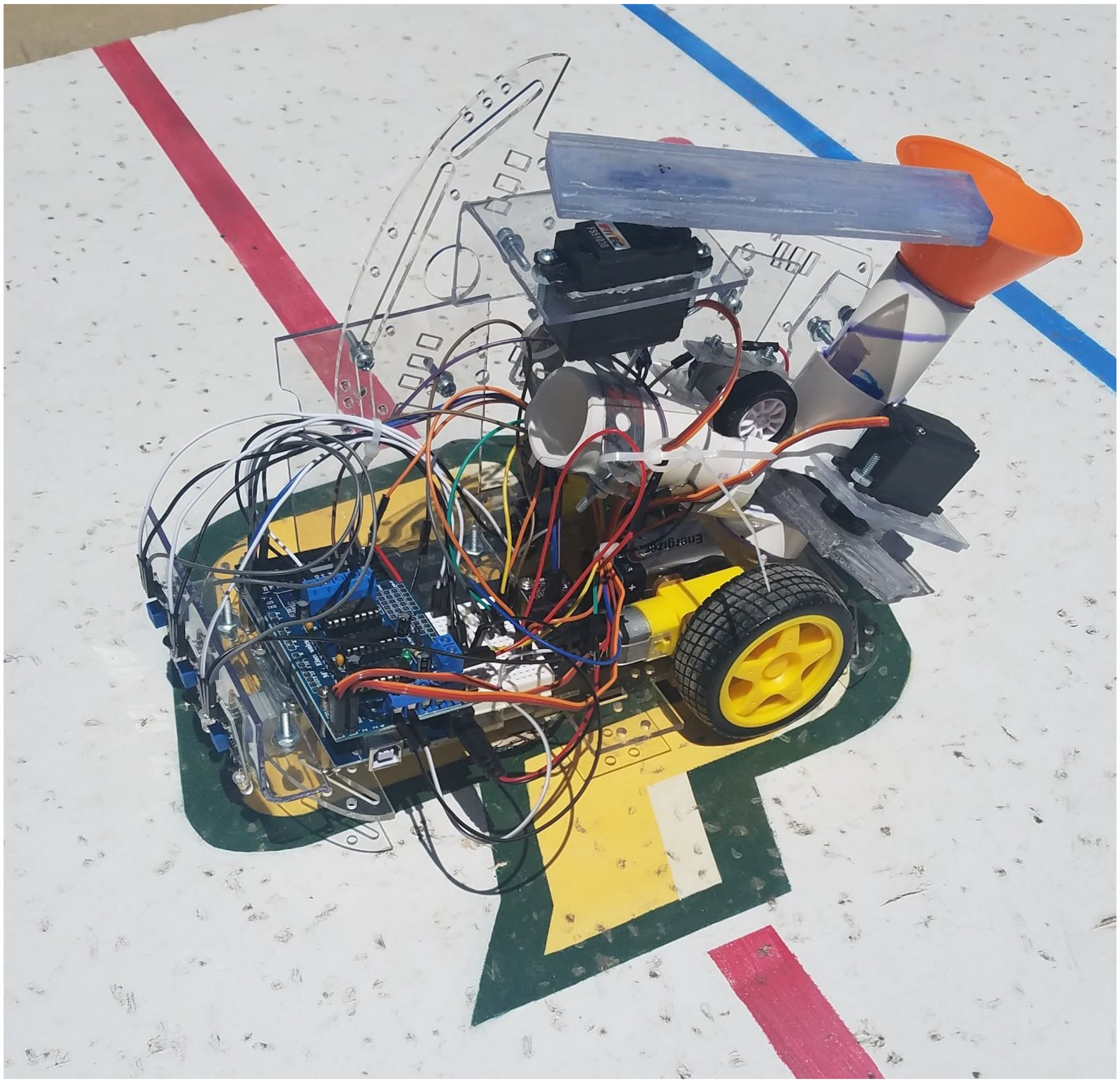
Senior Project - Roborodentia Robot



Nicholas Ilog

CPE 462 - 14 Spring 2018

Professor Seng

June 13, 2018

Table of Contents

Introduction

Every year Cal Poly hosts a robotics competition known as Roborodentia to showcase student created robots at Cal Poly Open House. These autonomous robots are assigned a variety of tasks to score points. In past years, robots collected balls and placed them in a designated location, located different colored cans, and picked up rings and hooked them on designated pegs to score points. This year, robots dispensed balls from a dispenser and shot them at targets across the playing field for points. There was also an optional objective to pick up balls at center court and dump them in the proper area. This report explores the design choices and implementation of the robot I created for this competition.

Problem Statement

Design and assemble an autonomous robot capable of traversing the course found in Figure 1 to complete a set of objectives for points. The robot shall dispense balls from the ball dispensers pictured in Figure 2. Ball dispensers are mounted 7" above the competition floor on the side walls. After collecting balls, the robot shall shoot the balls through hoops on the opposite side of the course. Each hoop corresponds to a different point value that can be found in Figure 3. There is a bonus for shooting through pairs of hoops as well as shooting through all four hoops. Additionally, robots may score by picking up "ultra balls" found in the troughs at the course center wall and dumping them in the "dump zone" found in the upper platform.

There are some limitations on robot design. Robots must start the match with a 12" x 14" footprint and may not expand past 14" x 17" at any point during the match. Height of the robot must also be below 14" at the beginning of the match, but there is no height limitation once the match begins. A robot cannot disassemble into multiple parts or go airborne during the match. Adhesives may be used by a robot, but they cannot damage the course, balls, or targets. Balls must also be shot at a speed less than 50 feet per second averaged over three test trials. Wireless communication with the robot is not allowed during the competition and jamming another robot's sensors during competition is also banned. Any wireless communication devices must be declared before the match begins. Failure to comply with these specifications will result in disqualification from the tournament.

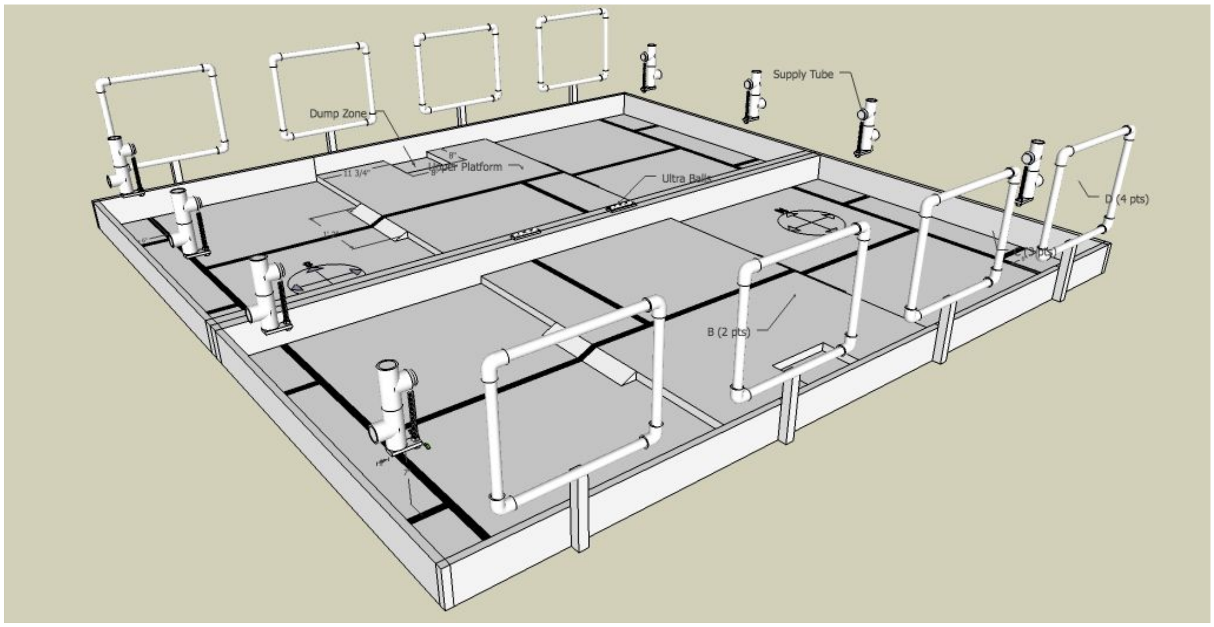


Figure 1 - Roborodentia Course Overview

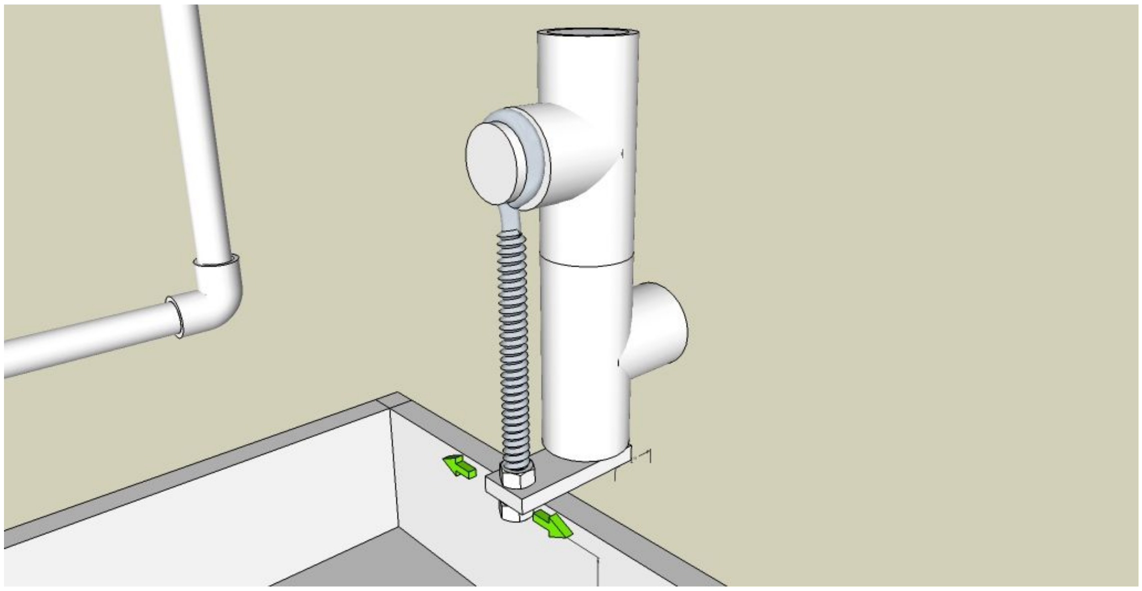


Figure 2 - Ball Dispenser Close Up

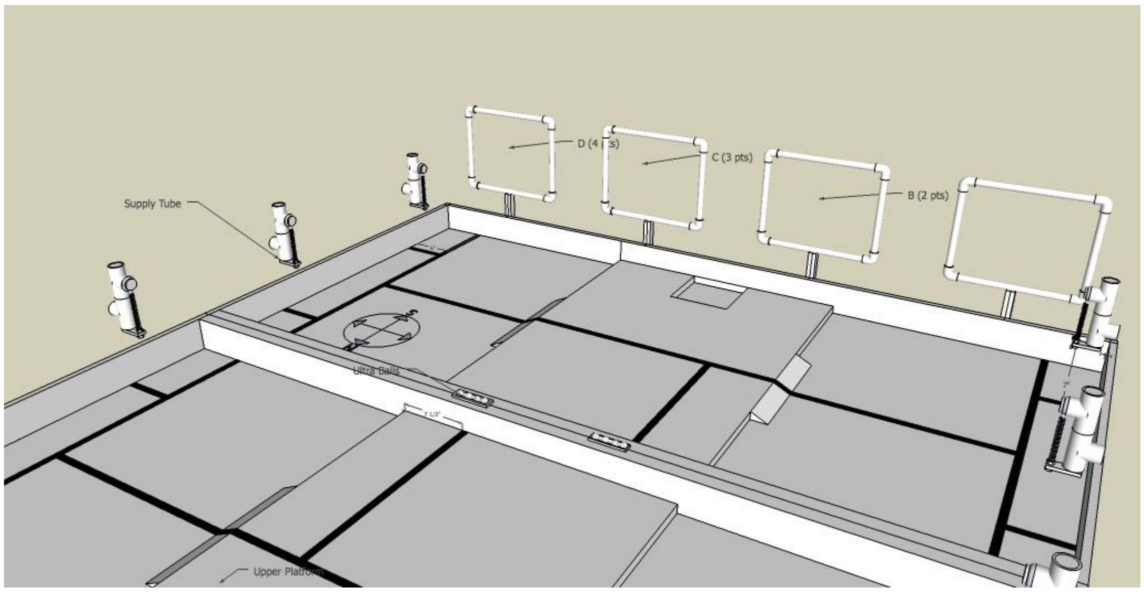


Figure 3 - Roborodentia Course Center View

Software

Software Block Diagram

$$\begin{tikzcd}
\boxed{\text{Setup}} \arrow[r] & \boxed{\text{Follow Black Line}} \arrow[r] & \boxed{\text{Shooting Routine}} \arrow[d] \\
& \boxed{\text{Dispense Balls}} \arrow[u] & \boxed{\text{Follow Black Line}} \arrow[l]
\end{tikzcd}$$

Figure 4 - Software Block Diagram

Software Overview

My code makes use of the Arduino Servo library to control two servos on the robot. One servo is responsible for moving an arm to dispense balls from dispensers, and the other is responsible for an arm that catches and feeds balls to the pitching wheel. The attach() function is used during setup to associate each servo with a specific pin on the Mega board. Then the write() function is called throughout my code to move the servo arms to the desired positions.

The drive shield also uses a library to control the speed and direction of motors. Using this library, the drive shield can control up to four motors. Each motor is represented in the code by the AF_DCMotor class. One AF_DCMotor must be declared for each motor that the drive shield wants to control. There are two functions in the AF_DCMotor class for controlling the motors, setSpeed() and run(). SetSpeed() is used to set the speed of the motor by taking the desired speed as a function parameter. Valid values range between 0 and 255 with 0 being off and 255 as full throttle. The run() function takes one parameter, the direction to spin the motor. Forward, backward, and release are the options here. Forward and backward are obviously forward and backward respectively, but release will stop the motor from spinning.

There are four major functions found in my code. These functions are the follow_line(), shoot_left(), shoot_right(), and dispense_balls() functions. The follow_line() functions polls from the IR sensors located at the front of the robot once a second and adjusts the motor speeds and directions appropriately to keep the robot on the black line. There are no adjustments to be made if the middle sensor is the only sensor to detect the line. If the left sensor detects the line, the robot must turn slightly to the left to stay on the line. In this case, the left motor is slowed to half speed while the right motor remains at full throttle. The opposite occurs when the line is sensed by the right sensor. There is a possibility that the robot does not make adjustments in time and it veers completely away from the black line. My code keeps track of the last sensor to track the line so that the robot can be guided back to the line. If the right sensor is the last sensor to track the line, the robot has veered off to the left of the line. The robot is turned slightly to the right and makes an attempt to find the line. Again, the opposite is true if the left sensor was the last to track the line.

Shoot_left() and shoot_right() are similar functions to one another except that the direction the robot turns is either left or right respectively. This function is called when the robot crosses the center black line that goes from one courtside to the other courtside. Depending on if the iteration is an odd or even one, the robot will have to turn left or right. There is a counter in my code to determine which direction the robot will have to turn to shoot at the correct targets. The robot will travel across the line, stopping to turn and shoot at each target. This process is rather time consuming as the robot has to turn to shoot at each target. Each target results in two turns because the robot must turn to face the target and then turn back to continue following the line.

Finally, the dispense_balls() function simply backs the robot up to the ball dispenser and opens the dispenser by moving a servo controlled arm. There is no error correction for lining up the robot funnel, arm, and ball dispenser. If the robot misses the ball dispenser, it misses the ball dispenser. A full cycle through the code and about 30 seconds of time can potentially be wasted when the robot misses the ball dispenser.

Software Functionality
During setup, the two servos and three dc motors are initialized. Each servo is connected to a pin on the Mega board and every dc motor is assigned a speed to spin at. After setup, the robot goes into a cycle of functions.

As mentioned above, there are four major functions in my code. The robot cycles between these four functions over and over again to score points. First, the robot calls the follow_line() function to follow the black line to the cross in front of a dispenser. Then the robot calls the dispense_balls() function to back up to the dispenser and dispense the balls. Next, follow_line() is called to follow the black line to the cross at the center of the side. Finally, shoot_left() or shoot_right() is called depending on whether the robot has to turn to the right or the left to shoot while traversing the middle black line. A counter is kept to make sure the robot turns in the correct direction. If it is an even iteration, the robot turns to the right. If it is odd, the robot turns to the left to shoot. This loop is repeated until time runs out. Since the dispense_balls() function has no error correction, there is potential for a full loop to be a waste of time as no balls will be dispensed to the robot. The robot will not attempt to dispense balls again until the dispense_balls() function comes back around the cycle.

Hardware

Hardware Block Diagram

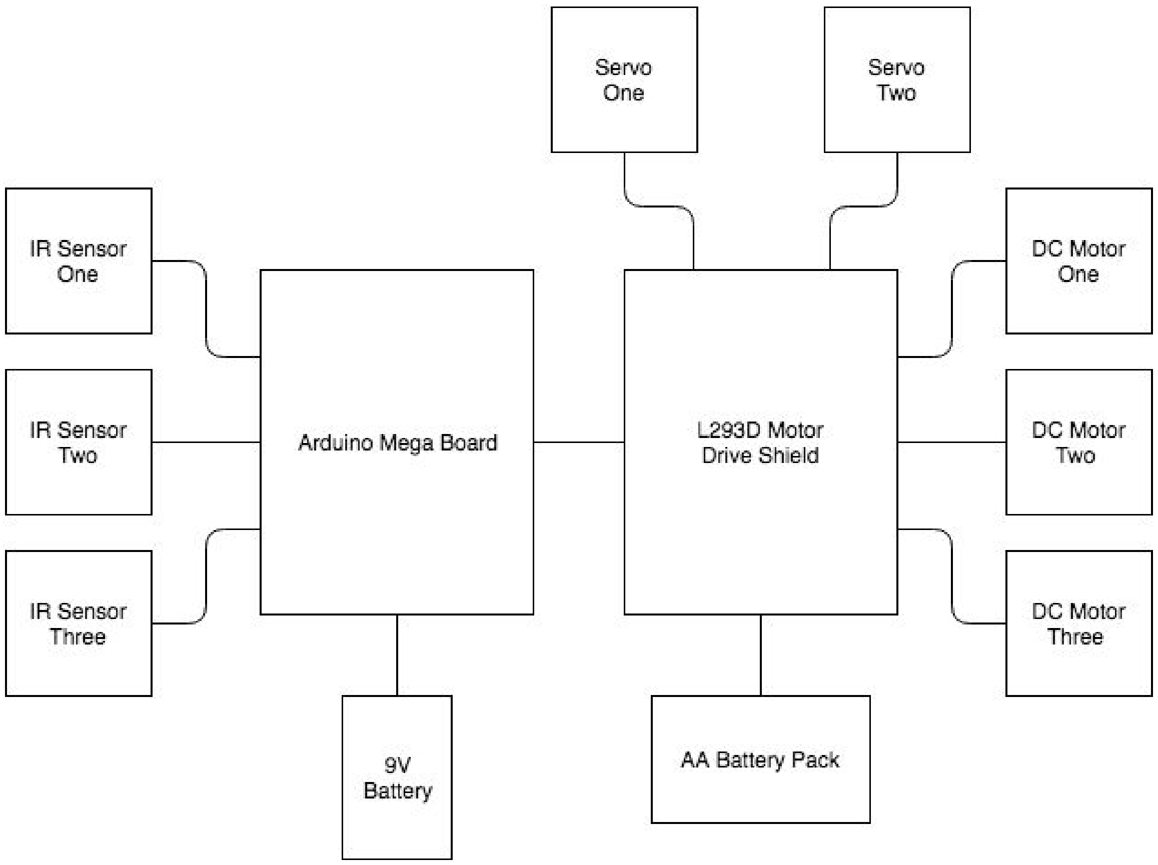


Figure 5 - Hardware Block Diagram

Hardware Overview

There are three IR sensors found at the front of the robot. Each sensor has a transmitter and a receiver LED. When there is no black surface to reflect the transmitter LED light, the output signal from the sensor is low. In the presence of a black surface, this light from the transmitter LED will be reflected back towards the sensor, allowing the receiver LED to detect this light. The signal from the IR sensor will be high in this case, and a green LED lights up indicating that a black surface has been detected. By using three sensors, the robot's location relative to the line can be determined, and the necessary adjustments can be made to keep the robot on the black line. More sensors could be used to more precisely track the line, but for the scope of this project, that is not necessary.

In addition to the three IR sensors, there are also three dc motors found on the robot. Two of these motors are used to spin the drive wheels of the robot. The speed and direction of these motors can be adjusted by changing the voltage applied to each motor. A lower voltage means the motor spins at a lower speed, and the voltage applied can be positive or negative to spin the wheel forward or backward respectively. Voltage to the drive wheels is constantly being adjusted to keep the robot on the black line. On the other hand, the third motor controls the pitching wheel and is kept at a constant voltage during its operation. This motor is only turned on when it is time for the robot to shoot balls. It is spun at full throttle to ensure shots have the distance needed to hit the targets. Even at full throttle, this motor does not spin fast enough to shoot balls faster than 50 feet per second.

Other motors on the robot include two servos controlling two plastic arms. One of these arms is responsible for opening the ball dispenser. The servo controlling this arm rotates 30° when it is time to dispense balls. Another servo controls the arm responsible for catching balls that fall through the robot funnel and feeding the pitching wheel. This servo rotates backwards to allow a ball to fall and then rotates forward again to push the ball into the pitching wheel.

The Arduino Mega Board acts as the brain of the robot. All code is executed by the Mega board. Information about the black line is relayed by the three IR sensors at the front of the robot to the Mega board. Then the Mega board determines what speed and direction the drive motors need to spin to keep the robot on the black line. Signals are also sent from the Mega board when the servos need to rotate to dispense balls and feed the pitching wheel. The pitching motor is also powered on and off at the control of the Mega board.

The drive shield is another board that gets physically inserted right on top of the Mega board. It acts as the interface between the Mega board and the motor components. Signals are sent through the drive shield from the Mega board to control the speed and direction of each motor individually. Protection for the Mega board from load voltages and relays is offered by the drive shield. Additional power supplies can also be plugged into the drive shield to offer separate power supplies for the Mega board and motor components.

A 9V battery powers the Arduino Mega Board while a AA battery pack powers the drive shield with four AA batteries wired in series. Since three IR sensors and two servos are all powered by 5V output pins found on the Mega board, power consumption from the 9V battery is very high. Each 9V battery only powers the Arduino Mega Board for a few hours before it runs out of juice. All motors are powered by the same battery pack providing power to the drive shield. The batteries in the battery pack are wired in series to create more voltage for the motors. Although the battery pack juice is not drained as quickly as the 9V, the batteries must still be changed out rather regularly. While the 9V powered components always turn on if given enough voltage, the battery pack powered components must maintain a certain spin velocity to be effective. When the batteries are too low, the motors spin too slow to move the robot or shoot balls.

Hardware Schematic

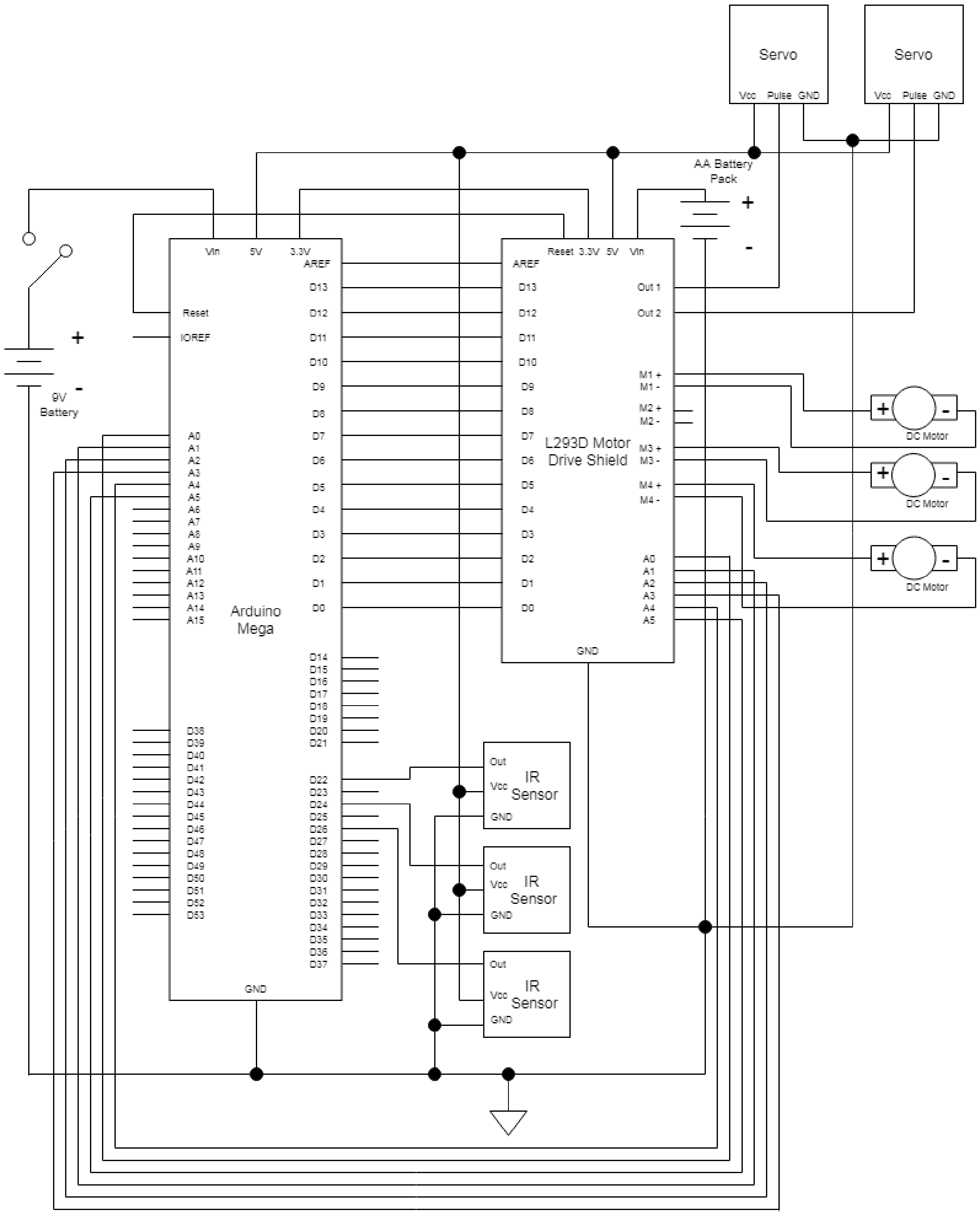


Figure 6 - Hardware Schematic

Mechanical

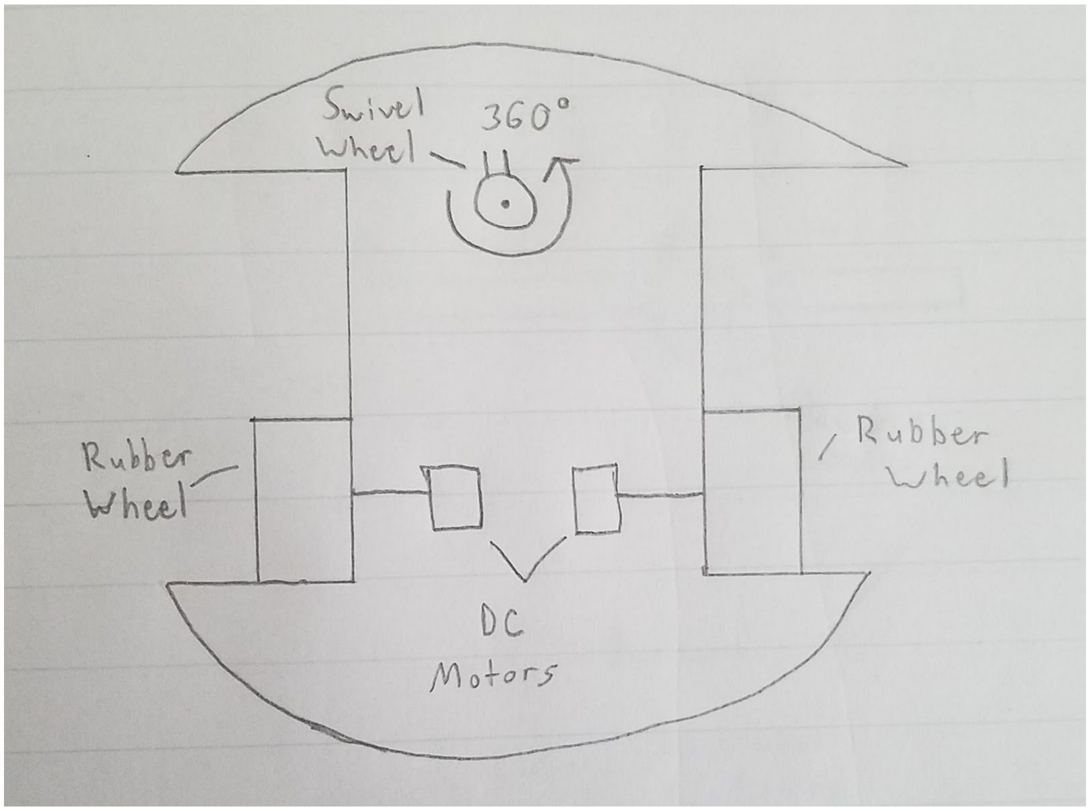


Figure 7 - Drive Aerial View

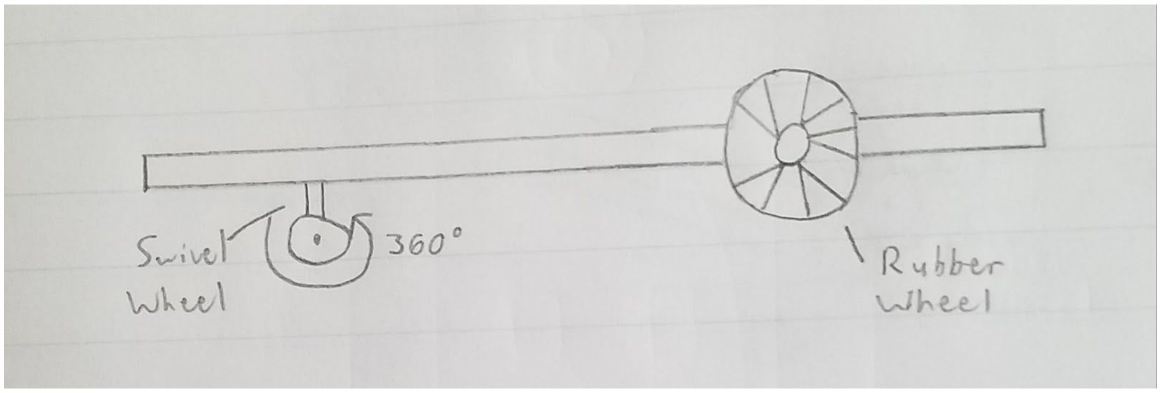


Figure 8 - Drive Side View

The robot is driven by two wheels powered by one motor each, and a swivel wheel towards the front of the robot with 360° range of motion pictured in Figure 7 above. Since the IR sensors are located at the front of the robot, the drive wheels are located in the back so that the robot has more control over turns. If the drive wheels were near the front, the robot be limited in its turning ability.

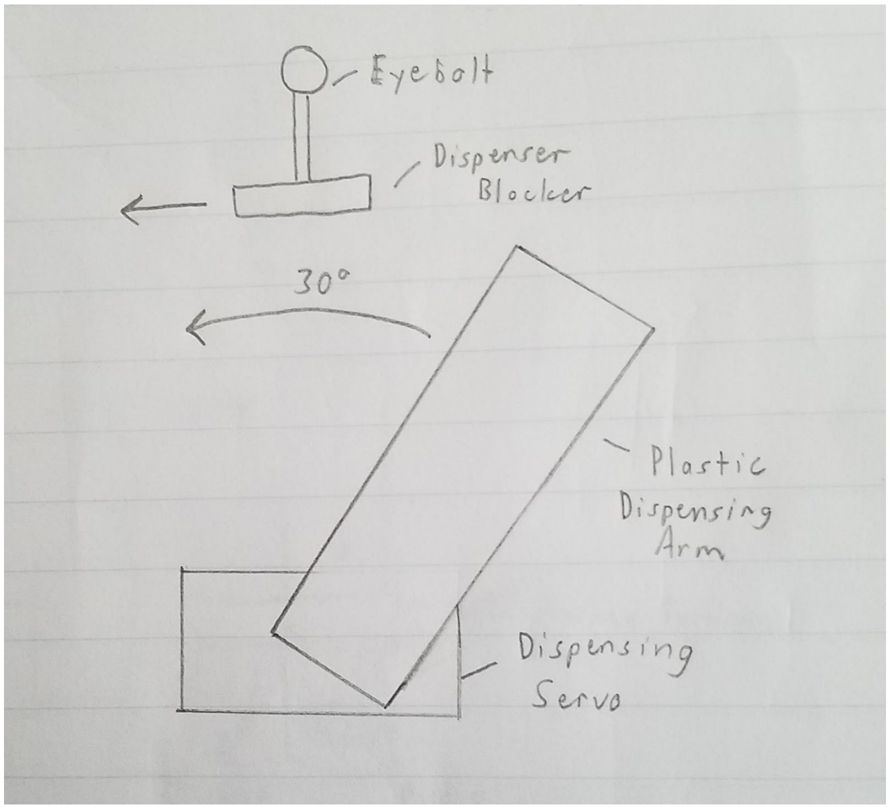


Figure 9 - Dispensing Arm Aerial View

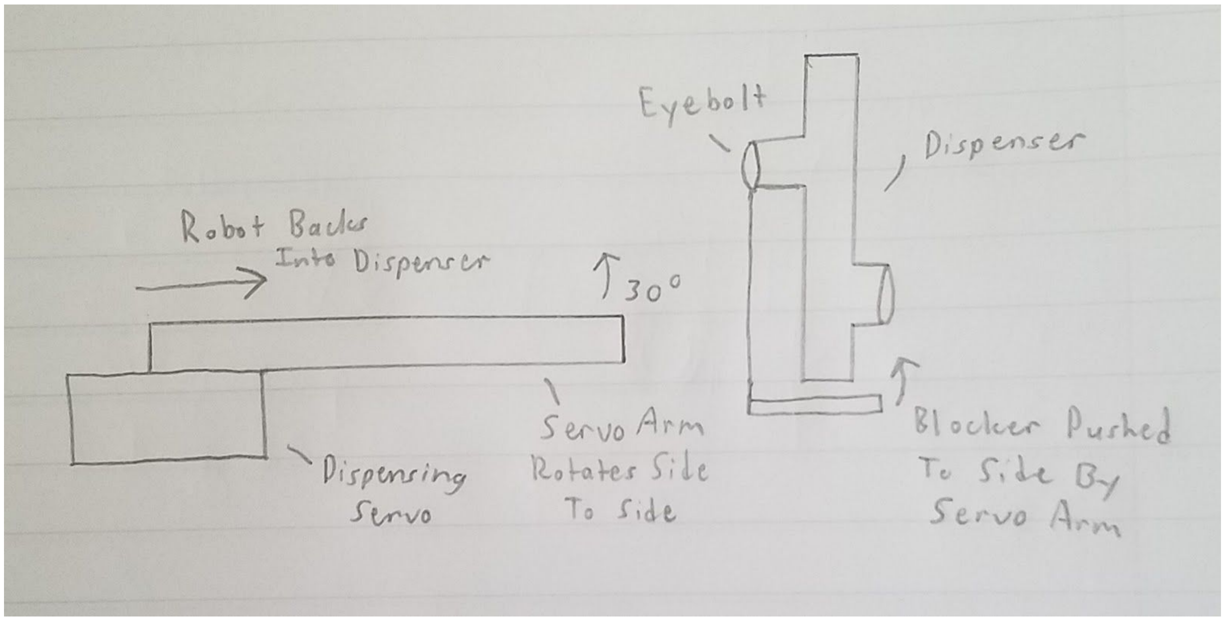

Eyebolt
Dispenser
Robot Backs Into Dispenser
30°
Servo Arm Rotates Side To Side
Dispensing Servo
Blocker Pushed To Side By Servo Arm

Figure 10 - Dispensing Arm Side View

Balls are dispensed from the dispenser with a plastic arm that is controlled by a servo pictured in Figure 9. The servo is mounted right above the pitching unit at 7” height. On top of the servo is a 1” x 5¼” rectangular plastic arm that is capable of swiping side to side to push the dispenser holder out of the way. This arm rotates about 30° to push the holder out of the way and allow the balls to fall through to the funnel found in Figure 11 below. Since the range of motion on this servo arm is limited to just side to side movement, the robot must be precise in its lining up of the funnel, servo arm, and dispenser. As mentioned above in the software section, there is no error correction for this alignment. It is entirely possible for the servo arm to miss the dispenser or for the balls to fall from the dispenser, but miss the funnel. In these cases, about 30 seconds of competition time is wasted because the robot will continue its function cycle without balls.

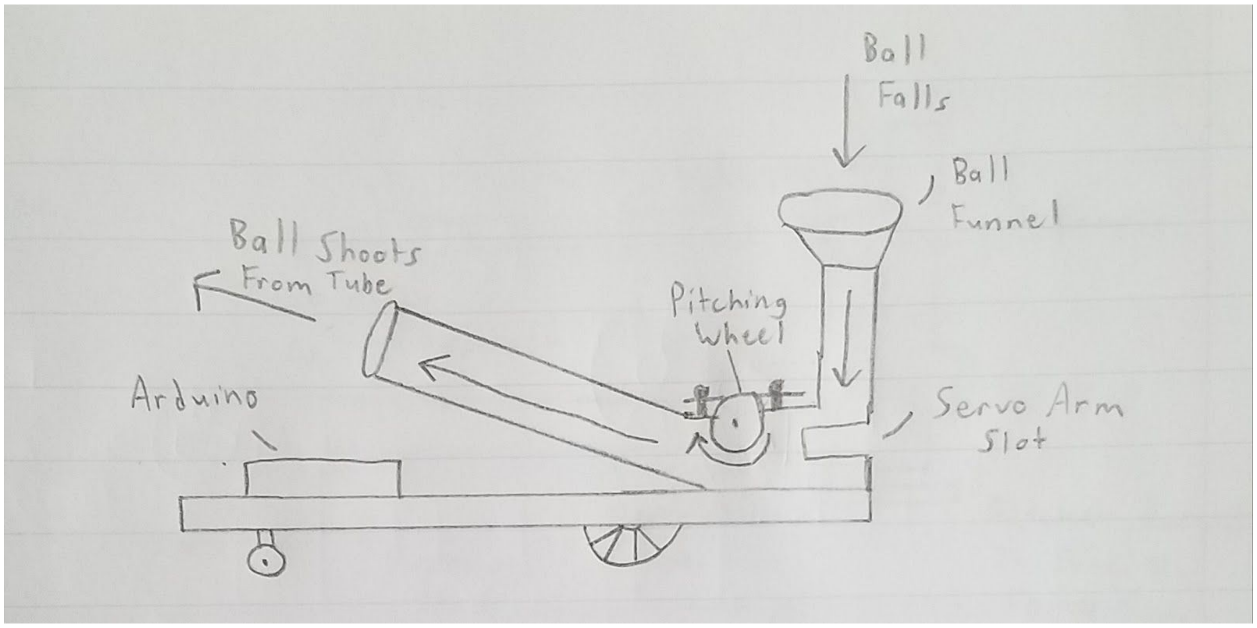


Figure 11 - Pitching Mechanics Side View

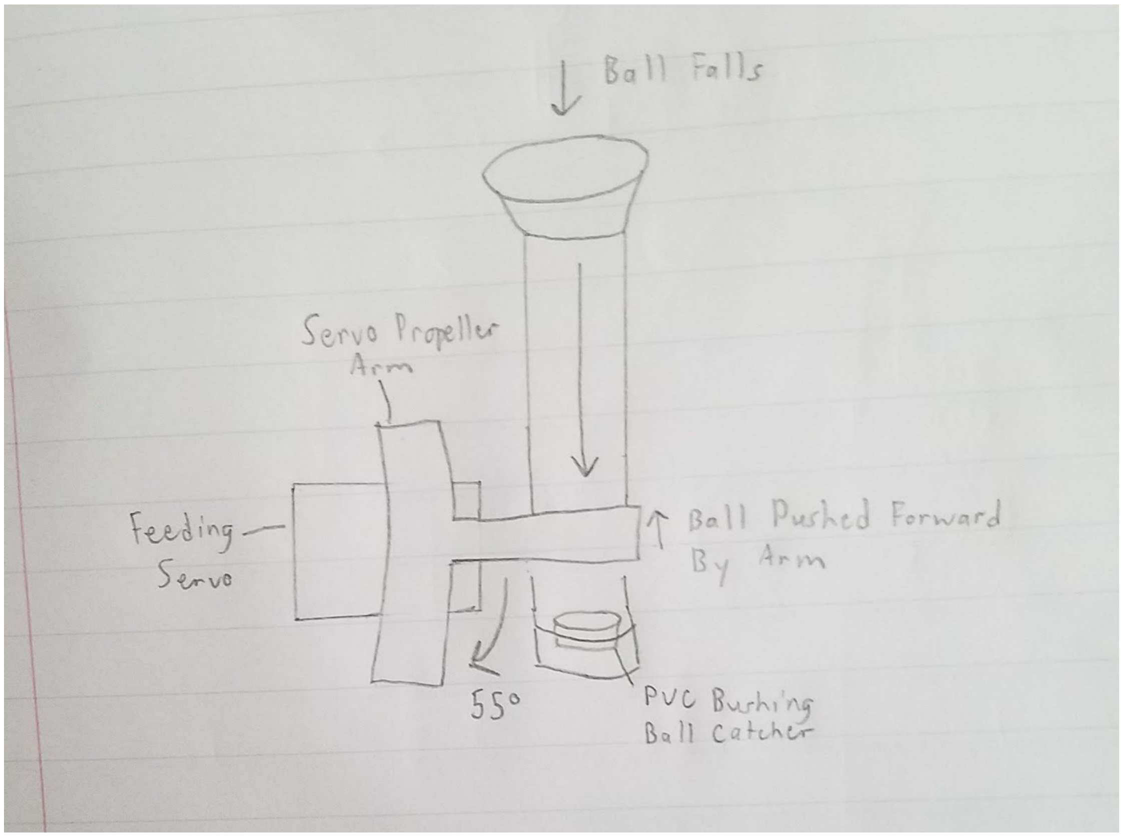


Figure 12 - Robot Funnel Aerial View

After balls are dispensed by the arm pictured above in Figure 9, they fall into the funnel and down into a pvc pipe. A servo arm shaped as a propeller catches the balls at the bottom of the pipe and holds them in place until it is time to shoot. When it is time to shoot, the servo arm rotates towards the back of the robot to allow one ball to fall and then rotates back to its original position to push the ball to the pitching wheel and catch the next ball that falls. Balls fall onto a platform constructed using a pvc bushing pictured in Figure 12. This ensures that balls do not fall and bounce towards the pitching wheel. By controlling the feeding of the pitching wheel, shots are more consistent. Once the ball reaches the pitching wheel, it is shot through a pipe towards the target. The pitching wheel takes a second or two before it gets up to full speed. Because of this, there must be a small delay between shots to allow the wheel to get back up to speed.

Like the dispensing arm servo, both the dc motor and servo are mounted to the pitching unit. The servo is mounted near the bottom of the feeding tube to ensure the servo arm is always in the correct spot to catch and feed balls. By mounting the motor in a fixed position, shots are more consistent. If the pitching motor were not mounted, the vibrations it generates would cause the wheel to move around in the pitching slot and shots would be random.

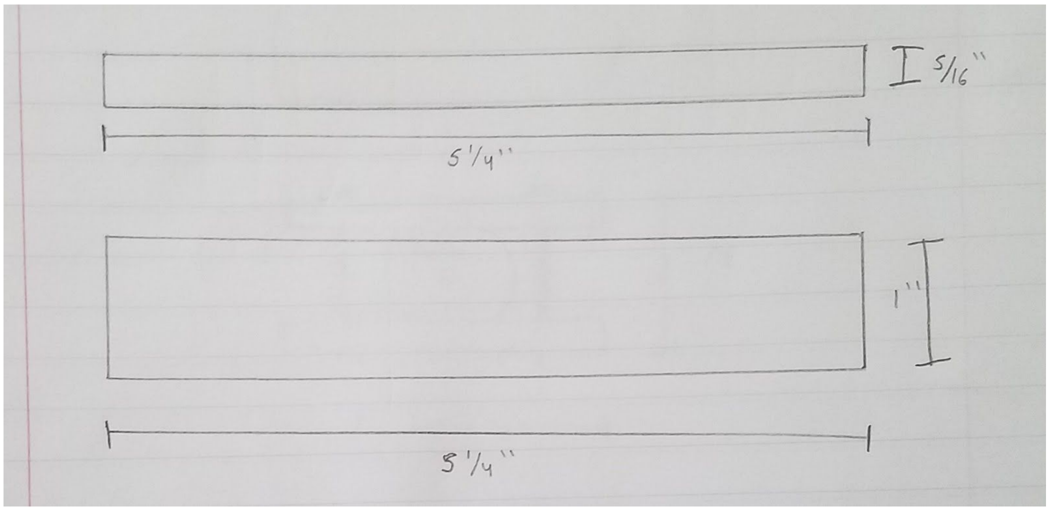


Figure 13 - Top: Dispensing Arm Side View. Bottom: Dispensing Arm Aerial View

Each plastic servo arm was manufactured by layering sheets of polycarbonate. The rectangular arm pictured above in Figure 13 was created by first cutting four rectangles from a larger sheet of polycarbonate. Then these four rectangular pieces were glued directly on top of each other and finally glued onto a servo propeller.

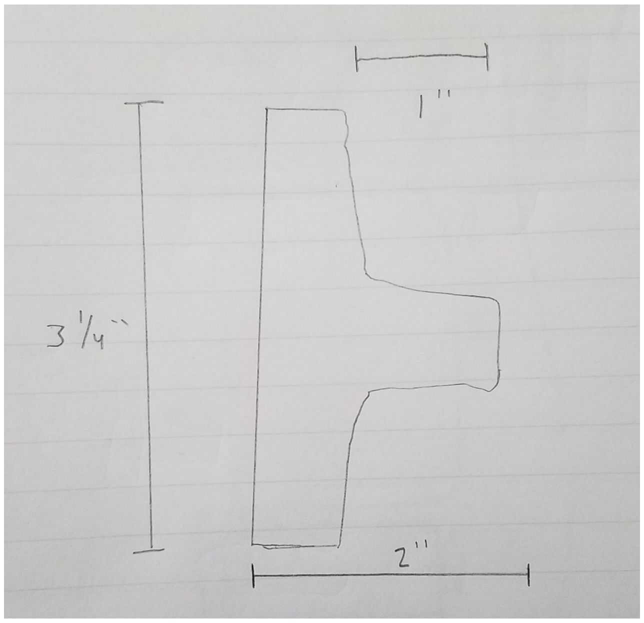


Figure 14 - Feeding Propeller Arm Aerial View

Servo propellers are rather small and are not large enough to work as the feeder for the pitching wheel. Another larger plastic propeller pictured in Figure 14 was made in a similar manner to the rectangular arm. Four propellers were cut from a larger sheet of polycarbonate and then glued together and finally glued to a servo propeller.

All mounts for the two servos and pitching wheel were also constructed using the same polycarbonate as the servo arms. The dispensing arm mount was created by first cutting a rectangular hole into a sheet of polycarbonate the size of the servo. Heat was then applied to the sheet to bend the sheet at a 90° angle and create an "L" shape. Holes were then drilled in the appropriate areas. Finally, the "L" piece was screwed onto the chassis, and the servo was inserted into the hole and screwed into place.

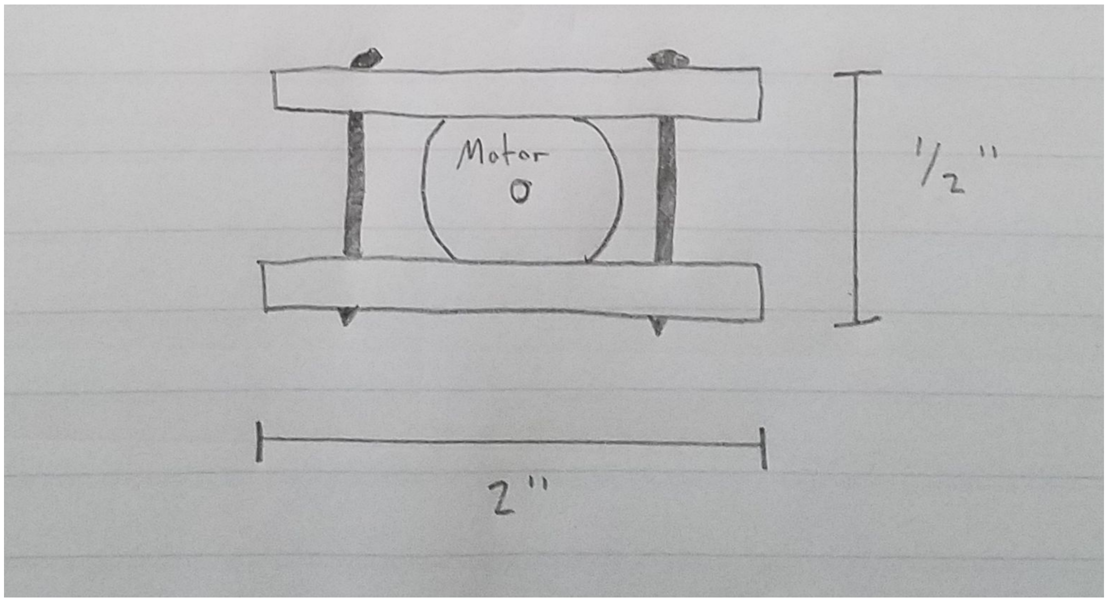


Figure 15 - Motor Mount Head-On View

Two small squares were cut out of a sheet of polycarbonate to make the pitching wheel mount shown in Figure 15. Both squares were then heated and bent into an "L" shape. These squares were then mounted to the chassis with the motor between them and screwed together as tight as possible.

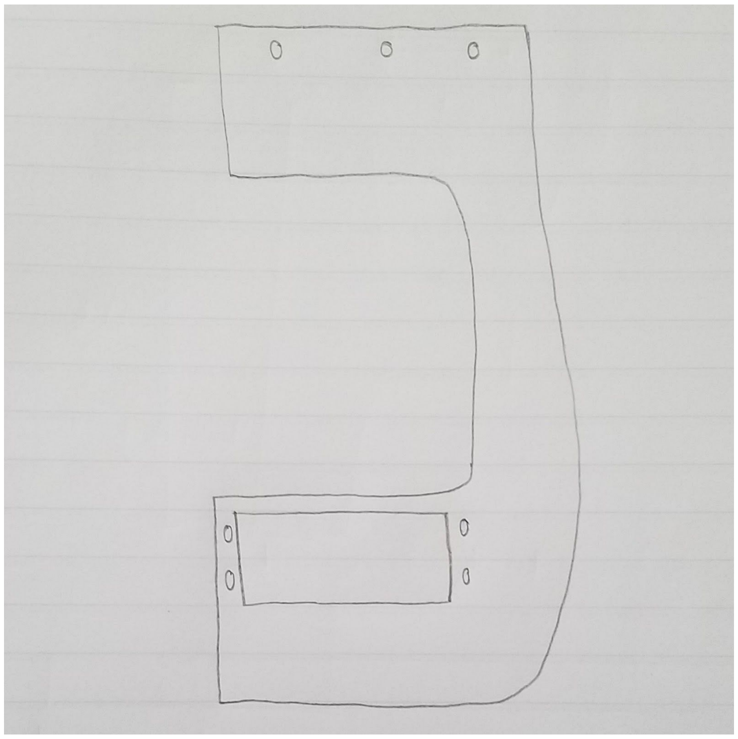


Figure 16 - Feeding Servo Mount Before Bend

The last mount was the piece that took the longest to fabricate. First, a rectangular hole the size of the servo was cut from a sheet of polycarbonate. The sheet was then cut into the shape pictured in Figure 16, heated and bent like the other pieces. This mount was then screwed into the chassis with the servo screwed into the mount.

Budget and Bill of Materials

There was no set budget when beginning this project, but the idea was to keep cost at a minimum. Components were purchased with this idea in mind and that may not have been the best course of action. A few of the IR sensors purchased simply did not work. Luckily, they were purchased in a pack of ten and only three were needed for this project. The dc motors that were purchased are inconsistent in their spin speed. Even when applied a constant voltage over multiple trials, each motor spins at a considerably different speed from its previous trial. This resulted in varying results between robot runs.

Other errors include the purchasing of kits when only one or two pieces were used from the kit. Out of the four wheel chassis kit, the only piece used was a piece of polycarbonate. The four wheels and motors found in the kit were just thrown to the side. Another kit that was not necessary was the plastic gear set. Although one of the robot prototypes did use multiple gears from this set, ultimately, only one small rubber tire was taken and used from this kit in the final project.

Even with these questionable kit purchases, the total cost of the robot is relatively low with a grand total of $131.76, as shown in the bill of materials below. I believe that if better IR sensors and motors were purchased and the individual pieces were purchased instead of kits, the price would still be around the $130-$150 range.

Item #	Part Description	Part #	Supplier Name	Quantity	Price	Extended Price
1	Arduino Mega Board	B01H4ZLZLQ	Amazon	1	$14.99	$14.99
2	L293D Motor Drive Shield	B00TMA4YSS	Amazon	1	$8.99	$8.99
3	Two Wheel Drive Robot Chassis Kit	B01L0ZY842	Amazon	1	$12.99	$12.99
4	Four Wheel Drive Robot Chassis Kit	B00NAT3VF4	Amazon	1	$15.99	$15.99
5	IR Sensor	B01I57HIJ0	Amazon	10	$0.99	$9.99
6	DC Motor	B073Q2Y3RC	Amazon	6	$1.16	$6.99
7	8” Ribbon Cables Kit	B01EV70C78	Amazon	1	$7.86	$7.86
8	Mini Breadboard	B01EV6SBXQ	Amazon	6	$1.31	$7.86
9	Plastic Gear Set	B079M5SY6D	Amazon	1	$10.99	$10.99
10	Plastic Funnel	B00OABYBJG	Amazon	4	$1.57	$6.29
11	9V Battery Clip	B00NIP0P9U	Amazon	5	$1.20	$5.99
12	Polycarbonate Sheet	SKU 987295	Home Depot	3	$4.98	$14.94
13	1” PVC Pipe	SKU 193755	Home Depot	1	$3.97	$3.97
14	1-½” x 1-½” x 1” PVC Tee	SKU 294179	Home Depot	1	$2.48	$2.48
15	1” x ¾” PVC Bushing	SKU 745078	Home Depot	1	$1.44	$1.44
					Total:	$131.76

Table 1 - Bill of Materials

Lessons Learned

There were many lessons to be taken from this project. The biggest lesson was the value in prototyping and testing. Each prototype improves upon the one before, and each test reveals new problems that need to be solved. Testing must be done in a proper manner though. Tests should be designed to fully analyze the product's functionality to make sure the product does everything that it needs to do. Some tests I performed focused on only using one component of the robot instead of everything together. The test would pass with just the isolated component and then fail once every component was implemented. Other tests were simply designed without the big picture in mind and did not truly evaluate the robot's functionality in the correct manner. The more iterations of effective testing and prototyping that a product goes through, the more developed the product is going to be. A more developed product will hopefully be more successful than a product that has only had one or two primitive prototypes.

Another lesson learned was that skimping out on certain components can be more costly than purchasing a more expensive, reliable component to begin with. The drive wheels and the motors powering these wheels were very unreliable. Each wheel had a design pattern on the rubber which affected how quickly the wheel could spin based on what direction the wheel was spinning. In turn, the motors also differed in how quickly they would spin even when powered by a constant voltage. Sometimes the motors would spin at max speed, but other times the motors did not generate enough power to even move the robot more than a couple inches over a few seconds. A stronger, more reliable motor would have ensured consistency between trials in the project.

If I were to restart from the beginning, there are a few design choices I would change. First, I would eliminate the need for the robot to physically turn and change its orientation by giving the bot a square shape. Then I would change the robot so that it uses the walls for tracking instead of the black line. It is quicker to just have the robot run into the wall and then drive along the wall to dispense balls than it is to follow the line to the dispenser. The method of dispensing balls would also have to change. Instead of having a servo arm dispense the balls, there would just be a static plastic piece at 7" height that hits the dispenser as the robot drives by. Balls would drop into a collection area found on either side of the bot. The pitching mechanism would still be similar to my current design. I believe with this redesign the robot would have potential for a much higher score in competition.

Conclusion

Overall, this project provided an educational and fulfilling experience. The project exposed me to the product design, implementation, and testing process. It also exposed errors or things I had not thought about during my design process that I can account for in the future. Assembling the robot and making sure all the parts worked together applied many of the techniques and processes that I had learned from previous courses at Cal Poly. The Roborodentia competition itself was also educational and fulfilling. Seeing how other people developed a solution to the same problem was interesting. I would recommend participating in Roborodentia to anyone that has an interest in robotics.