General Engineering Department

DemeterBot

Project Requirements Specifications

Version 2.0

Project Team Members: Jordi Del Castillo, Alvin Li, Brynja Schultz

1. Introduction

1.1 Purpose of the Project

Due to a lack of suitable land for agriculture to feed the growing human population, the United States National Aeronautics and Space Administration (NASA) has resorted to lunar agriculture. The moon has the potential to sustain 70% of Earth's current supply. However, the moon has a slower rotation period than the Earth. Because of this, stationary crops will only be exposed to sunlight for 27.3 continuous days before being covered in darkness. In order to maximize sun exposure for the plants, they have issued a request for proposal (RFP) for a rover capable of traversing the Lunar Agricultural Zone (LAZ), locate various crops, and move crops to areas with higher light exposure levels.

Our project mission is to help solve the world's hunger issue by fulfilling NASA's RFP and providing a means to efficiently cultivate crops through lunar agriculture. DemeterBot is an autonomous rover that travels across the surface of the Moon and collects different packages of plants and transports them to their designated area.

SeleneRobotics is a company dedicated to designing innovative robots that are specialized in traversing the moon. Our mission is to autonomously aid the sustainability of the growing human population through the innovation of lunar agriculture technology. To fulfill this mission, SeleneRobotics is made up of qualified team members. Head of Production, Jordi del Castillo, has managed many projects and maintained an efficient cost effectiveness. Head of Design, Alvin Li, has experience in coming up with designs for products and executing them in various computer aided design (CAD) software. Head of Programming, Brynja Schultz, has mastered multiple coding languages and skillfully applies them to many successful projects. With our combined expertise, our Demeterbot will be a decisive robot in the advancement of lunar agriculture that NASA should invest in. We hope that our contribution to the LAZ RFP will help alleviate the stress on available food sources on Earth.

1.2 Background Information

The task was to design and construct an autonomous robot that could complete the tasks laid out by NASA's RFP. In order to fulfill the requirements, the robot needs the means to travel and pick-up packages. DemeterBot moves by a two-wheel drive system powered by two motors and is capable of full rotational movement. When DemeterBot arrives at a package, it uses the lifting mechanism that inserts two forks under the package handle and lifts it with a motor-operated arm. DemeterBot can move while holding the package and deliver it to the destination.

2. Requirements

2.1 Physical Components

DemeterBot's physical construction consists of mainly two parts: a chassis and an arm.

The chassis consists of two flat metal plates and two folded metal plates. Two motors are connected to the folded metal plates. Four wheels are attached to the chassis and use front-wheel drive. Gears allow the rotation of the rear wheels.

The arm of DemeterBot consists of two long metal plates that act as a tower, a motor to lift the arm, and an extension from the tower to be inserted into the plant basket. Two teeth are placed at the end of the extension to hold the plant while the rover is moving. The motor is attached at the near end of the tower to connect the tower to the extension using gears. The tower is connected to the chassis.

A small metal plate is inserted between the two plates that consist of a tower to hold the Arduino board. The motor battery is connected to the Arduino board and is inserted beneath the small metal plate.

2.2 Software Components

The DemeterBot is coded using C/C++ in the Arduino integrated development environment (IDE). As shown in Figure 1, Arduino IDE allows code to be placed in setup to be run once and in the loop, in which code is repeatedly run until the program ends.

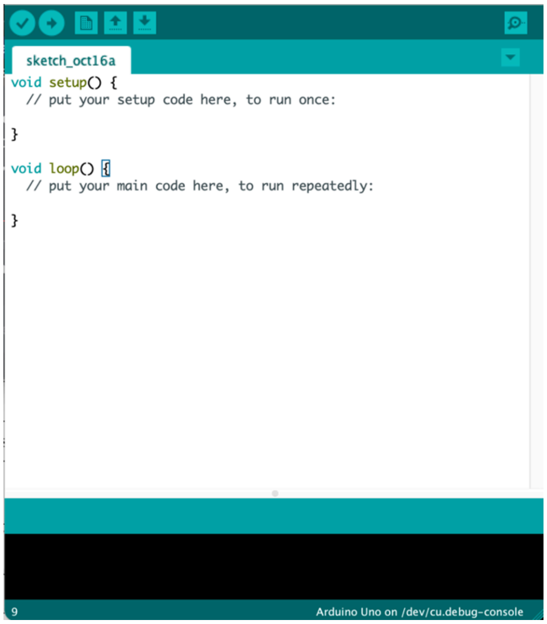

Figure 1: Arduino IDE

The libraries used include the EG VEX library provided by the manual.

Fusion360 was used to design a model of the robot as well as the company logo. The company logo was then 3-D printed using Cura. Microsoft Project was used to outline all of the tasks required for completion of the project and outlined all of the due dates and persons involved. Scheme-it was used to create the circuit and schematic diagrams to illustrate the wiring of the robot. Word was used to create the Preliminary Design Investigation, engineering notebook, and Final Design Report.

2.3 Project Schedule

The project workload is split based on our individual strengths and the time available. While we all aim to accomplish the tasks set in the beginning, our collaboration is relatively flexible. If one person is unable to complete a task, they may swap with another. Our engineering notebook is updated each time we meet as a group in the open lab. We broke up our tasks into two stages: before and after Benchmark A (Figures 2-3).

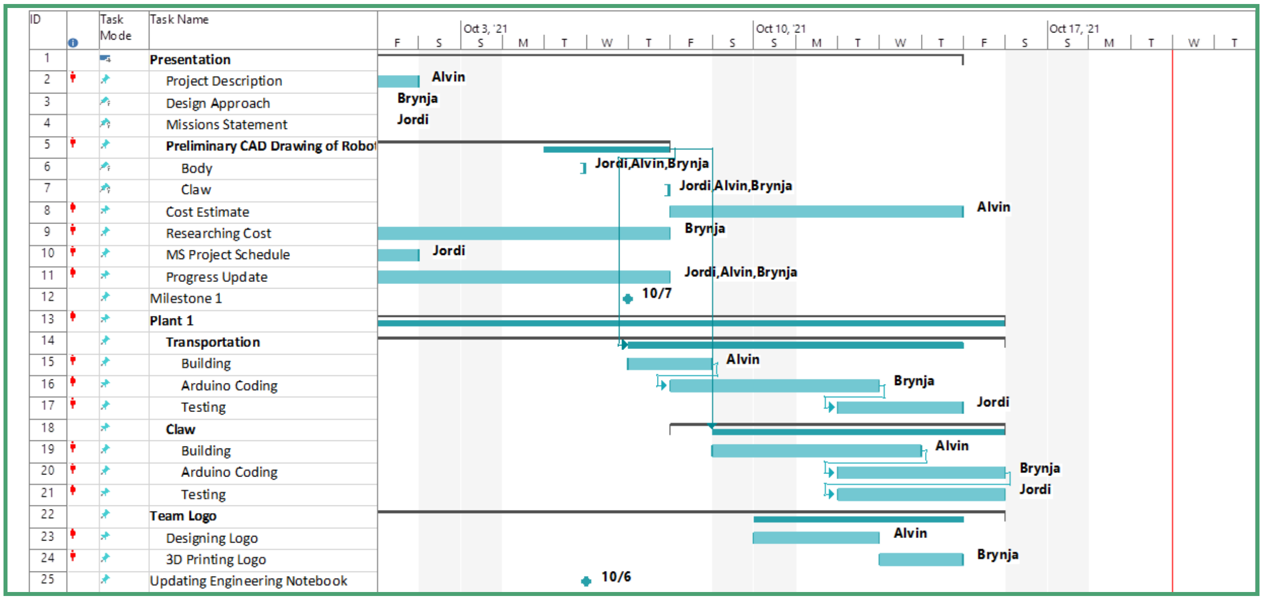

Figure 2: Project Schedule Stage 1

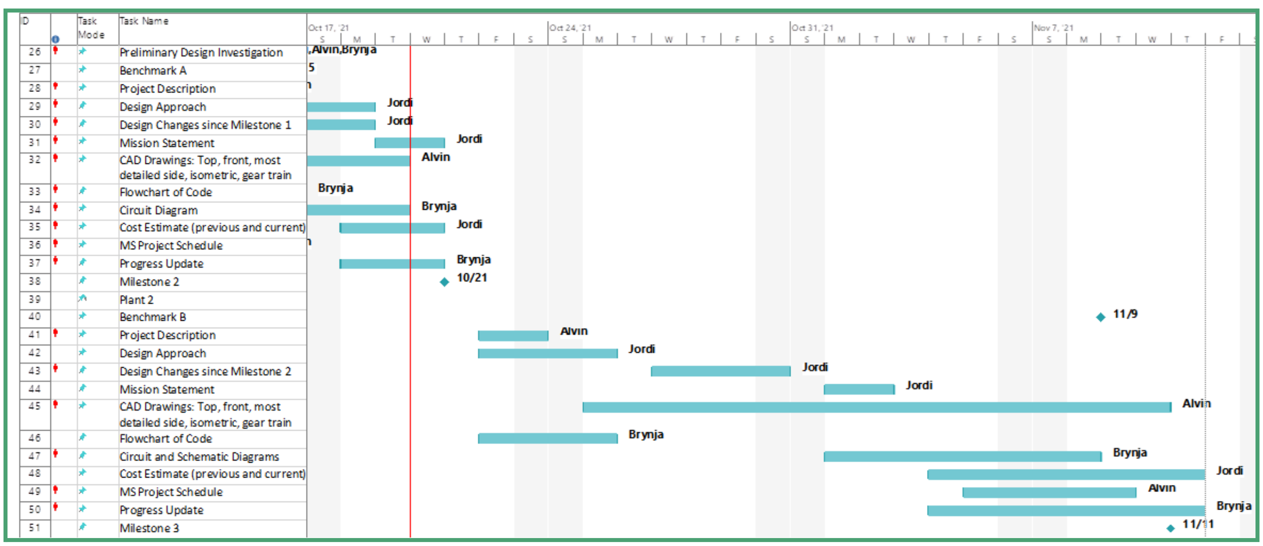

Figure 3: Project Schedule Stage 3

2.4 Cost Estimate

The costs of individual parts used to build DemeterBot are listed below in Table 1.

Table 1: Cost Estimate for DemeterBot

Description	Cost Per Unit	Quantity	Cost
Spur Gear, 84-tooth	$2.00	2	$4.00
Spur Gear, 60-tooth	$1.00	6	$6.00
Spur Gear, 12-tooth	$0.75	1	$0.75
Shaft Collar	$0.06	18	$1.08
Half Inch Screw	$0.09	50	$4.50
3 in. Shaft	$0.75	6	$4.50
2-Wire Motor 393	$16.99	3	$50.97
Keps Nut	$0.04	50	$2.00
4in Wheel	$5.50	4	$22.00
12in Shaft	$3.00	1	$3.00
7.2V Robot Battery NiMH	$34.99	1	$34.99
Arduino Board	$17.60	1	$17.60
Arduino Motor Shield	$10.00	1	$10.00
1x25 Aluminum Bar	$2.07	1	$2.07
1x2x25 Rails	$16.99	4	$67.96
C Channel 1x35	$9.99	1	$9.99
C Channel 1x10	$2.00	1	$2.00
AmazonBasics Battery	$3.00	1	$3.00
Anticipated Labor	$50.00	180 (hours)	$9000
Total			$9,246.41
Total with 20% Slack			$11,095.69

3. Procedures

3.1 Physical Construction

To construct the DemeterBot, a chassis was first built. Metal plate swerve connected using bolts and nuts (Figure 2).

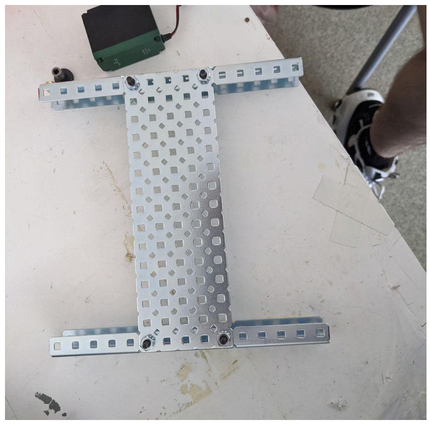

Figure 4: Skeletal Chassis

Two motors were attached to the chassis using screws. Gears were attached to the motors and plates on the sides of the chassis using shafts, bolts, and nuts. Wheels were placed on top of the gears so that they will turn when the gears rotate (Figure 3).

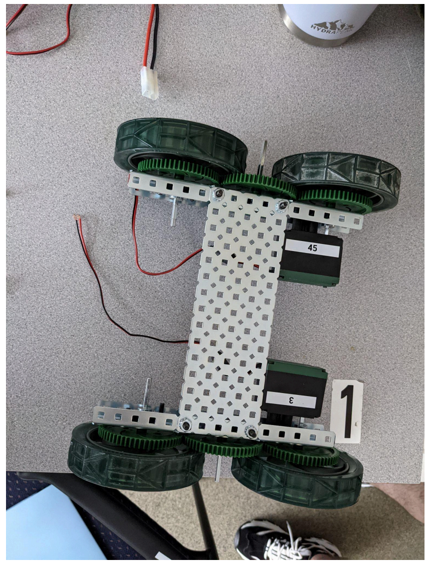

Figure 5: Chassis of DemeterBot

The tower of DemeterBot was constructed using two long metal plates and was attached to the chassis using bolts and nuts. A motor was attached to one side of the tower and a shaft going through the two plates. One small gear was added to the shaft. Two large gears were added on each side of the tower and a shaft went through the two large gears. The shafts were fixed using shaft collars. The extension was built using two folded metal plates and two skinny plates. The extension was connected to the tower using bolts and nuts (Figure 4).

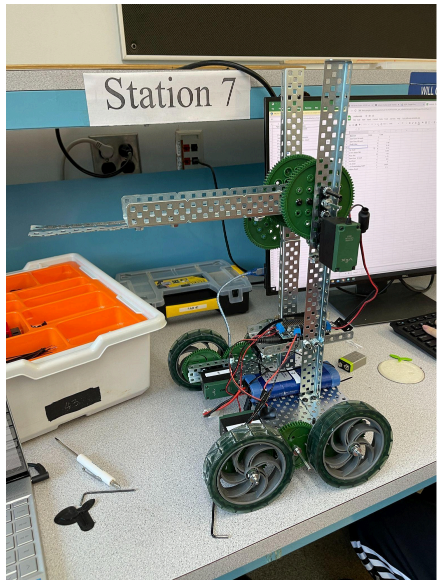

Figure 6: Arm and Tower Attached

The extension of the arm was modified by adding one flat wide plate and two standoffs- connected using bolts and nuts (Figure 5).

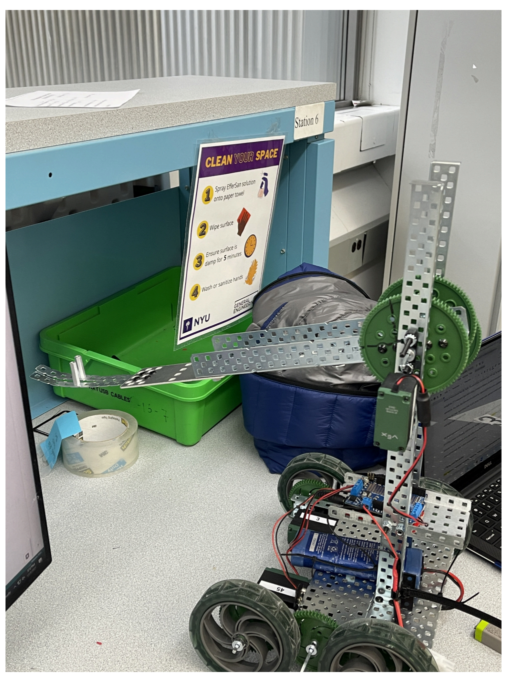

Figure 7: Modified Arm

A small metal plate was placed in the middle of the tower to hold the Arduino board. Three motors and two batteries were connected to the Arduino board. The motor battery and the Arduino board were fixed to the DemeterBot using zip ties.

3.2 Software Set-Up

The code controls the DemeterBot's three motors responsible for motion by initializing each of the motors as objects within Arduino. Two motors control the wheels and one motor controls the the claw (Figures 6 and 7).

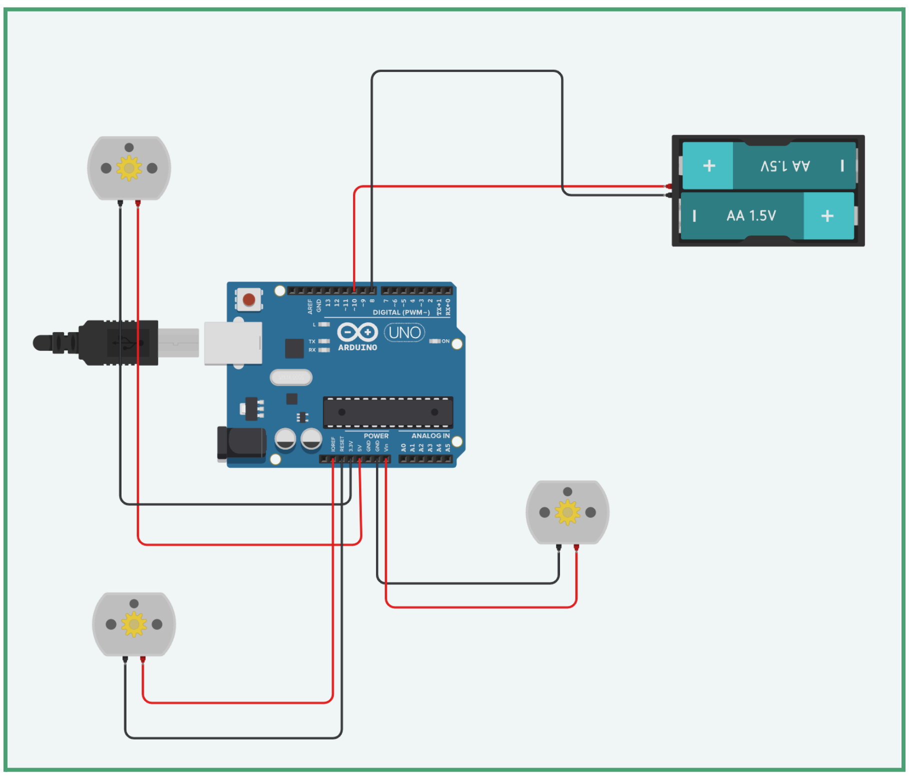

Figure 8: Circuit Diagram

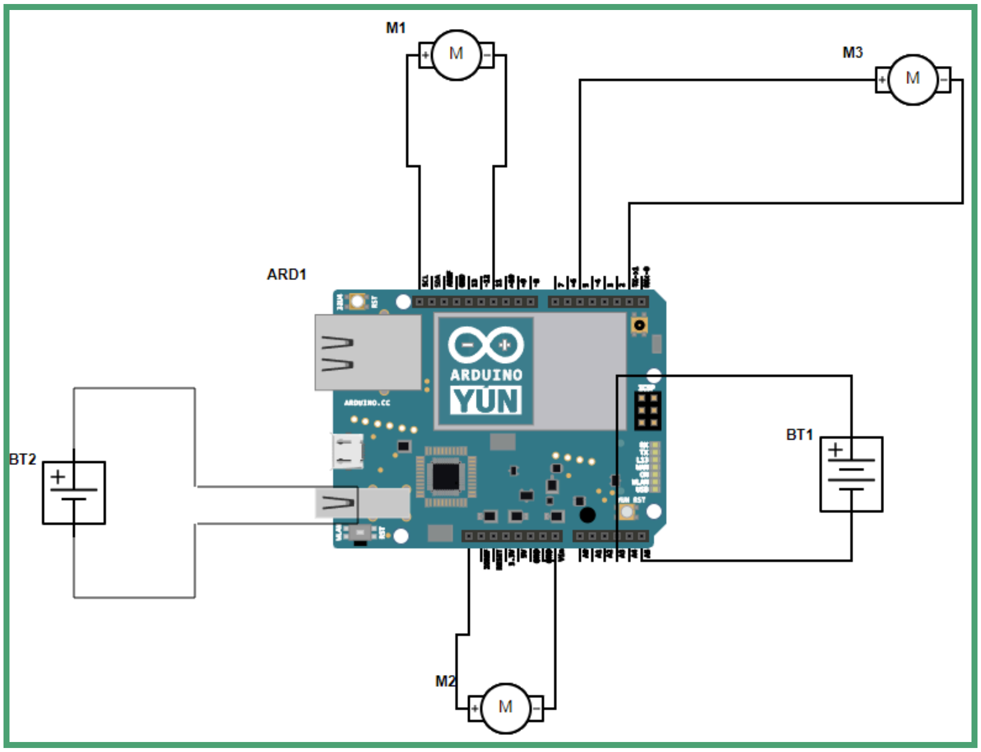

Figure 9: Schematic Diagram

As shown in Figure 10, the code instructs the motors connected to the wheels and claw to run for a set amount of time.

LAZ §

```
#include <Vex.h>

Vex Robot;


Adafruit_DCMotor *motor3 = Robot.setMotor(1);
Adafruit_DCMotor *motor45 = Robot.setMotor(2);
Adafruit_DCMotor *motor18 = Robot.setMotor(3);



void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Robot.begin();


}

void loop() {
  // put your main code here, to run repeatedly:
  Robot.moveTank(motor3, motor45, 50, 50, 7);
  delay(2000);
  Robot.moveMotor(motor18, 50, 2);
  delay(2000);
  Robot.moveMotor(motor18, -50, 1);

  Robot.end();

}
```

Figure 10: Benchmark A Code

This movement is what allows the robot to move from the plants to their drop-off points. The code also instructs the claw's motor to run when a plant is picked up or dropped off. A delay was added to the code before and after the claw moves in order to minimize the jerk of the plant to ensure its safe delivery. As shown in Figure 9, the code was written to allow the robot to successfully traverse the course while delivering the plants to their designated drop-off points with care.

```
          BENCHMARK A
    # move to plant 1
Move wheel motors
    # pick up plant 1
Delay
Extend arm
Raise arm
________________________________

          BENCHMARK B
    # move to plant 1 drop site
Move wheel motors
Turn 90 degrees left
Move wheel motors
Turn 90 degrees right
    # drop off plant 1
Delay
Lower arm
    # move to plant 2
Move wheel motors
Turn right
Move wheel motors
Turn left
________________________________

          COMMISSIONING
    # pick up plant 2
Delay
Extend arm
Delay
Raise arm
    #move to plant 2 drop site
Move wheel motors
Turn left
Move wheel motors
Turn right
Move wheel motors
Turn left
    # drop off plant 2
Delay
Lower arm
```

Figure 11: Initial Pseudocode

As shown in Figure 12, a delay was added at the beginning of the code to ease starting the robot. The route was also changed to pick up and drop off Plant 2 at an angle. Delays before and after turning were also removed and placed after movement of the claw to minimize jerk and allow stable transportation of the plants.

```
nclude <Vex.h>

Vex Robot;

Adafruit_DCMotor *motor3 = Robot.setMotor(1);
Adafruit_DCMotor *motor45 = Robot.setMotor(2);
Adafruit_DCMotor *motor18 = Robot.setMotor(3);

void setup() {
  Serial.begin(9600);
  Robot.begin();
}


void loop() {
  /** PLANT 1 **/
  delay(5000);

  // go forward
  Robot.moveTank(motor3, motor45, 50, 50.95, 5.5);

  //CLAW up --> plant 1
  delay(2000);
  Robot.moveMotor(motor18, 50, 0.75);

  //go backward
  Robot.moveTank(motor3, motor45, -50, -50, 1);

  //turn right (1/2)
  Robot.moveTank(motor3, motor45, -25, 25, 2.3);

  //move forward
  Robot.moveTank(motor3, motor45, 50, 50, 1.4);

  //turn right (2/2)
  Robot.moveTank(motor3, motor45, -25, 25, 3.13);
  delay(2000);

  //go forward
  Robot.moveTank(motor3, motor45, 49, 50, 3.4);

  //CLAW down --> drop off plant 1
  Robot.moveMotor(motor18, -50, 0.55);
  delay(2000);



  /** PLANT 2 **/

  //go backward
  Robot.moveTank(motor3, motor45, -50, -50, 3.35);

  //turn left
  Robot.moveTank(motor3, motor45, 25, -25, 4.5);

  //lower CLAW
  Robot.moveMotor(motor18, -50, 0.25);

  delay(2000);

  //go forward
  Robot.moveTank(motor3, motor45, 50, 50, 1.5);

  //CLAW up --> pick up plant 2
  delay(2000);
  Robot.moveMotor(motor18, 50, 1.68);

  //PLANT 2 DROP OFF
  //go backward
  Robot.moveTank(motor3, motor45, -50, -50, 3.2);

  //turn right
  Robot.moveTank(motor3, motor45, -25, 25, 2.4);

  //go forward
  Robot.moveTank(motor3, motor45, 50.3, 50, 5.3);


  //claw down --> DROP OFF PLANT 2
  Robot.moveMotor(motor18, -50, 0.5);


  Robot.end();
}
```

Figure 12: Final Code

3.3 Software Troubleshooting

During the initial coding phase, a gyro sensor was attached to the robot to aid in the turning of the robot. However, after much difficulty with the sensor, this feature was removed and the turning of the robot was manually coded.

There was also excessive difficulty with the jerkiness of the robot, causing the turns to be inconsistent. The delays coded before each turn were, therefore, removed. The remaining delays were kept during any movement of the claw to allow the plant to settle on the claw before transporting it.

Difficulty also occurred during the uploading phase in which the code would not upload to the robot, although the Arduino was plugged into the Lab PC. To prevent this error from occurring, the two batteries were consistently removed prior to uploading. After a small electrical fire, the Arduino Uno was replaced, and this issue no longer occurred.

During the process of coding the robot, the path of the robot consistently changed. To prevent coding issues stemming from changing the code too much, a file containing the code was uploaded after each meeting to the shared Google Drive file to allow us to return to the previous code.

4. Milestone and Final Product Requirements

4.1 Benchmark A Requirements

Benchmark A required that the LAZ robot be able to reach Plant 1 and pick it up with a functioning claw.

4.2 Benchmark B Requirements

In addition to the requirements already laid out by Benchmark A, Benchmark B required that the LAZ robot be able to travel to a designated location and drop off Plant 1. Then, the robot must travel to Plant 2 and touch it.

4.3 Final Commissioning Requirements

In addition to the requirements laid out by Benchmark A and Benchmark B, the robot must be able to reach another designated location and drop off Plant 2. Extra credit opportunities were to complete said requirements one week in advance or deliver a third plant.

4.4 Human Resources and Training

Teaching assistants were present in the open lab and provided assistance during the construction of DemeterBot. They helped figure out issues we had with the Arduino IDE, such as issues with uploading or the hardware being unresponsive. During the physical construction, they were able to answer questions we had and any requests for additional materials. They were also able to replace any malfunctioning parts such as a defective Arduino UNO board that was the root cause of many of our initial problems. YouTube videos were watched to understand how to make a functional claw with the given Vex Robotics parts. The training was done in the MakerSpace to understand how to utilize 3D printers for printing needs.

5. Results

5.1 Benchmark A Results

DemeterBot successfully reached and picked up Plant 1, completing Benchmark A. The Preliminary Design Investigation was created, and the engineering notebook was updated to include the work done to improve the robot.

5.2 Benchmark B Results

DemeterBot successfully delivered Plant 1 to its drop-off site and reached Plant 2, completing Benchmark B. During the benchmarking process, the code was no longer able to be uploaded to the robot. However, this issue ceased when a new Arduino Uno was placed on the robot.

5.3 Difficulties Experienced

While testing Demeter Bot, the rover occasionally didn't stop where it was supposed to stop and kept going forward. We found out that the Arduino board was causing the problem and the rover started to function normally once the Arduino board was replaced.

We initially tried to use a gyro to control the turning of the rover, but the gyro didn't function as denoted by the code and kept going around circles without stopping. We solved this problem by hard coding the motors to make the rover turn.

When Jordi tried to connect the battery to Demeter Bot, the wire that connects the battery to the Arduino board caught on a small fire. The small fire melted the wire, but no one was hurt. We replaced the battery cable with a new one and solved the problem.

One factor that had to be considered throughout the construction and testing process was the fact that the speed at which the motors turned was proportional to the power remaining in the Vex Robotics battery. Due to this, DemeterBot would be traveling at different speeds each time so its path would be significantly different from previous trials.

6. Conclusion

6.1 Results of Project

DemeterBot was able to complete all tasks laid out in NASA's RFP. It reached Plant 1 and brought it to the designated spot. It then located Plant 2 and brought it to the final destination. With a cost of approximately $11,000, DemeterBot was a successful project.

6.2 Future Improvements

To improve this design, plates would be placed on the outside of the wheels on both sides. This would ensure that the wheels stay in place and allow the robot to move in a consistently straight path. The robot would also be made to be shorter, with a longer claw to reach the plants placed on higher blocks. The robot would also utilize the walls when traversing the LAZ, this would ensure a more consistent path.