Principles of Engineering

Mini-Project 3 - DC Motor Control

Repository Link: `https://github.com/toluooshy/PIE-Mini-Project-3`

Section 1

Aaron Huang, Leopoldo Sanchez IV, Walter Villa, & Tolulope Oshinowo

Due: October 21, 2021

Abstract

The goal of mini-project 3 was to make our own line-following wheeled robot. We created a two sensor scanner with IR sensors in order to distinguish the floor and a black piece of electrical tape. The wheeled robot makes slight adjustments left and right according to the readings that either of the two scanners produces. The IR sensors have to read less than 600 in order for the wheel to turn. If it had to make a sharp turn, it had to read a value that is greater than 600 and less than 750. We assembled the body of the wheeled robot out of the provided parts (acrylic chassis, Arduino, DC motors, Arduino/Motor-Shield combo, and a wheel).

Materials

- 3D printed filament
- Arduino
- USB cable
- 2 IR reflectance sensors
- 1 12V power supply
- 1 Adafruit v2.3 Motor Shield
- 1 male barrel jack to screw terminal connector
- 1 Four pin Molex wire-to-wire connector
- Some 24AWG stranded, silicone-insulated wire
- 2 DC motors and wheel
- 1 acrylic chassis platform with caster
- 1 Breadboard

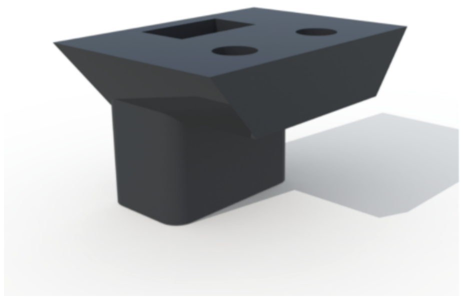
Robot Assembly

We originally began assembling the chassis of our robot according to the instructions given, but also found it helpful to watch this video in parallel. Once the chassis was successfully assembled, we then began to plan out how we would go about attaching the necessary circuitry. Because of the fact that one pair of our group was stronger in mechanical design, and the other in electrical-software design we decided to split up the work from this point on. The mechanical pairing would handle the prototyping and fabrication of a mount to the chassis that could effectively suspend our two IR sensors above the ground and allow for the sensors to guide the robot based on their respective readings. The electrical-software pairing would handle coming up with a space-effective circuit that would relay data from the sensors to the Arduino, and the controller script.

Sensor Mount Design

The optimum distance for the two IR reflectance sensors is 0.2 mm-15mm (peak 2.5 mm) from the ground so we had to make a mount. We chose to create a dovetail slide design because it is modular and adaptable. There were some unknowns when we started this project, such as “How close do the two sensors have to be?” or “How many sensors do we need?” Having a dovetail design allows us to easily change the distance of the sensors, add more sensors, or swap out parts if needed. Some CAD renders can be seen in **Figure 1.0**.

You can see in figure **Figure 1.1a** that there are 2 holes on the sensor mount. M4 screws can be screwed in to stop the sensor mount from sliding. Cutouts and holes were made to let the sensor wires connect to the Arduino as seen in figure **Figure 1.1b** (Note: We did not use the holes on the side of the cylinder because it would be hard to thread the wire through). The body of the mount is made of two pieces and press-fitted together because the acrylic chassis gets in the way.


(a) CAD Render Sensor Mount.

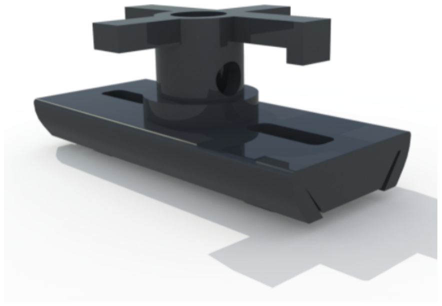
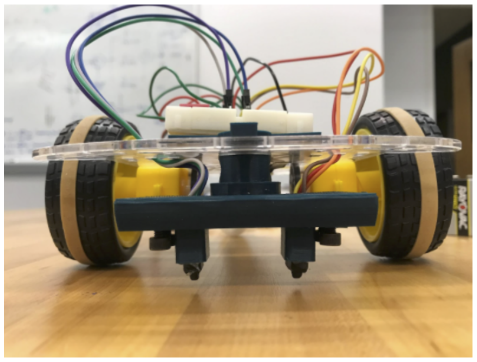

(b) CAD Render Isometric View.

Figure 1.0 - Mount CAD Renders.


(a) Mount Front View.

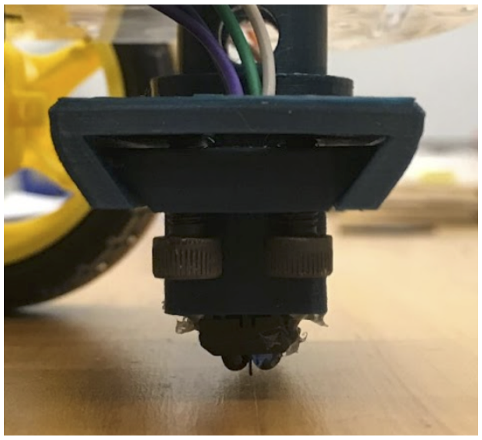

(b) Dovetail Up Close.

Figure 1.1 - Mount Photos.

Building our Circuit

Thanks to a newfound level of familiarity with the infrared sensor from the previous mini-project, we understood their basic functionality and began to translate what we knew about the sensor from MP2 into the logic behind the circuit setup for the new sensors in MP3. We started out by building the circuit for connecting the IR sensors provided by the teaching team in the project description. To accomplish this we used **Figure 2.0** as a guide.

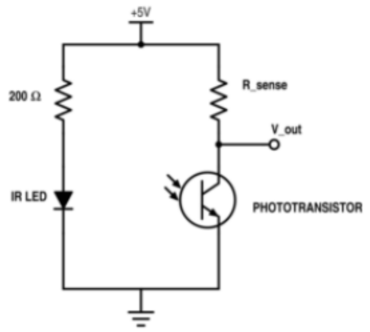
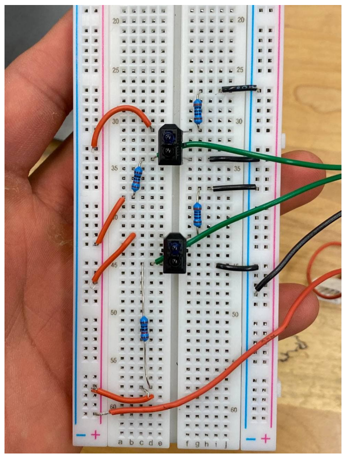
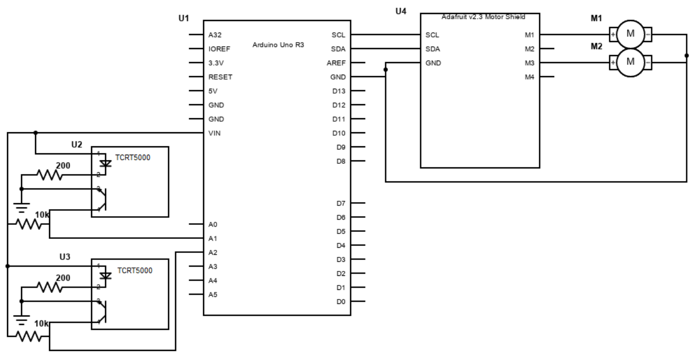


Figure 2.0 - Circuit diagram for the TCRT5000 IR sensor.

We decided to use a two resistors, one of 200Ω and one of 10KΩ based off of the TCRT5000 data sheet, since these were values that were recommended on the data sheet for the IR sensor. We built out this circuit diagram twice because we wanted to include two sensors on our robot in order to be able to control both left and right motors. In the end we ended up with a circuit as shown in **Figure 3.0**.



(a) Circuit Photo.



(b) Circuit Diagram.

Figure 3.0 - Final Circuit.

Testing The Sensor

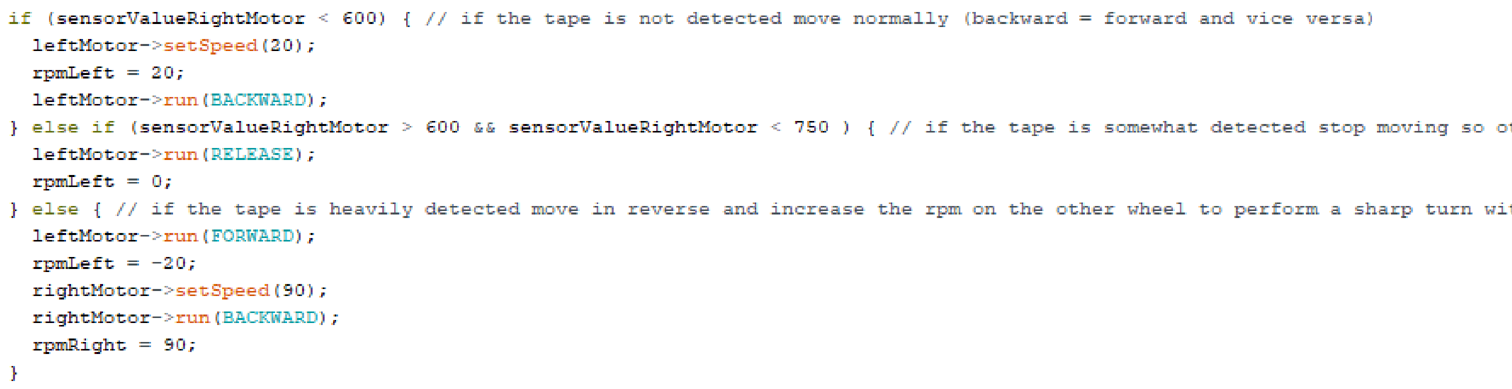
In order to test our IR sensors with our current circuit, we opened up the Serial Monitor on the Arduino IDE and ran the sample code from `AnalogInOutSerial`. This was the way we tested our values in our previous mini-project, so we decided that this would be the best way to test our sensors this time as well. If our values were able to read different values by simply hovering our fingers over the sensors to see how responsive they were to different objects/surfaces.

Finding the Threshold between Tape and Floor

Once we were able to get a responsive reading from the sensors, we decided to get the robot and sensors close to the ground in order to observe what values the sensors were reading. These readings were the main ingredients in helping our robot know what motors to turn at specific times, whether that be on a curve or on a sharp turn. From our readings, we realized that the readings from the tape were low in magnitude as compared to the ground readings, which were higher. Finding out the readings from both sensors, we were able to modify and start implementing our logic with our controller.

Writing The Script

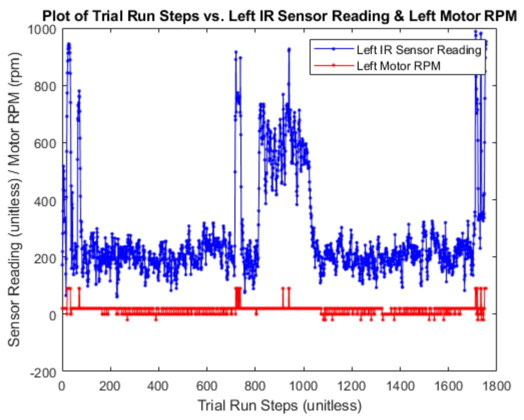
Once we were able to determine a rough range for the IR sensor readings on the surface of the floor against the surface of the electrical tape path, we used those constants to come up with our controller code. We decided to opt for a more intuitive algorithm to guide the robot, which consisted of using the IR sensor data to determine the rpm of the respective motor. By utilizing two sensors spaced just far enough for the path to fit in between with a little bit of wiggle room, we could correct the robot's trajectory when one of the two sensors picked up on the tape path. If the sensor detected that the robot was steering off course (by the sensor reading being above the certain threshold) the motor's rpm on the opposite side of the respective sensor would be set to 0 to perform a turn. If the sensor detected that the robot was back on course (by the sensor reading being within the threshold) the motor's rpm on the opposite side of the respective sensor would be set back to the regular speed to return to moving straight. If the sensor detected that the robot was really steering off course (by the sensor reading being especially above the threshold) the motor's rpm on the opposite side of the respective sensor would be put in reverse, and the motor's rpm on the same side of the sensor would be increased to perform a sharp turn with additional traction. Below in **Figure 4.0** is a snippet of the code behind this algorithm.



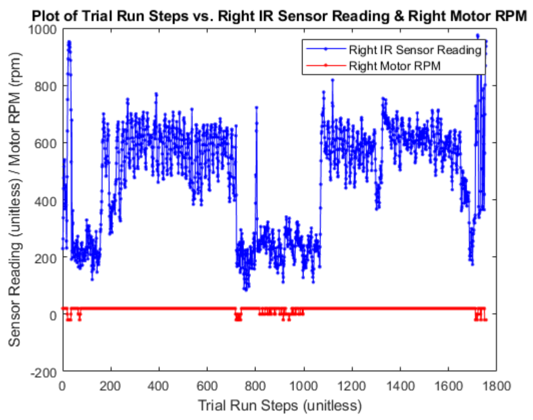
```
if (sensorValueRightMotor < 600) { // if the tape is not detected move normally (backward = forward and vice versa)
  leftMotor->setSpeed(20);
  rpmLeft = 20;
  leftMotor->run(BACKWARD);
} else if (sensorValueRightMotor > 600 && sensorValueRightMotor < 750 ) { // if the tape is somewhat detected stop moving so ot
  leftMotor->run(RELEASE);
  rpmLeft = 0;
} else { // if the tape is heavily detected move in reverse and increase the rpm on the other wheel to perform a sharp turn wit
  leftMotor->run(FORWARD);
  rpmLeft = -20;
  rightMotor->setSpeed(90);
  rightMotor->run(BACKWARD);
  rpmRight = 90;
}
```

Figure 4.0 - Snippet of Controller Code.

Once we had the code working we then uploaded it to the Arduino and tried it out on a section of the course. Below in **Figure 4.1** are the plots of the left and right motor sensor readings and motor RPM's.



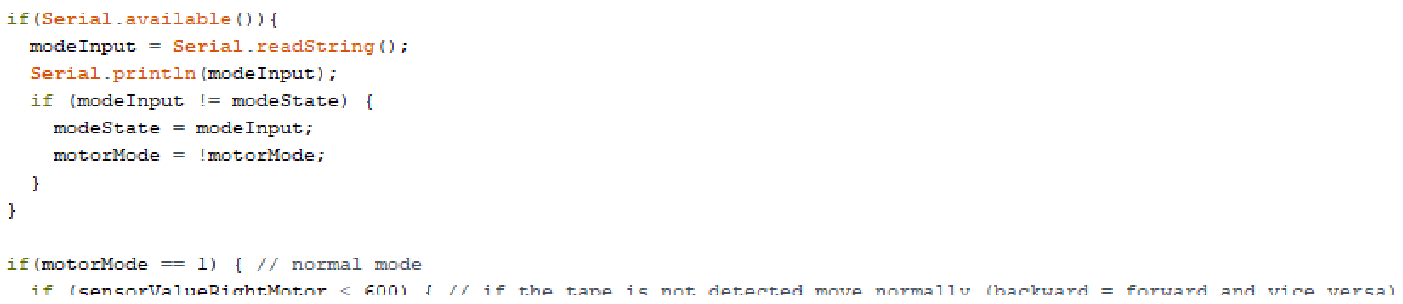
(a) Left Sensor/Motor plot.



(b) Right Sensor/Motor plot.

Figure 4.1 - Superimposed Plots from Test Drive.

With the confidence in knowing that our robot would react appropriately to all of the twits and turns of the path, we then went back to the code and added in the ability to interface with the controller in real time though the serial connection. To do this we used some of the built-in Arduino functions like `Serial.available()` and `Serial.readString()` to send data from the serial monitor input box to the flashed script on the Arduino. In doing this we created a flip-flop like module that allows for any received input to switch the robot from normal mode to "hellcat" mode, where all the RPM's were significantly higher. This can be seen below in **Figure 4.2** where we utilized plenty of `if` statements to implement our logic.



```
if(Serial.available()){
  modeInput = Serial.readString();
  Serial.println(modeInput);
  if (modeInput != modeState) {
    modeState = modeInput;
    motorMode = !motorMode;
  }
}

if(motorMode == 1) { // normal mode
  if (sensorValueRightMotor < 600) { // if the tape is not detected move normally (backward = forward and vice versa)
```

Figure 4.2 - Snippet of Mode Switch Logic.

Results

Our working robot can be seen in **Figure 5.0**. We added rubber bands on the tires to increase traction with the slippery ground. The aforementioned electrical tape path that the robot had to complete can be seen in **Figure 5.1**. A video of our robot can be seen in this video (long link https://youtu.be/YLsDAxX3CIs).

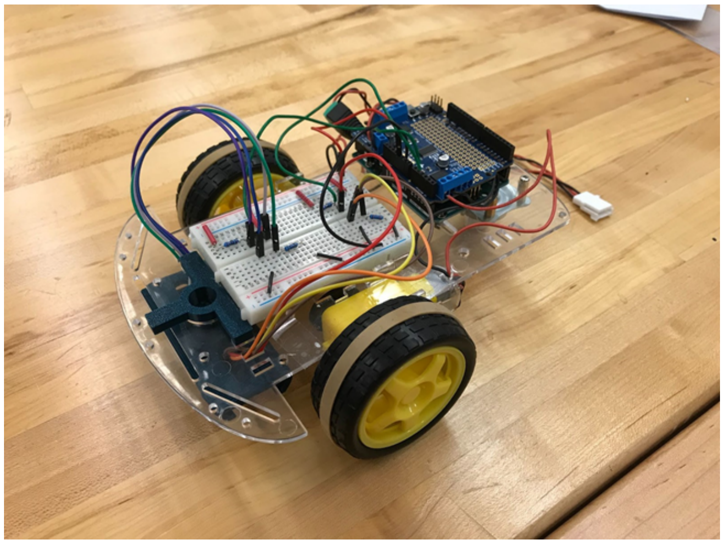


Figure 5.0 - The Working Robot.

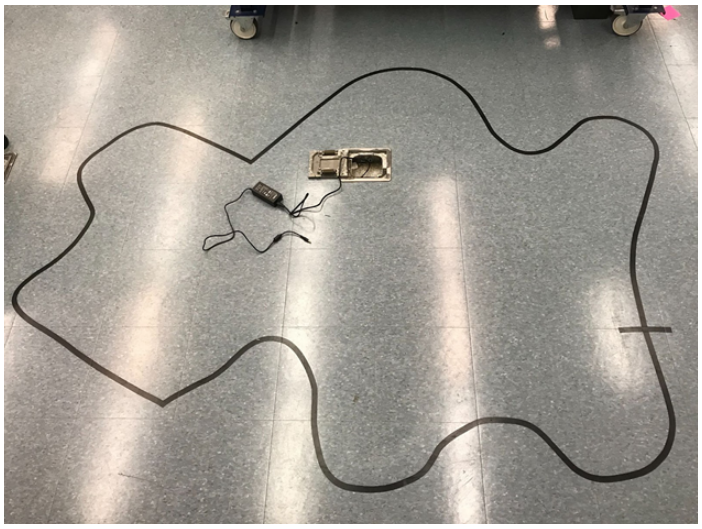


Figure 5.1 - The Electrical Tape Path.

Reflections

Aaron: I liked working on this project a lot. Since we had some people on our team who are ECE and software oriented, I got the opportunity to explore different sensor mount designs. I eventually landed on making a dovetail after getting trained on the lathe and seeing how it mounts the cutting blade. I found this design cool and interesting because of how modular it is. It saves time in reprinting the mount when we run into issues.

Leo: This mini project felt like a transition for what I could expect to do in the final PIE project. There was a lot of potential for me to do things, but I did not have the time to invest in this project. However, I definitely saw the potential to do more which may act as a fuel for me to do more in the final PIE project. For this project, I took a more passive role. I mainly focused on making the CAD for the prebuilt parts given in order to make mounts for the sensors.

Walter: It was an interesting experience to work specifically on the software for this project. I felt like I contributed a great amount to Tolu's work by helping him with smaller, modular code fixes, and I felt a great team dynamic with him on the team, which I know will lead into a great PIE final team. I am very proud of the trouble shooting that we were able to do on this robot, and it was nice to do it with another software guy.

Tolu: In this project I felt as though it was less straight coding/circuitry work and more problem solving and figuring out the best way to approach a problem before necessarily building. I found this to be refreshing and saw the challenge presented in this mini-project as something that I have never really seen before. Thanks to our overall team dynamic and bouncing algorithm ideas with Walter we were able to implement a robot that not only successfully traversed the path, but a robot that was operational in two different modes.

References

- Using DC Motors with Adafruit
  `https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino/using-dc-motors`
- IR Reflectance Sensors
  `https://www.vishay.com/docs/83760/tcrt5000.pdf`
- Helpful Chassis Construction YouTube Video
  `https://www.youtube.com/watch?v=XDAOUo7Mhr4`