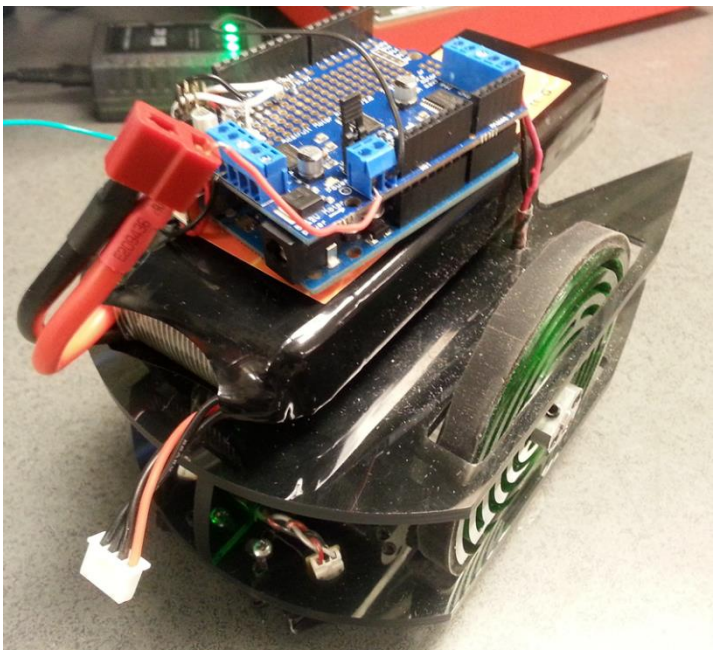
THE SHOW

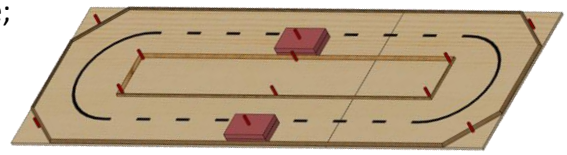
Josh Wilkins, Felisa Sze
Ahmad Hussien, Salwan Omar



MAY 20, 2014
MONROE COMMUNITY COLLEGE

Executive Summary

Design challenges are essential for engineering students throughout their education. In this instance, our team had to find and create the optimal design for a race. This race consists of a rectangular track, on which four trials take place; the first and second are without obstacles, while the third and fourth are with obstacles in different configurations. All four trials are comprised of five laps each, the first two with a time constraint of 60 seconds each, meanwhile the latter two have a time constraint of 75 seconds each.



From the initiation to the termination of the creation of our design, our team encountered a few problems. Initially, ultrasonic sensors seemed to be the most logical solution to the design problem. We thought we could use the distance from the robot to the wall to steer it, remaining a constant distance from the wall. With a little research however, the ultrasound sensor was deemed less fit than an infrared sensor. Not only would ambient noise in the room affect the ultrasonic readings, but the infrared sensors are less affected by differing materials. An alternative idea we had was to force the robot against a wall and use touch sensors to detect if it was against the wall. Due to the friction caused by this design, a lot of speed would be lost, making this option better than the ultrasound, but not optimal. The optimal choice we then believed was the use of two infrared sensors. Evidently, through various and conclusive results and research, our design is the optimal choice from the deficiencies of alternative options.

Looking back at the project, one can see that we were the team to beat for a while. Having been the first team to complete each robot testing, well in advance of their due dates, shows that we are one of the best and most successful teams. The total cost of the robot came out to be an astounding $296.06, a small price compared to the cost of labor at $11474. In conjunction with the high cost of labor, the sustainability analysis shows harmful results. With a high energy consumption and waste output, this robot is not suited for mass production, but with a few changes could become something more suited for the market. Overall, the design of the robot is adequate and can be easily changed for its own improvement.

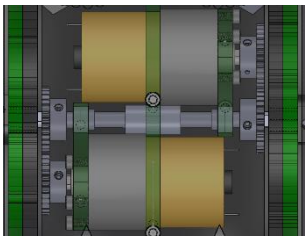
Table of Contents

Design Evolution

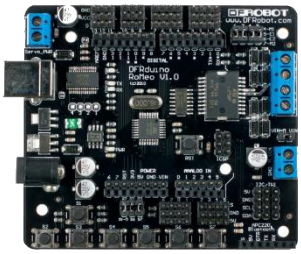
In the very beginning, our team decided that wall following, through the use of some arrangement of IR sensors, would be the optimal way to navigate. The reason being is that the lines on the track don't lend themselves to an easy navigation system. Not only are the lines on the track dashed on the straightaways, but they are only in the middle of the track, limiting their usefulness for obstacle avoidance. Alternative methods aside from line or wall following methods had not been thought of at this point.

This brought us to our next decision. What type of sensor were we going to use for wall following, infrared sensors or ultrasound sensors? Infrared sensors were chosen for numerous reasons. Firstly, they have narrow sight, which means a lower chance of detecting stray dowels. Furthermore, they are faster than ultrasonic sensors, they are unaffected by ambient noise in the room, and differing materials won't affect an IR sensor. Initially, only one short range sensor was used to follow the inner wall, but as our strategy developed, the use of two sensors became essential. After many trials and attempts, with limited success, we then switched to longer range IR sensors.

The propulsion system on our robot was created with simplicity in mind. To drive and steer the robot, our team decided on two separate drive motors, each connected through a gear train to the wheels. This way, turning requires that each motor move at difference speeds. The alternative was to have four wheels; the front two controlled by a servo motor and the rear two powered by a motor. The servo motor would control the direction the front wheels point, allowing the robot to turn; however, not only does this create a whole new system just for steering, but in order for the wheels to turn correctly, without slippage, the use of a differential would be needed in the front.



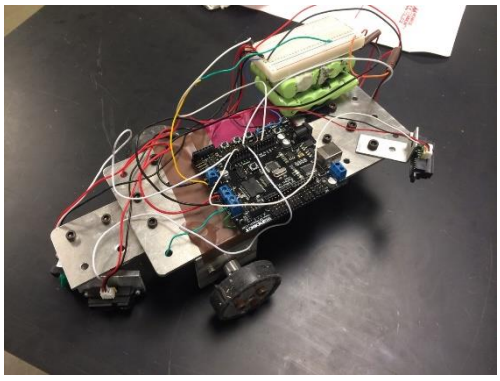
For the microcontroller, we chose to use Arduino microcontrollers as they are easy to learn and have plenty of libraries and examples for them. Initially, the Arduino Romeo microcontroller was used, an all-in-one Arduino compatible microcontroller. This Arduino microcontroller was chosen for the simple fact that it had motor drivers built in, meaning that a separate motor shield was unnecessary. However, we eventually switched to an Arduino Uno, with an Adafruit motorshield, as the Romeo was giving us communication errors. The benefits of the Arduino Uno included being smaller than the Romeo and having direct access to a ProtoBaord included on the Adafruit



motorshield. This ProtoBoard enabled us to solder wires directly onto it, improving the neatness and the reliability of the robot.

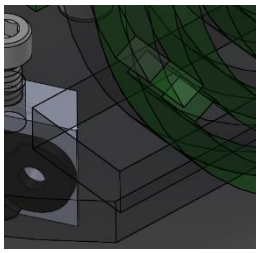
Shortly after the brainstorming and decision making process, we chose to have two robots: a prototype purely for testing and learning the Arduino code, and a final model for all the official tests and competitions. The prototype was built within the week using materials from a previous course. Throughout the use of the prototype, the Romeo was used as the microcontroller. The prototype had four IR sensors: two angled in the front to follow either the inner or outer wall, one in the back for initial detection of the block, and one facing inwards on the back left corner, to see the position of the robot in relation to the inner block.

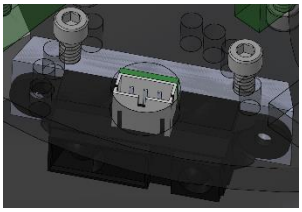
A PID (Proportional-Integral-Derivative) library for Arduino was used to wall follow. This is an algorithm that uses the sensors as inputs, to control the motor speeds (output). In



conjunction with this control system, we used our initial idea for Trials 3 and 4; passively avoiding the blocks. By starting in the inner lane, the rear sensor was used to determine the configuration, then corners were counted to control which lane the robot should be in to avoid the block. For example, the track is in the Tails configuration. The robot would count four corners then switch to the outer lane to avoid the oncoming inner lane block. Afterwards, the side sensor would see the block and the robot would then know to switch back to the inner lane again. This process would be looped to passively avoid the blocks. With the use of our prototype, we were able to complete the three assigned tests well ahead of their respective deadlines, and were the first to do so in each case.

During all this testing of the prototype, our new robot was being designed and built. It was made smaller, lighter, and faster than the prototype robot, with larger wheels, new motors and a better battery. Multiple issues that were discovered with the prototype were fixed on the new robot; the first problem was that the prototype robot would climb the quarter round on the track. To fix this, the addition of wheel guards and bumpers were added on the new design. Other problems included the voltage drain on the batteries and the maximum achievable speeds of the prototype. To further increase the speed, new motors were bought, motors with both higher rpm and torque ratings. In addition, gears were



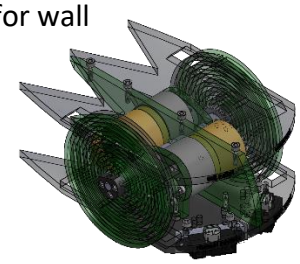


added in a 2:1 ratio fr om the motor to the wheel shaft. To fix the drain on the battery, a lipo battery was purchased for the new robot. These batteries last longer and are able to provide a more consistent rate of discharge to the motors, further increasing the consistency and reliability of the robot. Furthermore, an adjustable mounting system for the front sensors was added to the new design, allowing for the ability to change their respective angles.

```
void sharpTurn() {
  int j=0;
  if (input < 100) {
    analogWrite(E2, setpoint);
    analogWrite(E1, setpoint);
    delay(200);

    digitalWrite(M2, LOW);
    analogWrite(E2, 120);
    analogWrite(E1, 230);

    time1 = millis();
    while (j < 40) {
      input = analogRead(analogPin);
      if (input > 110)
      {
        j++;
      }
    }
    time2 = millis();
    if (time2-time1 > 157)
      turnCount++;
    digitalWrite(M2,HIGH);
  }
}
```

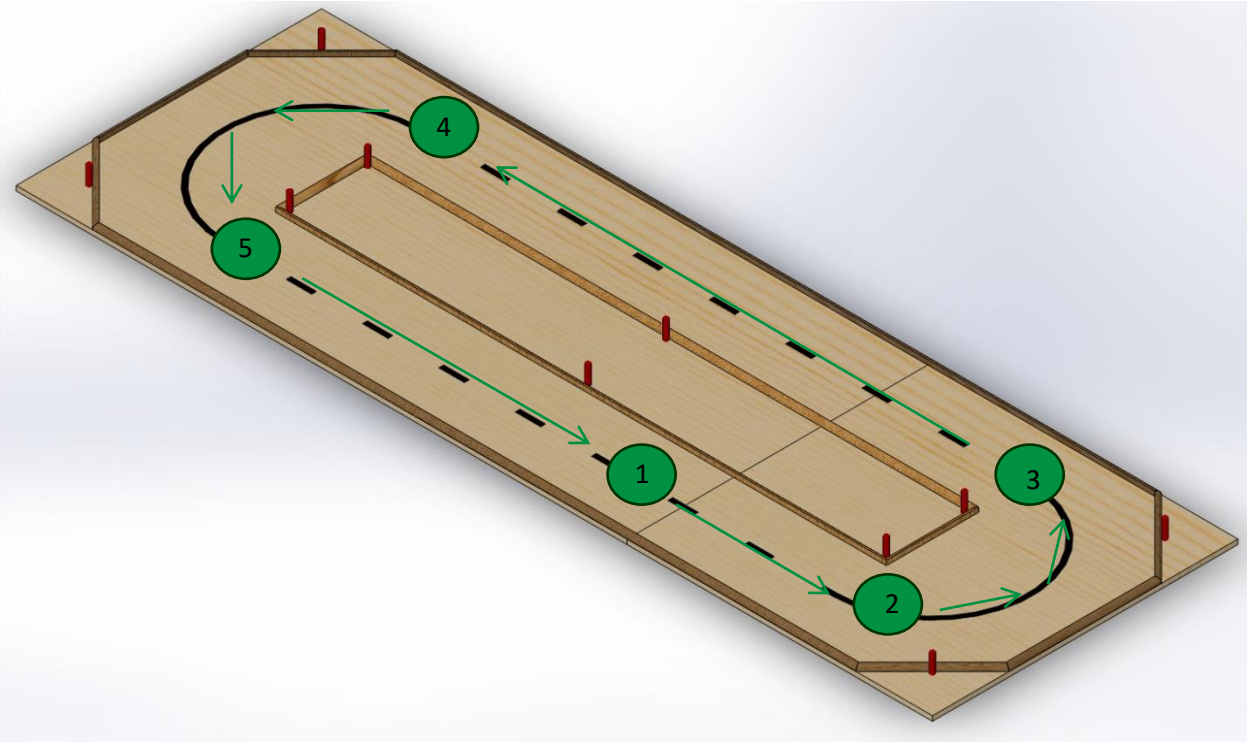
With the final robot assembled, the first logical step was to attempt using the code developed for the prototype. Despite attempts at tweaking the code, nothing we did made our developed code compatible with the new robot. It was at this point that we realized counting corners would not work with the new robot's shape and speed. This led us to an alternative form of the PID algorithm, just without the library.

In the end, the robot used three sensors in total: two in the front for wall following and one, front facing, in the upper left. The front two, unlike the prototype, were used in tandem to determine the position of the robot. This setup was used, mainly because of the observed success many other teams had with a similar setup. Our new algorithm for wall following took into consideration, both sensor inputs, from which we could position the robot in any lane on the track. Due to the fact that corner counting would no longer work, we made the robot travel in the inner lane. This way, only one block would have to be actively avoided. The outer block would be passed without a problem, but the inner block would now have to be seen and bypassed. The upper left IR sensor was used to detect this block and once this occurred, the robot would switch to the outer lane. From this point the robot would pass the block and then transition briefly to the middle of the track (to avoid major oscillations), and then back to the inner lane. Our final solution was the result of lots of planning, testing, mistakes, errors, and determination, all of which led to a better, more ideal solution to the design challenge.



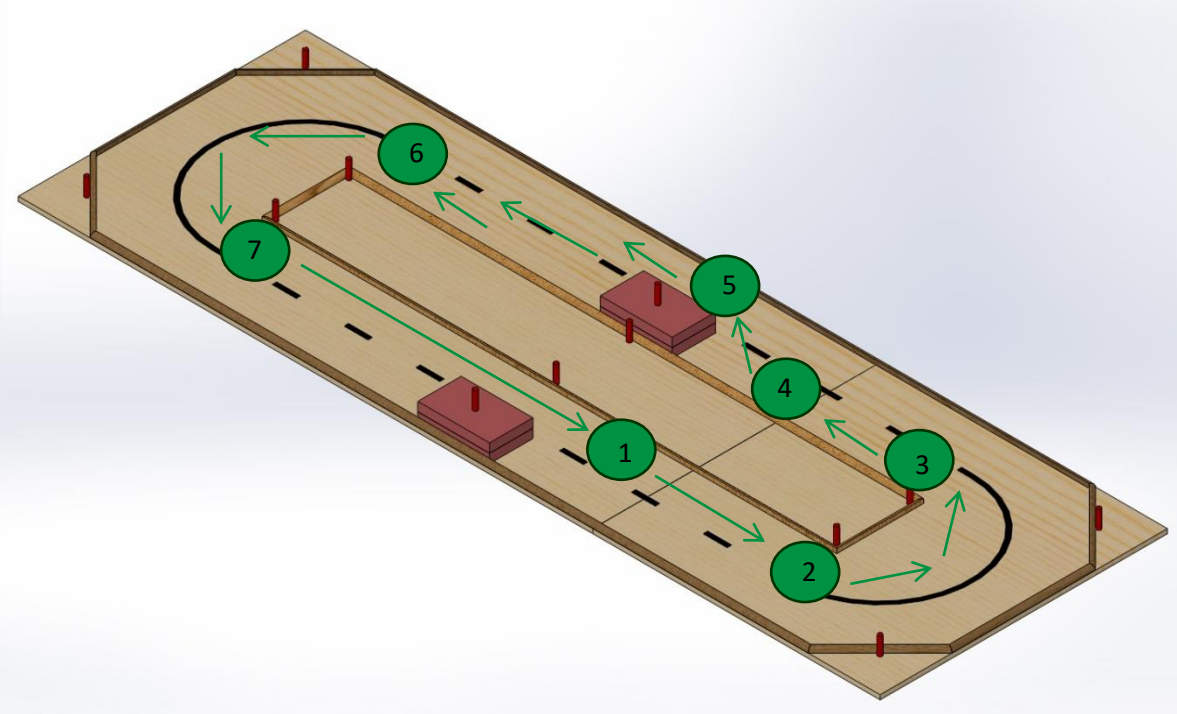
Robot Operation

Trials 1 and 2:



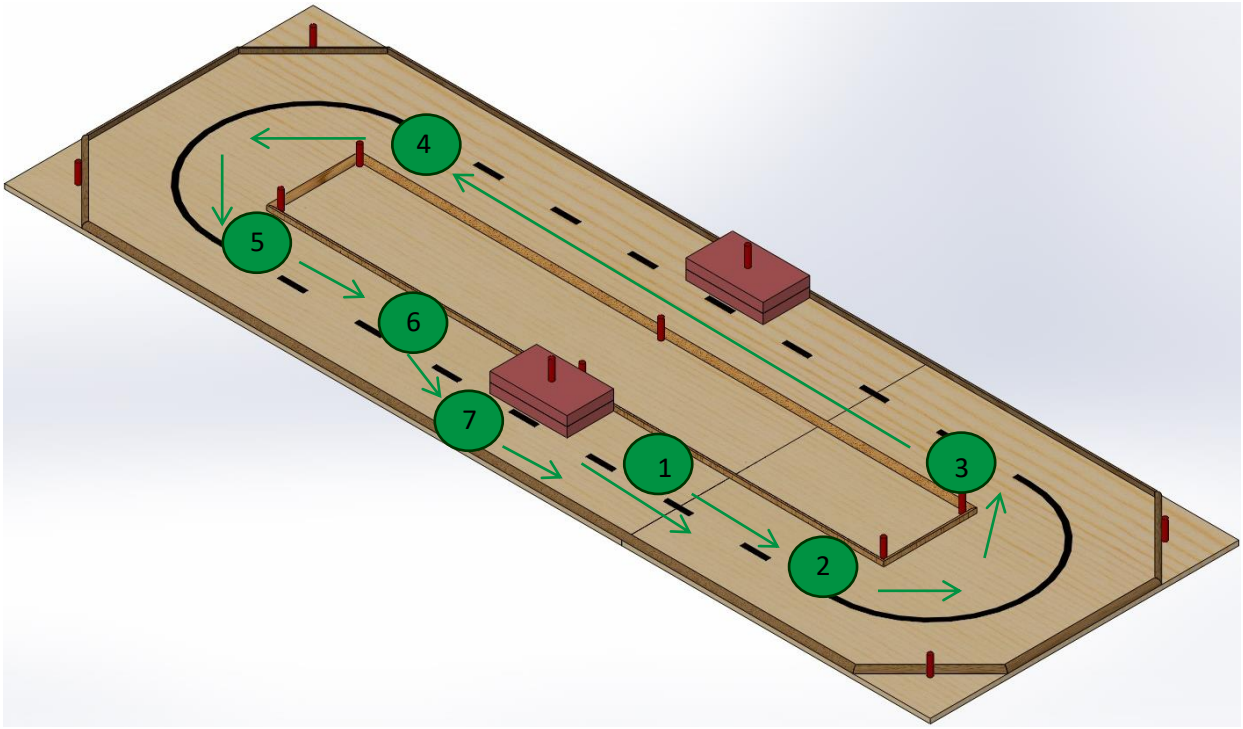
1. The robot starts behind the starting line, in the middle of the track. The switch is turned on, and the microcontroller is powered on.
2. The robot drives forward, following both the left and right walls until the front left sensor detects the drop-off in value that indicates a corner in the inner lane. The robot then slows down and changes its set points to make it around the corners safely.
3. The robot sees the inner wall for the straightaway, and drives forward.
4. The robot sees a corner again and changes its settings appropriately.
5. The robot sees the inner wall again, and drives forward. At position 1, it completes a lap and continues this process until it's turned off.

Trials 3 and 4 (Heads Configuration):



1. The robot starts behind the starting line, in the inner lane of the track. The switch is turned on, and the microcontroller is powered on.
2. The robot drives forward, following both the left and right walls until the front left sensor detects the drop-off in value that indicates a corner in the inner lane. The robot slows down and changes its set points to make it around the corners safely.
3. The robot sees the inner wall for the straightaway, and drives forward.
4. The top left sensor on the robot detects the inner lane block, and the robot changes its set points moving it to the outer lane.
5. The robot realizes it's passing a block, slows down, and changes its set points. After it finishes passing the block, the robot moves to the middle of the track for 800 milliseconds, and then goes back to the inner lane.
6. The robot drives forward until it sees a corner again and changes its settings appropriately.
7. The robot sees the inner wall again and drives forward, passing by the outer block. At position 1, it completes a lap and continues this process until it's turned off.

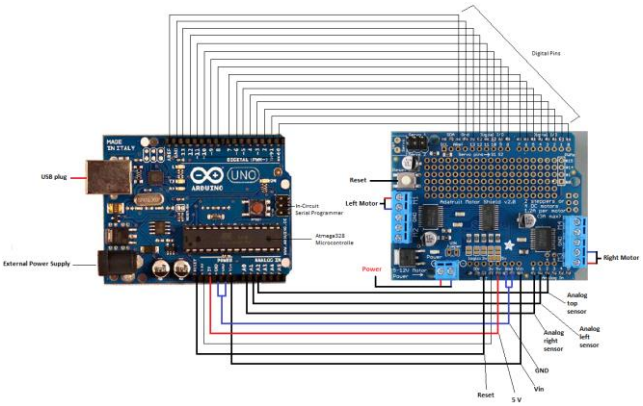
Trials 3 and 4 (Tails Configuration):



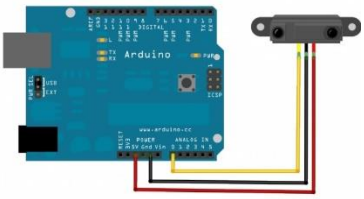
1. The robot starts behind the starting line, in the inner lane of the track. The switch is turned on, and the microcontroller is powered on.
2. The robot drives forward, following both the left and right walls until the front left sensor detects the drop-off in value that indicates a corner in the inner lane. The robot slows down and changes its set points to make it around the corners safely.
3. The robot sees the inner wall for the straightaway, and drives forward, passing the outer block.
4. The robot drives forward until it sees a corner again and changes its settings appropriately.
5. The robot sees the inner wall again and drives forward.
6. The top left sensor on the robot detects the inner lane block, and the robot changes its set points moving it to the outer lane.
7. The robot realizes it's passing a block, slows down, and changes its set points. After it finishes passing the block, the robot moves to the middle of the track for 800 milliseconds, and then goes back to the inner lane, where it goes into position 2, completing a lap. The robot continues this process until it's turned off.

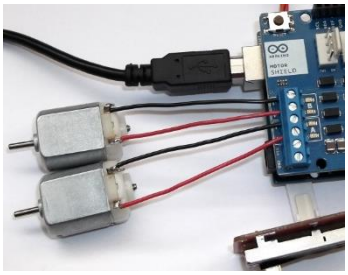
Electronics & Software

Circuit Analysis

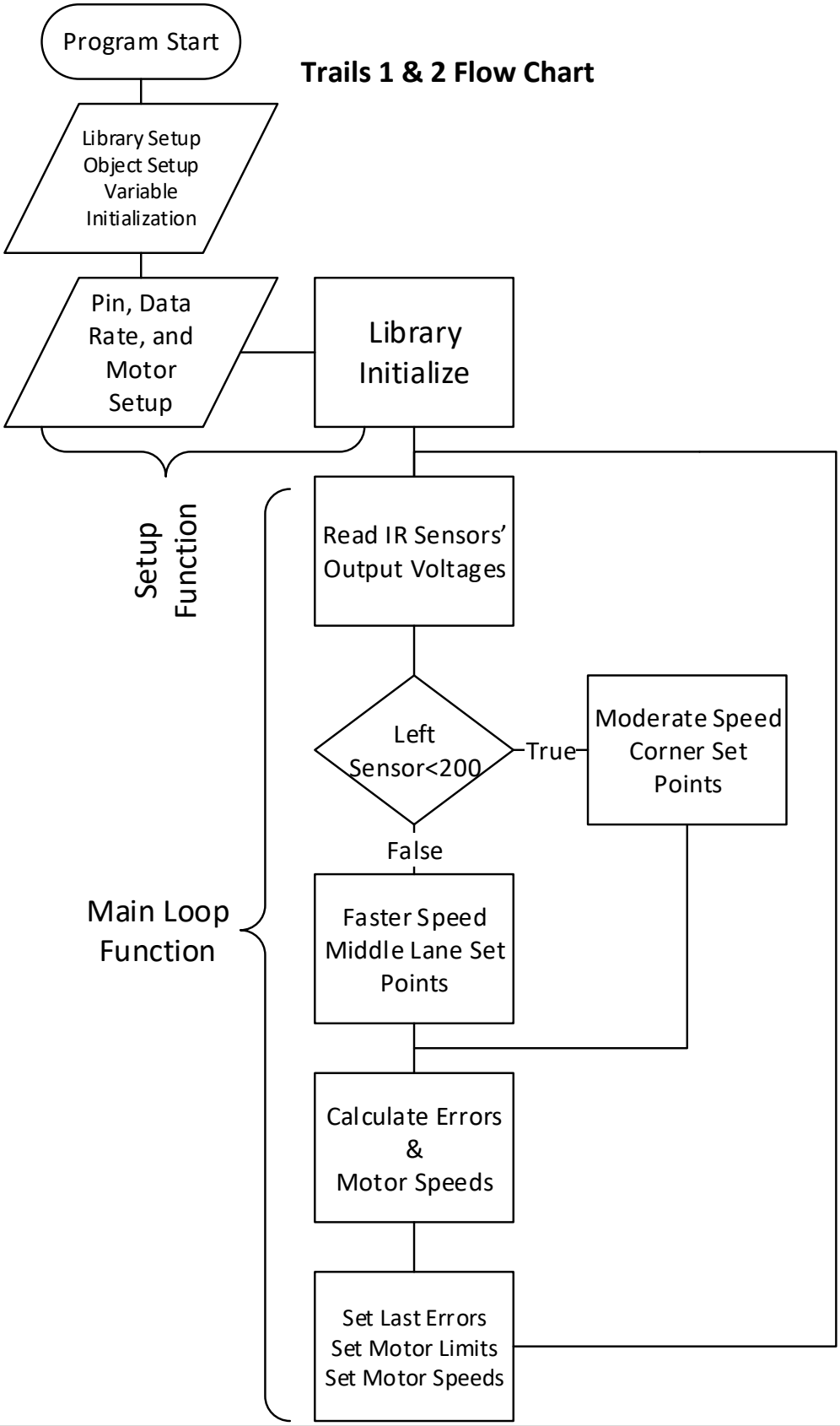


In order to fully understand the circuitry of our design, one must first look at the setup and connections of the microcontroller to the motor shield. As you can see, most of the pins on the Arduino Uno directly correspond to pins on the motor shield, however some new features are added with the addition of a motor shield. These features include a new input source for power, motor ports and a reset button. In addition to this, our circuitry involves 3 IR sensors, each connected to the motor shield as depicted in this image. Although the image shows an Arduino Uno, the wires would connect to the corresponding motor shield pins. Each sensors' ground wires can be wound together, then each of their power wires can also be wound together, then they can be put into their pins for the easiest layout. The third wire is then attached to three unique analog pins. The battery wires are connected in series with a switch and into the pins labeled power in the image above. The motors are the last aspect of our circuit and are connected similarly to the last image here, except on opposing sides of the motor shield. Each component has, with the exception of the IR sensors, two wires each. These wires are the ground wire and the power wire. For the motors, it supplies a voltage difference across it, providing the energy needed to move it, meanwhile the battery has a similar concept except it is what provides the voltage difference to each component. The sensors, in addition to these two wires, have a third wire for the measurement of infrared light reflected back to the sensor, measured in voltage representation. The combination of all these components, connected in a circuit and in conjunction with the Arduino Uno microcontroller, create our team's circuit.





Trails 1 & 2 Flow Chart

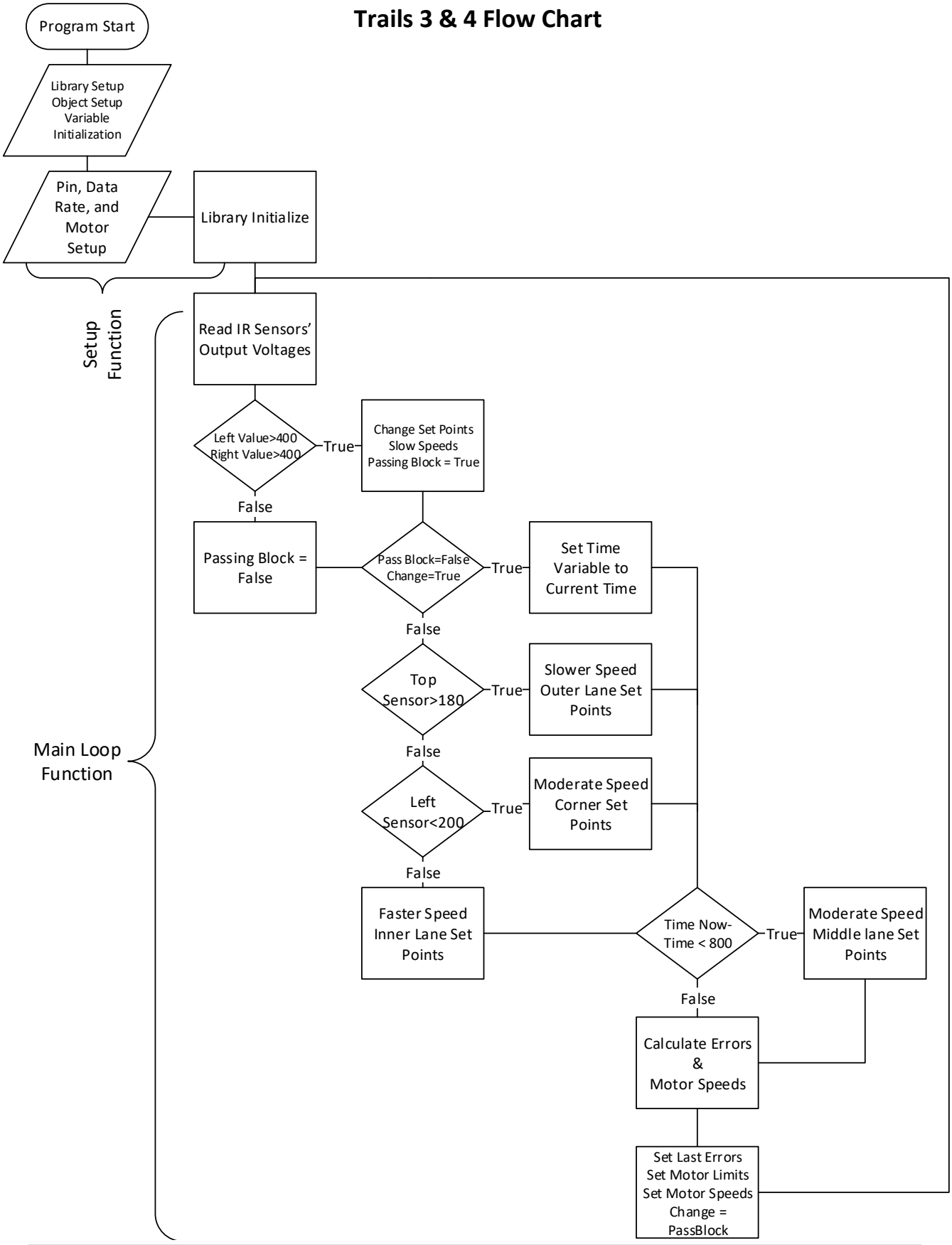


Trials 1 & 2 – Program Code

```
// Robot will traverse in the center of the track

// Libraries
#include "utility/Adafruit_PWMServoDriver.h"
#include <Adafruit_MotorShield.h>
#include <Wire.h>

// Objects (for the motors)
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
Adafruit_DCMotor *RWheel = AFMS.getMotor(3);
Adafruit_DCMotor *LWheel = AFMS.getMotor(1);

int LeftSensor = A1, RightSensor = A0;      // Pins of the sensors- only the front two
int Speed = 110;                            // Base speed
int rightValue, leftValue;                  // The sensor reading values
double errorright, errorleft;               // Error = sensor reading - setpoint
double distance1, distance2;                // Setpoints - Ideal distances away from walls
double distanceR = 280, distanceL=270;  // The setpoints for the middle of the track
double speeda, speedb;                      // Left and right speeds to motors
double k=.05, kd=7;                         // PID values - Multiply the error and change in error by
double lasterrorright=0, lasterrorleft=0;     // Previous values of error

// Function that runs once in the beginning, for set up - pin modes and motor directions
void setup()
{
  // Make the sensor pins inputs
  pinMode(LeftSensor, INPUT);
  pinMode(RightSensor, INPUT);

  Serial.begin(9600);
  AFMS.begin();

  // Rotate wheels forward
  RWheel->run(BACKWARD);
  LWheel->run(FORWARD);
 }
```

```
// A function that loops continuously until the Arduino is reset or turned off.
void loop()
{
 // Read and record sensor values
 rightValue = analogRead(RightSensor);
 leftValue = analogRead(LeftSensor);

 // If the robot sees the dropoff for a corner, slow down and change setpoints
 if (leftValue < 200) {
  Speed = 85;
  distance1 = 200;
  distance2 = 600;
 }
 // Otherwise go straight in the center of the track
 else {
  Speed= 160;
  distance1 = distanceR;
  distance2 = distanceL;
 }

 // Error based on the difference between desired distances and actual ones
 errorright = (rightValue - distance1);  //positive= too close
 errorleft = (leftValue - distance2);    //negative= too far

// Motor Speeds Calculated from both the errors and the change in errors of both sensors
 speeda= -k*errorright -kd*(errorright-lasterrorright)+ k*errorleft + kd*(errorleft-
         lasterrorleft)+Speed;
 speedb= -k*errorleft-kd*(errorleft-lasterrorleft) + k*errorright + kd*(errorright-
         lasterrorright)+Speed;

 // Record error values for the next loop
 lasterrorright=errorright;
 lasterrorleft=errorleft;

 // Limit the left and right speeds so the robot doesn't go too fast or not at all
 if(speeda>200) speeda=200;
 if(speedb>220) speedb=220;       // Upper limits are different due to motor differences
 if(speeda<0) speeda=0;
 if(speedb<0) speedb=0;

 // Sets the newly calculated speeds of the motors
 LWheel->setSpeed(speeda);
 RWheel->setSpeed(speedb);
}
```

Trails 3 & 4 Flow Chart



Trials 3 & 4 – Program Code

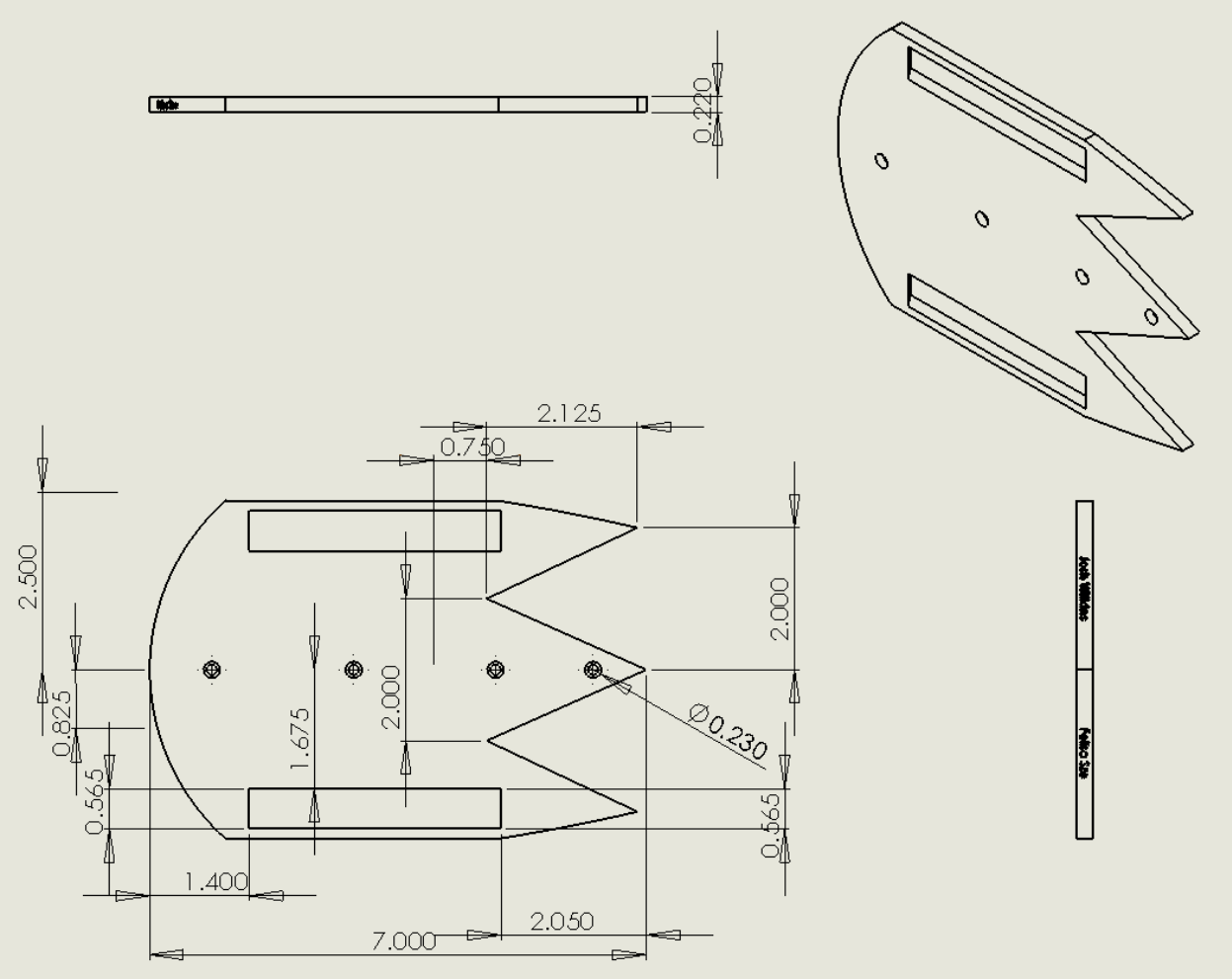
```
// Robot will follow inside lane and only actively avoid the inside block
// where it'll transition briefly to the outside lane to avoid it

// Libraries
#include "utility/Adafruit_PWMServoDriver.h"
#include <Adafruit_MotorShield.h>
#include <Wire.h>

// Objects (for the motors)
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
Adafruit_DCMotor *RWheel = AFMS.getMotor(3);
Adafruit_DCMotor *LWheel = AFMS.getMotor(1);

// Variables
int LeftSensor = A1, RightSensor = A0, TopLeftSensor = A2;  // Pins of all sensors
int Speed = 110;                          // Base speed
int rightValue, leftValue, topValue;      // The sensor reading values
double errorright, errorleft;             // Error = sensor reading - setpoint
double distance1, distance2;              // Setpoints- The ideal distances away from the walls
double speeda, speedb;                    // Left and right speeds to motors
double k=.05, kd=7;                       // PID values - Multiply the error and change in error by
double lasterrorright=0, lasterrorleft=0;          // Previous values of error
long time;                                         // A variable for a value of millis()
boolean passBlock = false;                         // Is the robot passing a block?
boolean change = false;                            // What's the previous value of passBlock?

// A function that runs only once in the beginning, to set up things like pin modes and motor
directions
void setup()
{
  // Make the sensor pins inputs
  pinMode(LeftSensor, INPUT);
  pinMode(RightSensor, INPUT);
  pinMode(TopLeftSensor, INPUT);

  Serial.begin(9600);

  AFMS.begin();

  // Wheels rotate forward
  RWheel->run(BACKWARD);
  LWheel->run(FORWARD);
 }
```

```
// A function that loops continuously until the Arduino is reset or turned off.
void loop()
{
 // Read and record sensor values
 rightValue = analogRead(RightSensor);
 leftValue = analogRead(LeftSensor);
 topValue = analogRead(TopLeftSensor);

 // When passing a block, slow down and record that you're doing so
 if ((leftValue > 400) && (rightValue > 400)) {
  Speed = 60;
  distance1 = 530;
  distance2 = 550;
  passBlock = true;
 }
 else {
  passBlock = false;
 }

 // After finishing passing a block, mark the time
 if ((passBlock == false) && (change == true)) {
  time = millis();
 }
 else {
  // If the robot sees a block, slow down and move to the outer lane
  if (topValue > 180) {
   Speed= 60;
   distance1 = 510;
   distance2 = 155;
  }
  // If the robot sees the dropoff for a corner, slow down and change setpoints
  else if (leftValue < 200) {
   Speed = 85;
   distance1 = 200;
   distance2 = 600;
  }
  // Go forward in the inner lane otherwise
  else {
   Speed= 130; //125
   distance1 = 150;
   distance2 = 490;
  }
 }
```

```
  // After passing a block, stay in the middle of the track for 800 milliseconds
  if (millis() - time < 800) {
    Speed = 80;
    distance1 = 230;
    distance2 = 250;
  }

  // Error based on the difference between desired distances and actual ones
  errorright = (rightValue - distance1);  // Positive= too close
  errorleft = (leftValue - distance2);    // Negative= too far

// Motor Speeds Calculated from both the errors and the change in errors of both sensors
speeda= -k*errorright -kd*(errorright-lasterrorright)+ k*errorleft + kd*(errorleft-
          lasterrorleft)+Speed;
  speedb= -k*errorleft-kd*(errorleft-lasterrorleft) + k*errorright + kd*(errorright-
          lasterrorright)+Speed;

  // Record values for the next loop
  lasterrorright=errorright;
  lasterrorleft=errorleft;
  change = passBlock;

  // Limit the left and right speeds so the robot doesn't go too fast or not at all
  // Upper limits are different due to motor differences
  if(speeda>200) speeda=200;
  if(speedb>220) speedb=220;
  if(speeda<0) speeda=0;
  if(speedb<0) speedb=0;

  // Set the newly calculated speeds of the motors
  LWheel->setSpeed(speeda);
  RWheel->setSpeed(speedb);
}
```

Fabrication Methods

Chassis Plate - Top



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

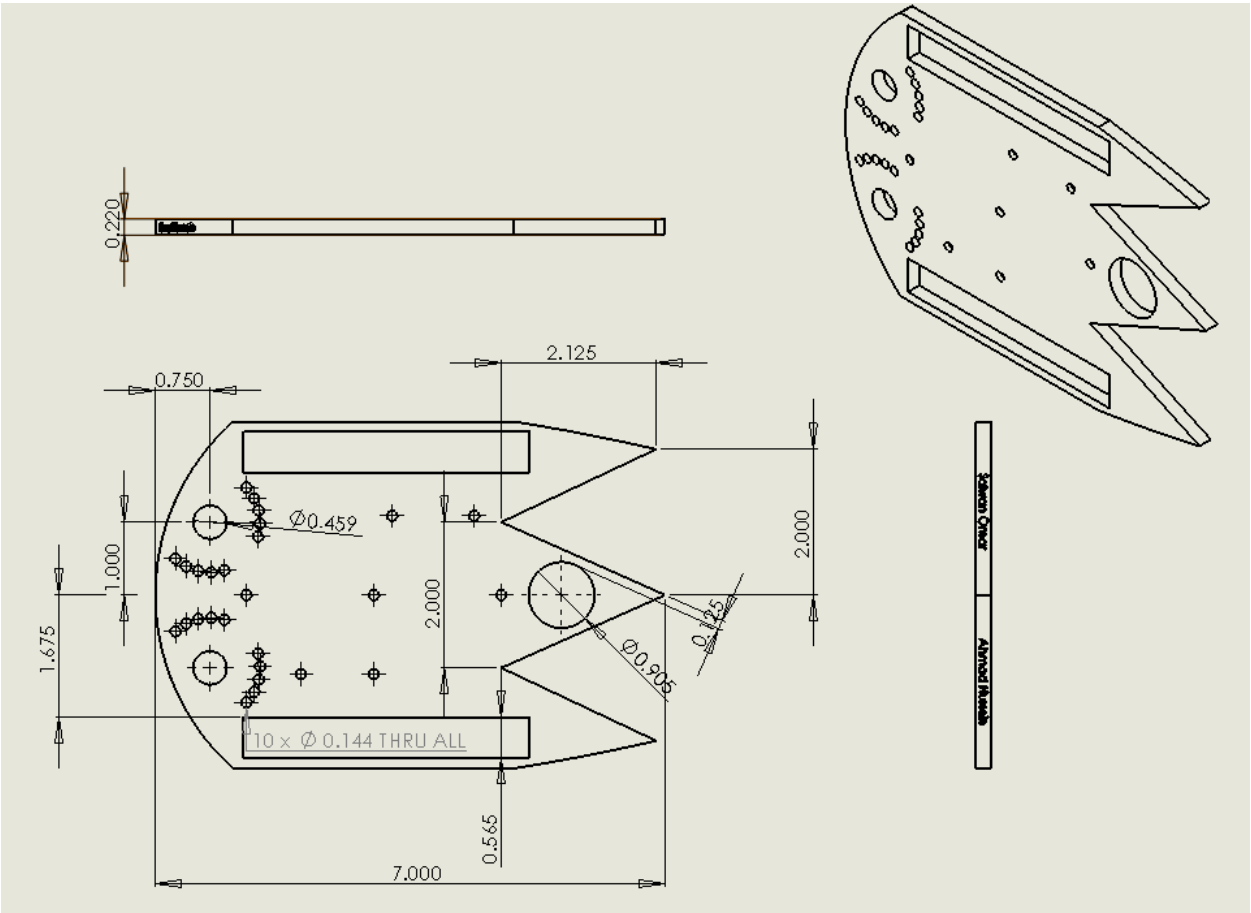
Tools used for fabrication - 1/8” End Mill, #27 Drill Bit, File, 1/8” Jobber Drill Bit

Resource used – 12” x 12” x .220” Transparent Black Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach

Chassis Plate - Bottom



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

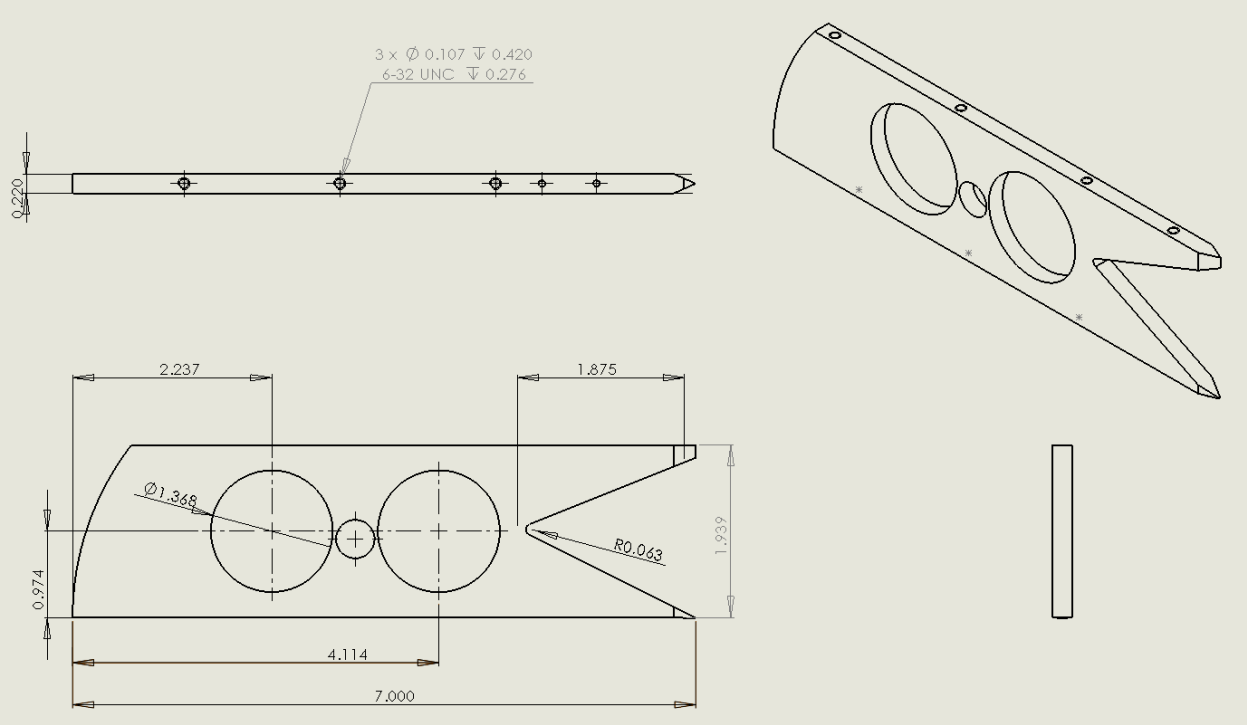
Tools used for fabrication - 1/8” End Mill, #27 Drill Bit

Resource used – 12” x 12” x .220” Transparent Black Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach

Chassis Plate – Mid Portion



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

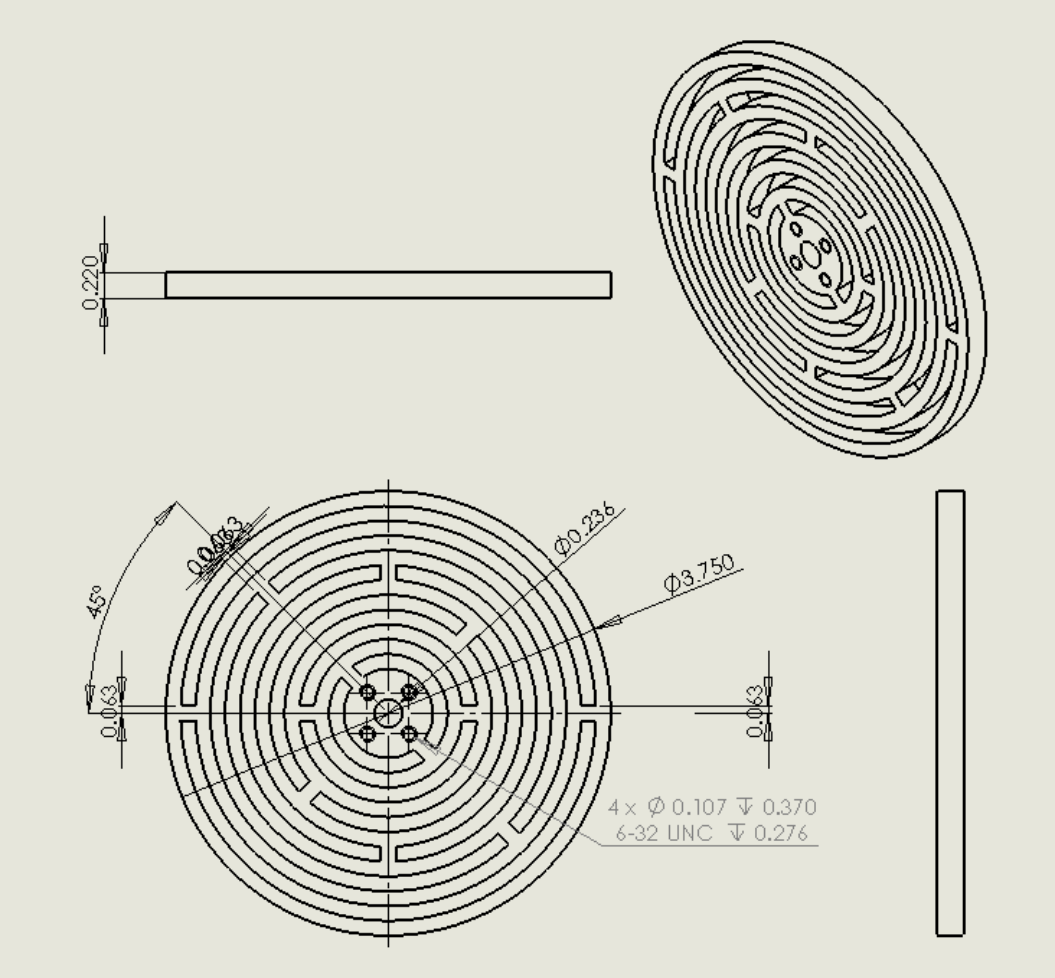
Tools used for fabrication - 1/8” End Mill, #36 Drill Bit, 6-32 Tap

Resource used - 12” x 12” x .220” Transparent Green Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach
5. Tap Holes with 6-32 Tap

Wheel



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

Tools used for fabrication - 1/8” End Mill, #27 Drill Bit

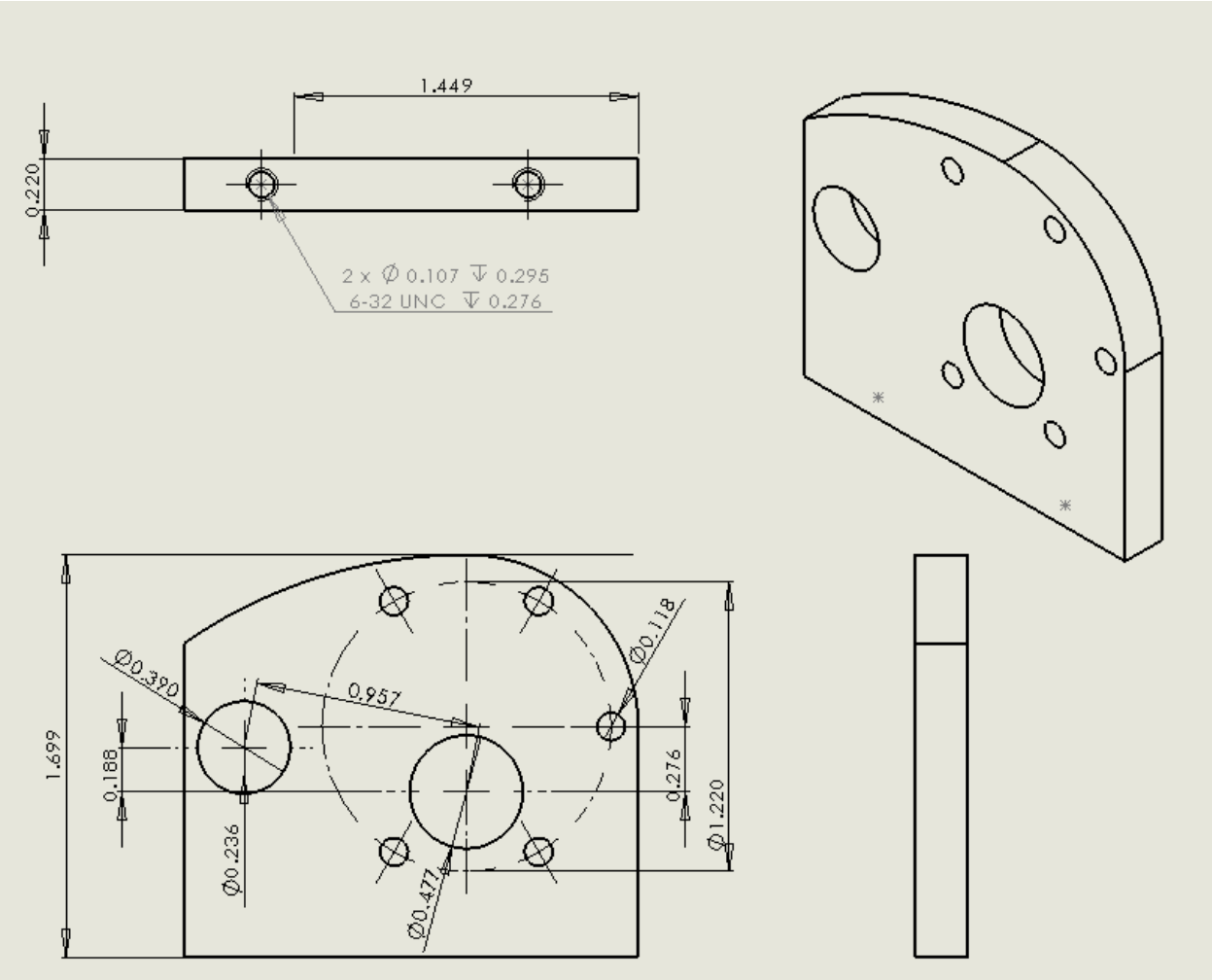
Resources used – 12” x 12” x .220” Transparent Black Acrylic Sheet

– 12” x 12” x .220” Transparent Green Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach
5. Tap Holes with 6-32 Tap

Motor Mount - Left



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

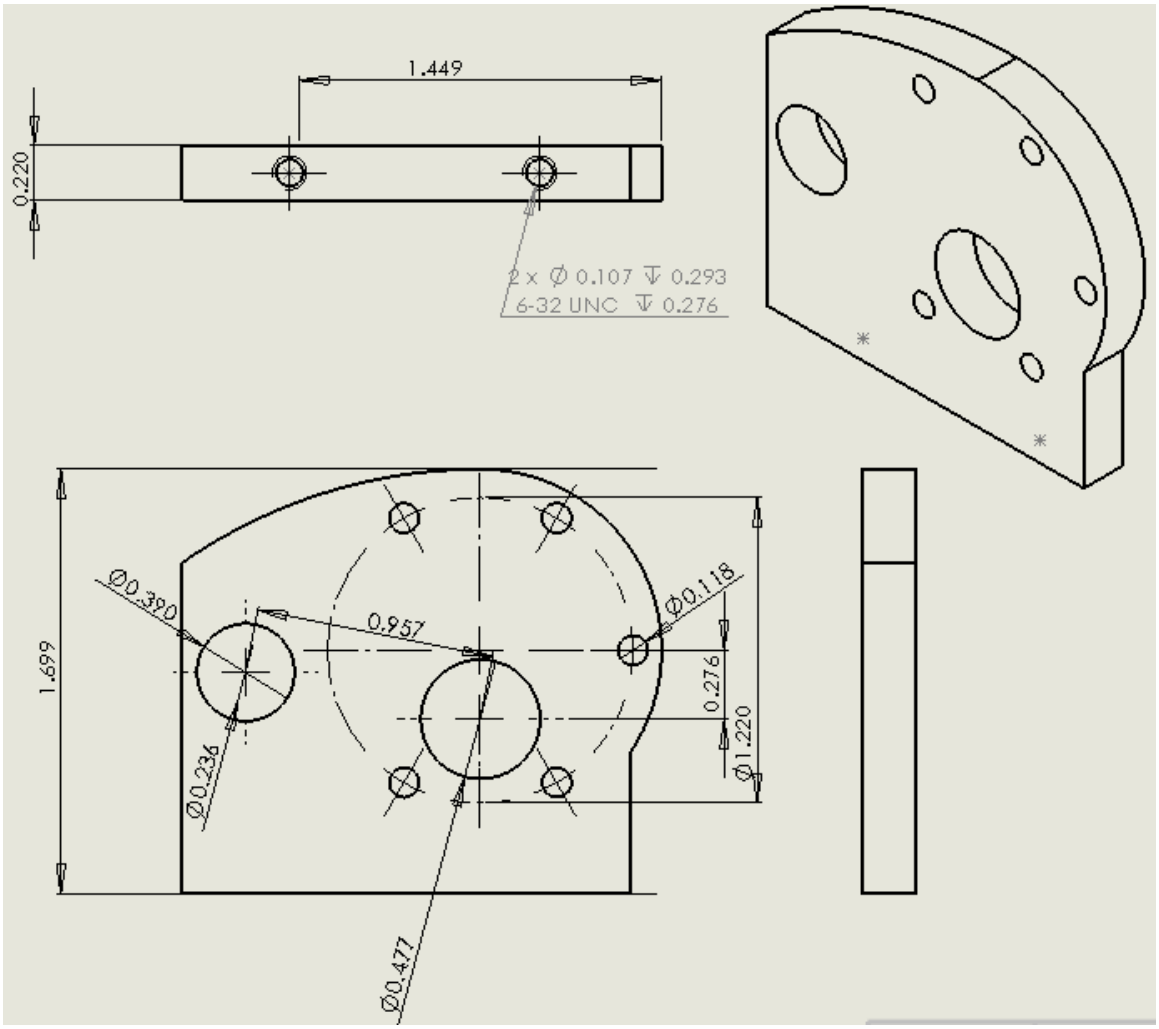
Tools used for fabrication - 1/8” End Mill, #29 Drill Bit, #36 Drill Bit, 6-32 Tap

Resource used - 12” x 12” x .220” Transparent Green Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach
5. Tap Holes with 6-32 Tap

Wheel Mount - Right



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

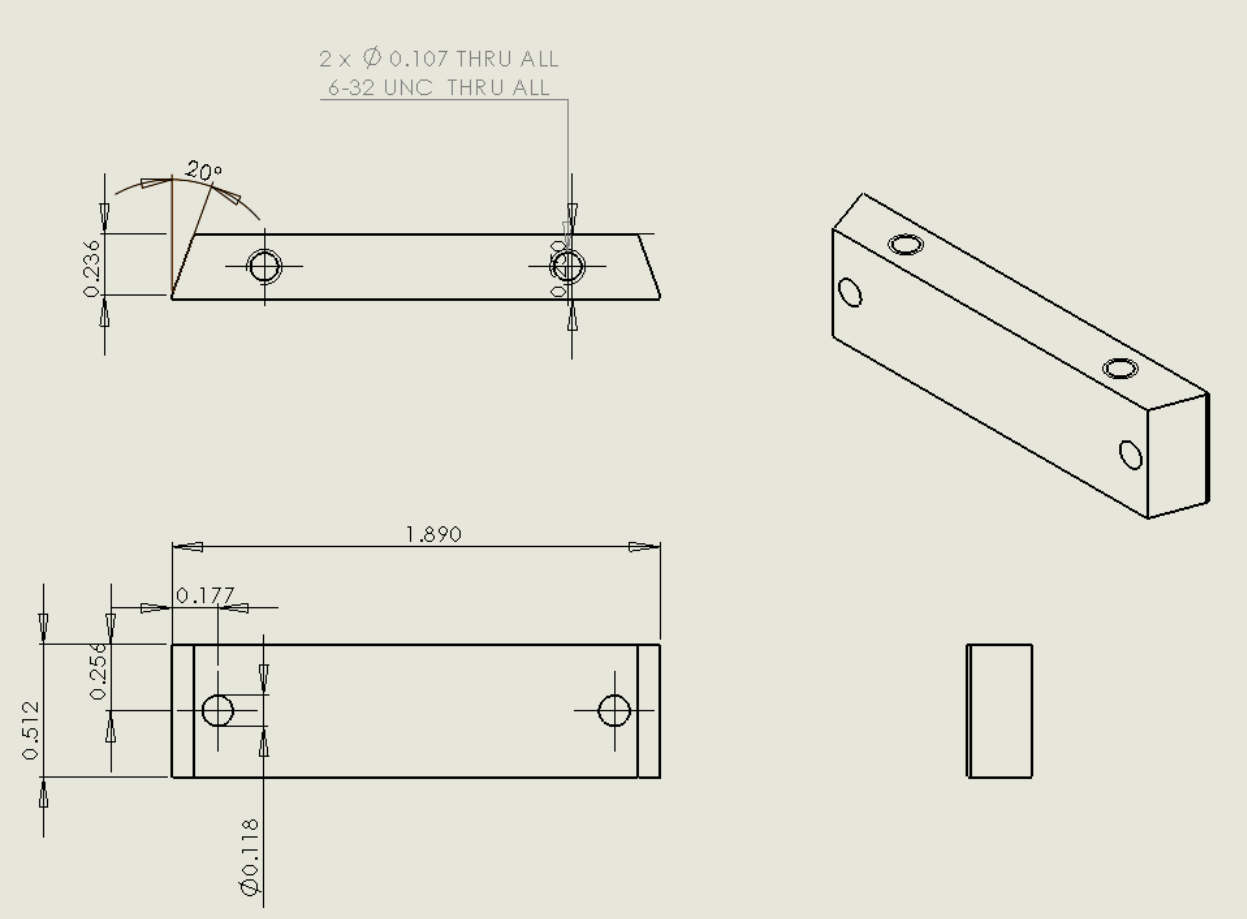
Tools used for fabrication - 1/8” End Mill, #29 Drill Bit, #36 Drill Bit, 6-32 Tap

Resource used - 12” x 12” x .220” Transparent Green Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach
5. Tap Holes with 6-32 Tap

Sensor Mount



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine, Belt Sander

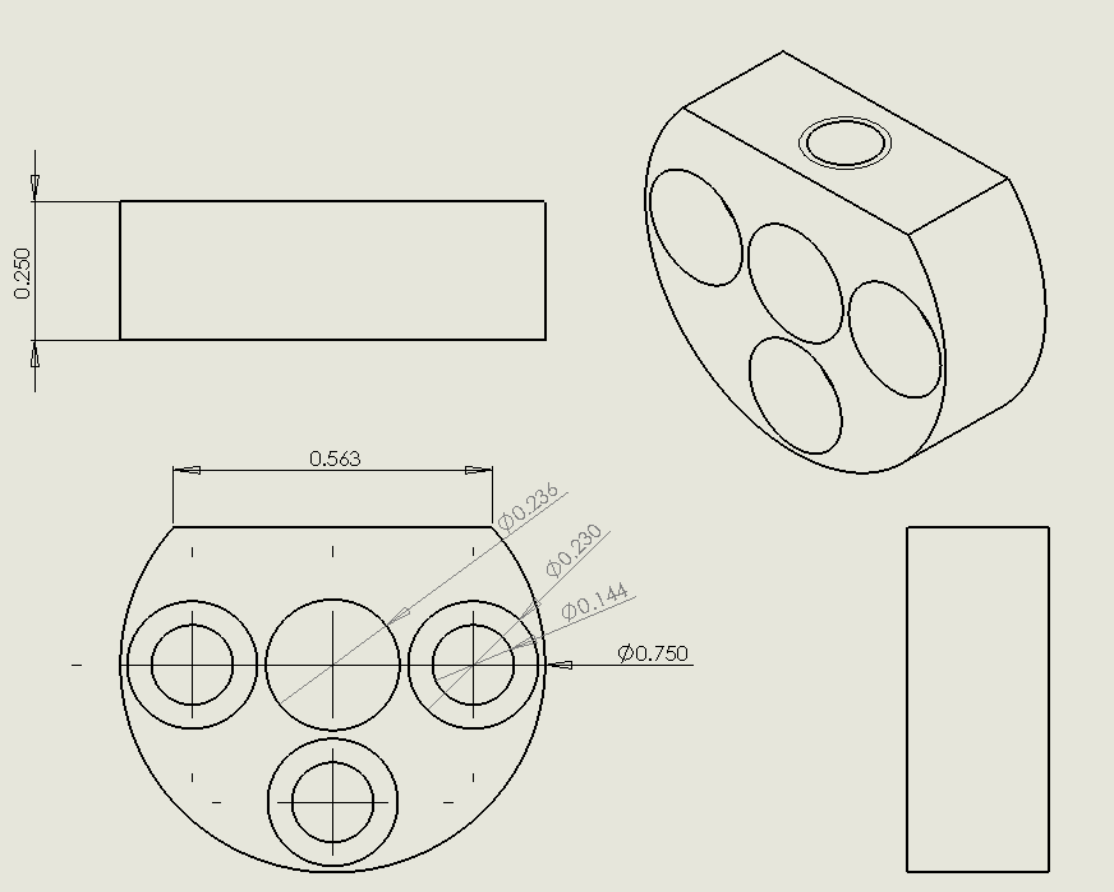
Tools used for fabrication - 1/8” End Mill, #36 Drill Bit, #36 Drill Bit, 6-32 Tap

Resource used - 12” x 12” x .220” Transparent Green Acrylic Sheet

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach
5. Tap Holes with 6-32 Tap
6. Sand Corners for Chamfer

Wheel Mount



Material – Acrylic (Plexi-glass)

Machines used for fabrication – Tormach Prolight 1000 CNC Machine

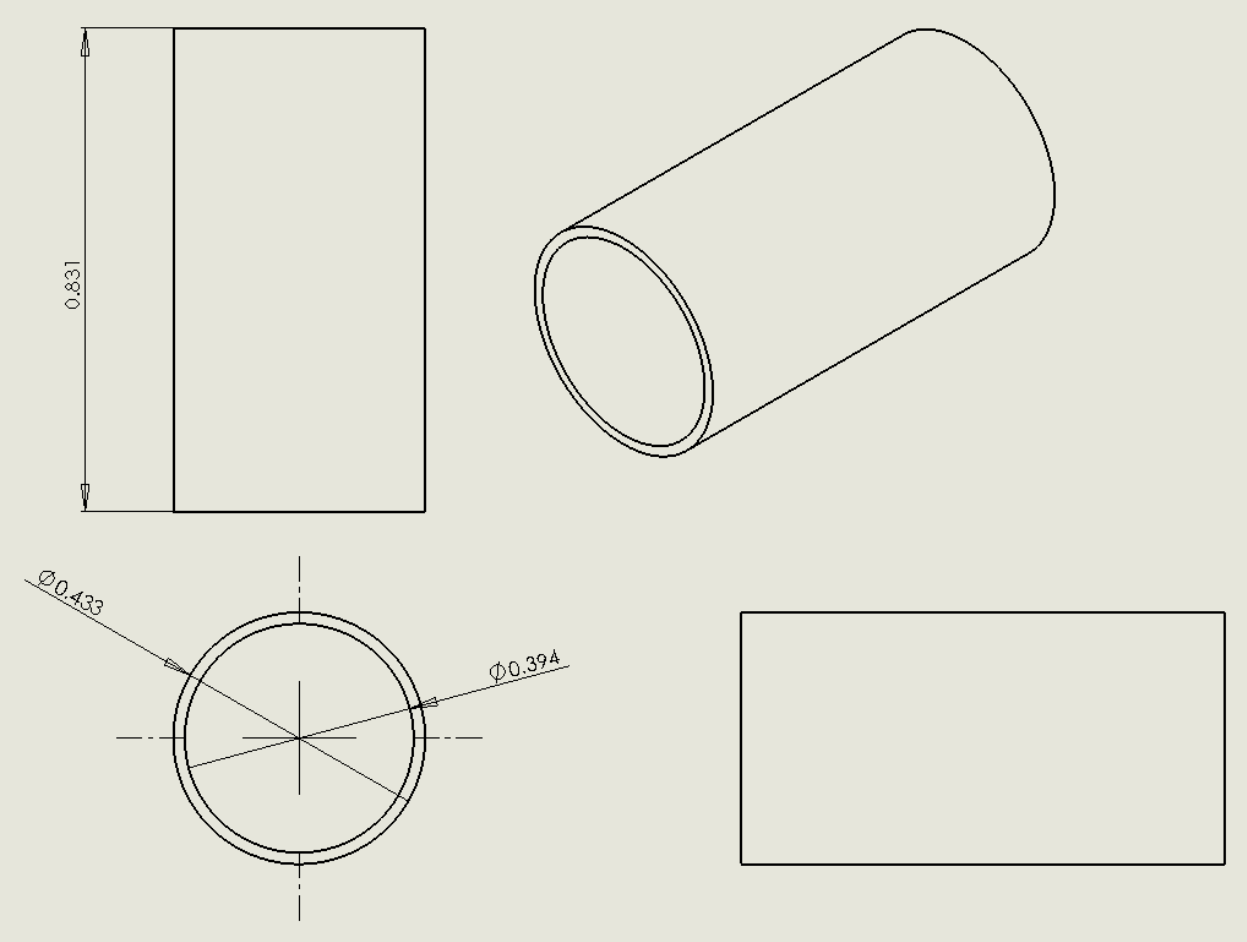
Tools used for fabrication - 1/8” End Mill, #27 Drill Bit

Resource used –

Process:

1. Construction of SolidWorks Model
2. CamWorks Modeling
3. Conversion into G-Code
4. Mount Part and Run G-Code on Tormach

Coupler



Material – Steel

Machines used for fabrication – Lathe, Bandsaw

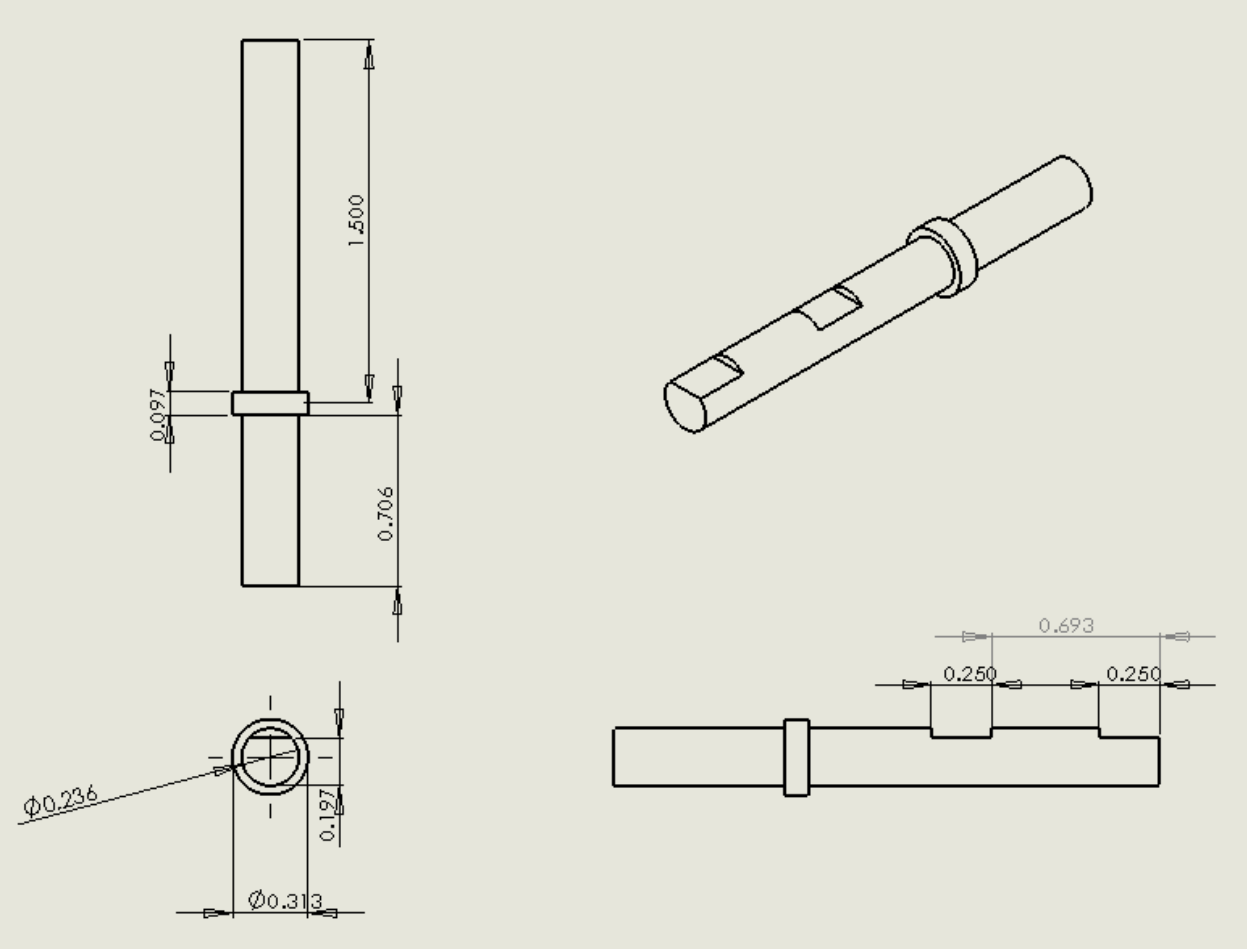
Tools used for fabrication - File

Resource used – 7mm Steel Pipe

Process:

1. Cut Part to Length with Bandsaw
2. Mount Part in Lathe and Reduce the Parts Thickness

Axle



Material – Steel

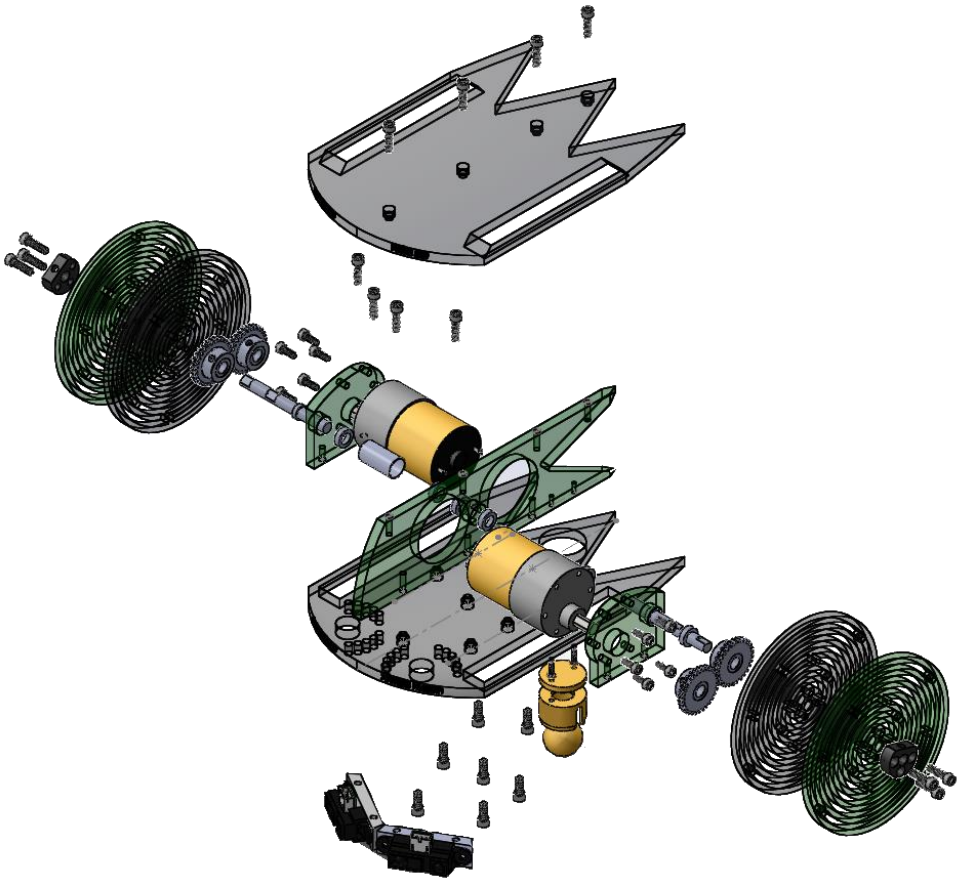
Machines used for fabrication – Tormach Prolight 1000 CNC Machine, Bandsaw

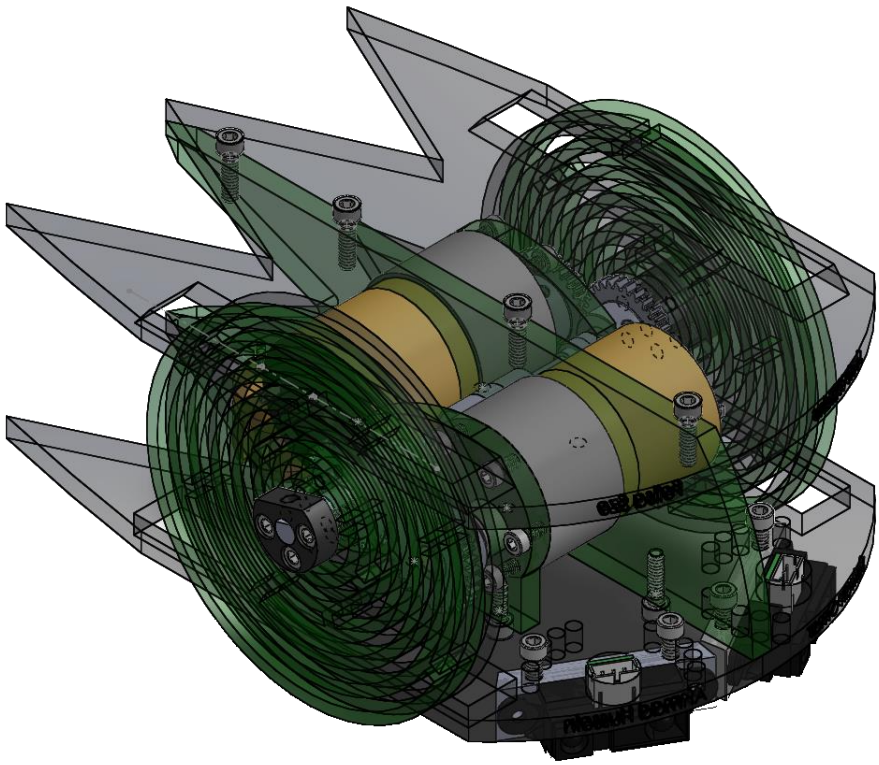
Tools used for fabrication – 1/4” End Mill, File

Resource used – 6mm Steel Rod

Process:

1. Cut Part to Length with Bandsaw
2. Mill Out Flat Spots on Axle
3. File Edges for Chamdfer





Design Analysis

Our team has excelled throughout this semester and one of the best ways to measure our success is through our robot design tests. The first test was to complete a straight section of the track. Although we are unsure who actually completed this first, it was not a challenge for us to do so. The second test was to complete one lap around the track without obstacles. We were the first team to do so, within only a couple weeks since the beginning of the semester. The third trial was to complete one lap around the track with obstacles. Not soon after we had test two done, we had completed test 3, way before its assigned deadline and long before any other group was close to doing so. The TYESA qualifier was only a couple weeks later, and as you can see a couple teams had caught up with us by then and we ended up in third place. We also went to TYESA afterwards and placed in the lower half, 8th place out of 11 teams. By testing four, we were once again able to get our robot working and placed 6th out of 12 teams in our class. Overall our team was quite successful and a lot has been learned from this experience.

TYESA Qualifier Scores

Team Name	Exhibit Score	Test Score	Total Score	Rank
Team Dynamics	97.17	129.60	226.77	1
Grinders	83.75	130.00	213.75	2
The SHOW	95.00	114.54	209.54	3
The Four Aces	94.00	100.00	194.00	4
Nitro	86.67	100.00	186.67	5
SOKOMS	104.67	30.00	134.67	6
AstroMechs	98.33	30.00	128.33	7
Jon Burns and the Burnouts	106.83	0.00	106.83	8
CARD	94.67	0.00	94.67	9
Geoff	0.00	0.00	0.00	10

2014 TYESA Spring Design Competition
Overall Scores

Team Rank	School	Team Name	Exhibit Score	Testing Score	Total Points
1	MCC	Grinders	98.30	332.00	430.30
2	MCC	Dynamics	106.30	262.00	368.30
3	BCC	Autocar	103.40	207.00	310.40
4	MCC	Four Aces	110.10	109.00	219.10
5	MCC	Nitro	100.30	100.00	200.30
6	JCC	Dirty Dogs	96.10	50.00	146.10
7	MCC	Burnouts	116.30	20.00	136.30
8	MCC	The Show	106.70	20.00	126.70
9	DCC	NASCARS	87.10	10.00	97.10
10	JCC	OMG	89.60	0.00	89.60
11	JCC	Samsquatches	79.10	0.00	79.10

2014 ENR259 Robot Testing 4 Scores														
Team Information	Trial 1			Trial 2			Trial 3			Trial 4			Overall	
Team Name	# Laps	Time (s)	Trial Score	# Laps	Time (s)	Trial Score	# Laps	Time (s)	Trial Score	# Laps	Time (s)	Trial Score	Total Score	Rank
The Burnouts	**5**	**24.6**	**85.37**	**5**	24.45	**85.55**	2	75	**40**	2	75	**40**	**250.9**	1
Astromechs	**5**	**56.5**	**53.51**	**5**	56.75	**53.25**	5	57.27	**117.7**	0	75	**0**	**224.5**	2
Team Dynamics	**3**	**60**	**30**	**5**	49.02	**60.98**	0	75	**0**	5	57.08	**117.9**	**208.9**	3
CARD	**2**	**60**	**20**	**3**	60	**30**	2	75	**40**	1	75	**20**	**110.0**	4
SOKOMS	**1**	**60**	**10**	**3**	60	**30**	0	75	**0**	2	75	**40**	**80.0**	5
The SHOW	**5**	**50.9**	**59.13**	**0**	60	**0**	1	75	**20**	0	75	**0**	**79.1**	6
Grinders	**5**	**56.5**	**53.47**	**2**	60	**20**	0	75	**0**	0	75	**0**	**73.5**	7
TED	**4**	**60**	**40**	**2**	60	**20**	0	75	**0**	0	75	**0**	**60.0**	8
Where's Tyler	**1**	**60**	**10**	**3**	60	**30**	0	75	**0**	0	75	**0**	**40.0**	9
Wide Open	**4**	**60**	**40**	**0**	60	**0**	0	75	**0**	0	75	**0**	**40.0**	9
Cause for Concern	**2**	**60**	**20**	**1**	60	**10**	0	75	**0**	0	75	**0**	**30.0**	11
Four Aces	**0**	**60**	**0**	**0**	60	**0**	0	75	**0**	0	75	**0**	**0.0**	12
Turbo	**0**	**60**	**0**	**0**	60	**0**	0	75	**0**	0	75	**0**	**0.0**	12
Nitro	**0**	**60**	**0**	**0**	60	**0**	0	75	**0**	0	75	**0**	**0.0**	12
Challengers	**0**	**60**	**0**	**0**	60	**0**	0	75	**0**	0	75	**0**	**0.0**	12

Bill of Materials

Component	Parts	Supplier	Description	Unit Cost	QTY.	Cost
12" X 12" X 1/4" Plexiglass	Chassis	Amazon	Base, Top and Midplane Plates	$12.00	2	$24.00
	Motor Bracket Right		Extra Support for Motors			
	Motor Bracket Left		Extra Support for Motors			
	Wheel		3.75" diam acrylic wheel			
	Sensor Attatchment		Acrylic Sensor Support			
Wheel System	40 Tooth Spur Gear	Amazon	Plastic spur gear	$11.78	2	$23.56
	20 Tooth Spur Gear	Amazon	Plastic spur gear	$9.31	2	$18.62
	Wheel Mount	Ebay	Aluminum Wheel Attachment	$3.77	2	$7.54
	Axle	Amazon	6mm Diam Steel Rod	$2.00	2	$4.00
	Ball_Bearing	Amazon	6mm ID, 10mm OD Ball Bearing	$0.78	6	$4.68
	Coupling	Amazon	Attachment for Bearings	$5.75	1	$5.75
	Caster ball set	Amazon	Ball Caster and Extension set	$5.67	1	$5.67
Electronics	Arduino Uno	Amazon	Micro Controller	$25.00	1	$25.00
	Adafruit motorshield	Amazon	Motor Shield for Arduino Uno	$10.00	1	$10.00
	IR Sensor	Amazon	10-80cm Sharp IR Sensor	$12.79	3	$38.37
Power Supply	Battery	Amazon	4500 mAh lipo batteries	$30.00	1	$30.00
Propulsion System	Motors	Pololu	37D Pololou Motors 19:1 Ratio	$30.00	2	$60.00
Fasteners	Pan Head Screw	Amazon	2-56 X 3/8 Pan Head Phillips	$0.04	2	$0.08
	Socket Head Cap Screw	Amazon	M3 X 8mm Socket Head Cap Screw	$0.30	10	$3.00
	6 - 32 x 1-2 cap screw	Fastenal	6-32 X 1/2" Screws	$0.59	17	$10.03
	6 - 32 x 1-2 cap screw	Home Depot	6-32 X 3/8" Screws	$0.55	4	$2.20
					Total:	$272.50

From our bill of materials, one can see that the electronics of the system are the most expensive. In order to reduce the cost the most, the use of fewer sensors would be ideal. The most expensive components were the batteries and the motors, both worth $30 each. The total cost of the robot came out to be $272.50, compared to other class robots, this design is fairly cheap, however its cost could still be reduced some. Overall, our system would not be ideal for mass production, a direct result of the complexity of assembly, to the cost of each component, further resulting in a marginal profitability.

Labor Times & Cost

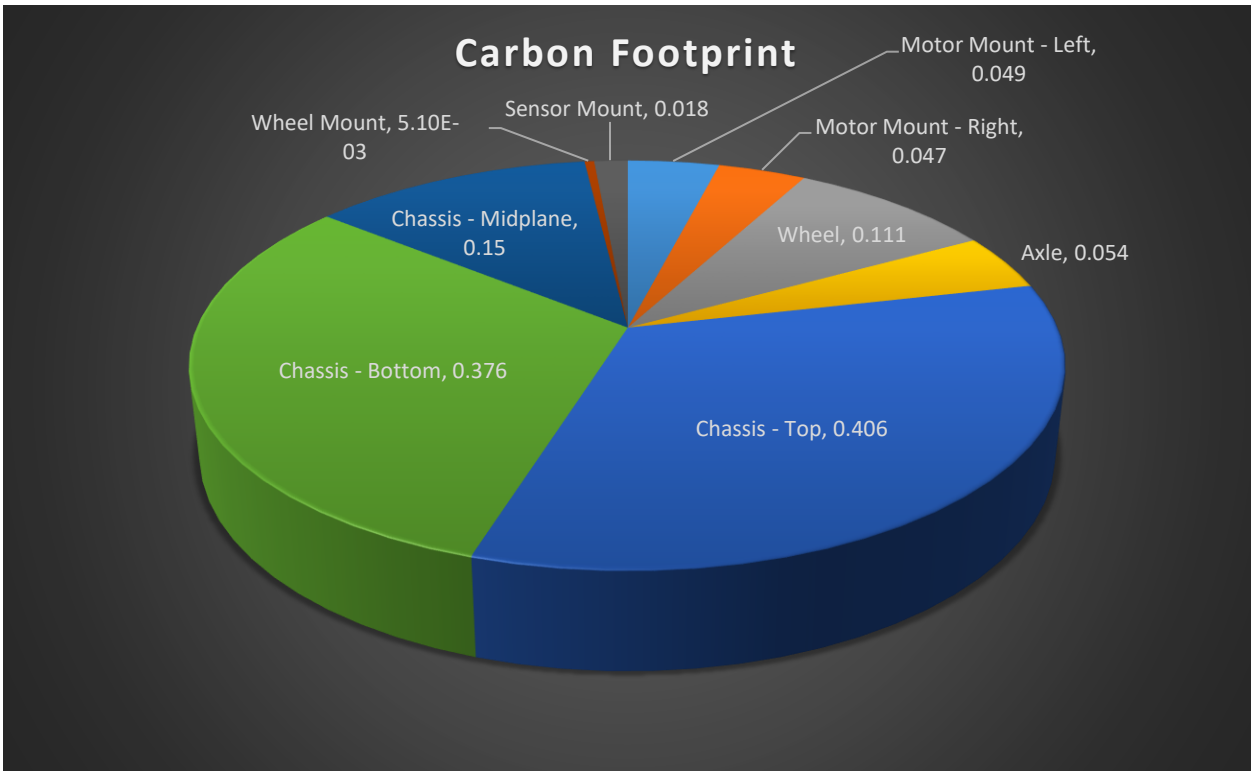
		Felisa Sze	Josh Wilkins	Ahmad Hussein	Salwan Omar	Weekly Totals	Weekly Labor Cost ($)
	27-Jan	8.5	7	10	13	38.5	770
	3-Feb	8	17	5	11	41	820
	10-Feb	8.5	12.5	2	8.5	31.5	630
VAC	17-Feb	12.5	6	2	10.5	31	620
	24-Feb	8.5	20.5	0.5	10	39.5	790
	3-Mar	10.75	9	0	10.5	30.25	605
	10-Mar	9.5	12	5.5	11.5	38.5	770
	17-Mar	6.5	9	2.2	9	26.7	534
	24-Mar	6.5	17.5	0	6	30	600
	31-Mar	7.5	7.5	7.5	4.75	27.25	545
	7-Apr	12	12	7.5	6.75	38.25	765
VAC	14-Apr	18.5	10.5	4.5	8.25	41.75	835
	21-Apr	27	24.5	1	20	72.5	1450
	28-Apr	5	11.5	4.5	4	25	500
	5-May	15	6	3	5	29	580
	12-May	11	12	2	8	33	660
	Totals	175.25	194.5	57.2	146.75	573.7	11474
	Cost ($)	3505	3890	1144	2935		11474

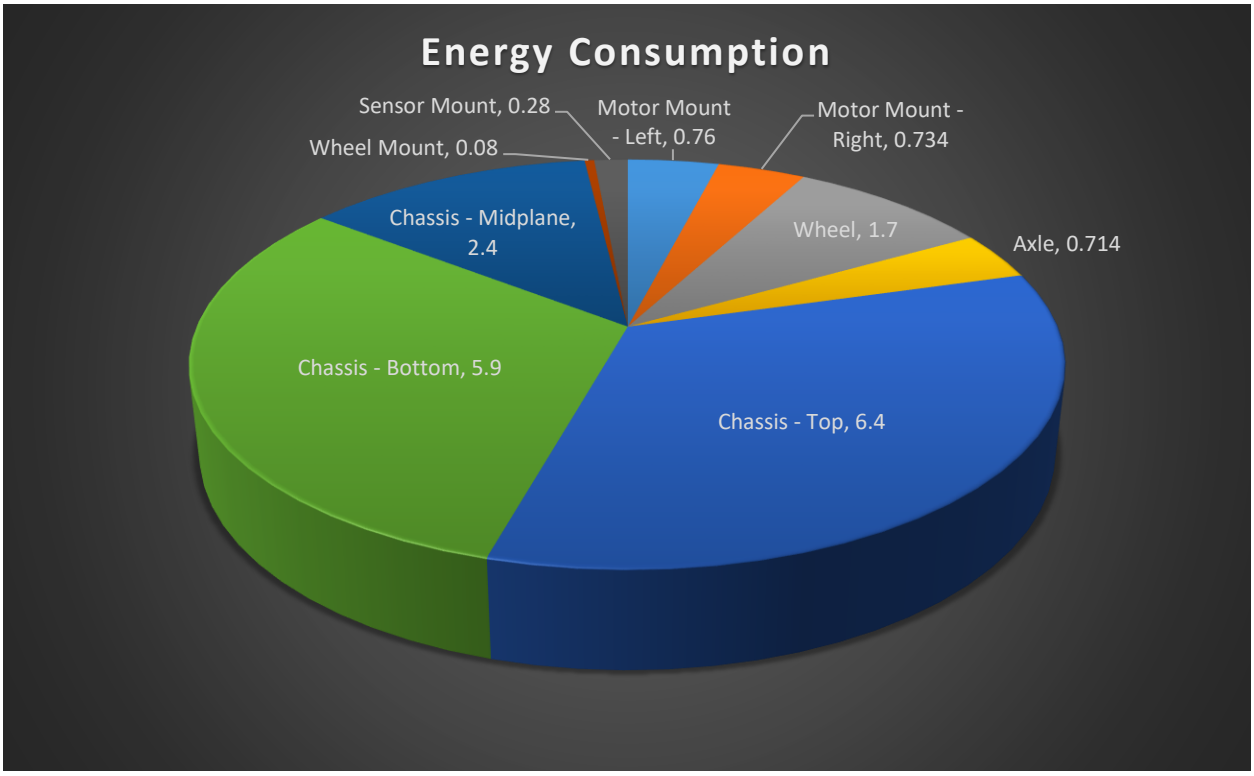
Unsurprisingly, labor cost exceeded all other expenses, however $11474 was a huge shock. From this table, we can see that, in order to produce a reliable and working machine, lots of effort must go into the design. It is clear that time and labor have more of an impact on projects than any other aspect of the design. Not only is it the most important aspect, it is the best way to limit the cost by finding and implementing the optimal solution as quickly as possible. Of course the opposite is true as well, if the optimal solution is not found initially, many hours and lots of money would be spent on an imperfect system, only to be replaced. In addition to the cost of the optimal solution, the time spent on the imperfect design would amount to much more than a couple extra hours brainstorming. The implication of this table is therefore that you must first find, then verify that your solution is optimal, before finalizing a decision.

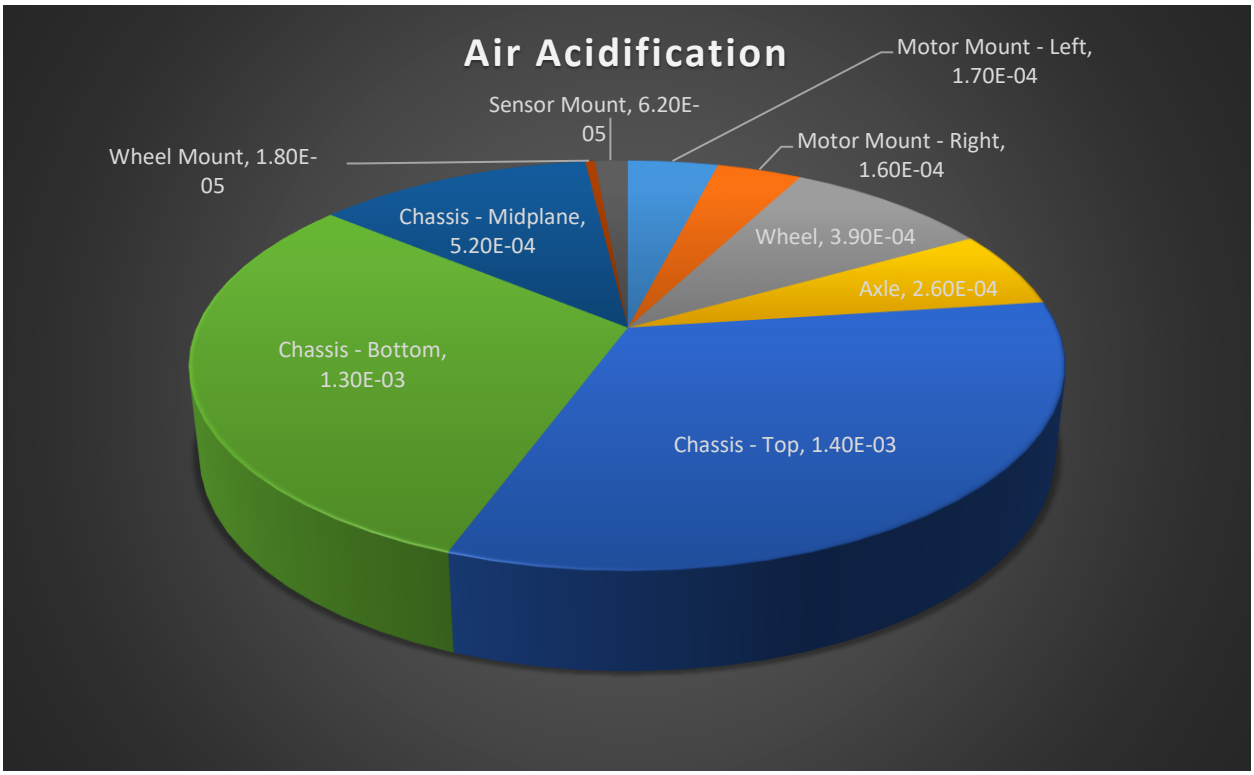
Environmental Impact

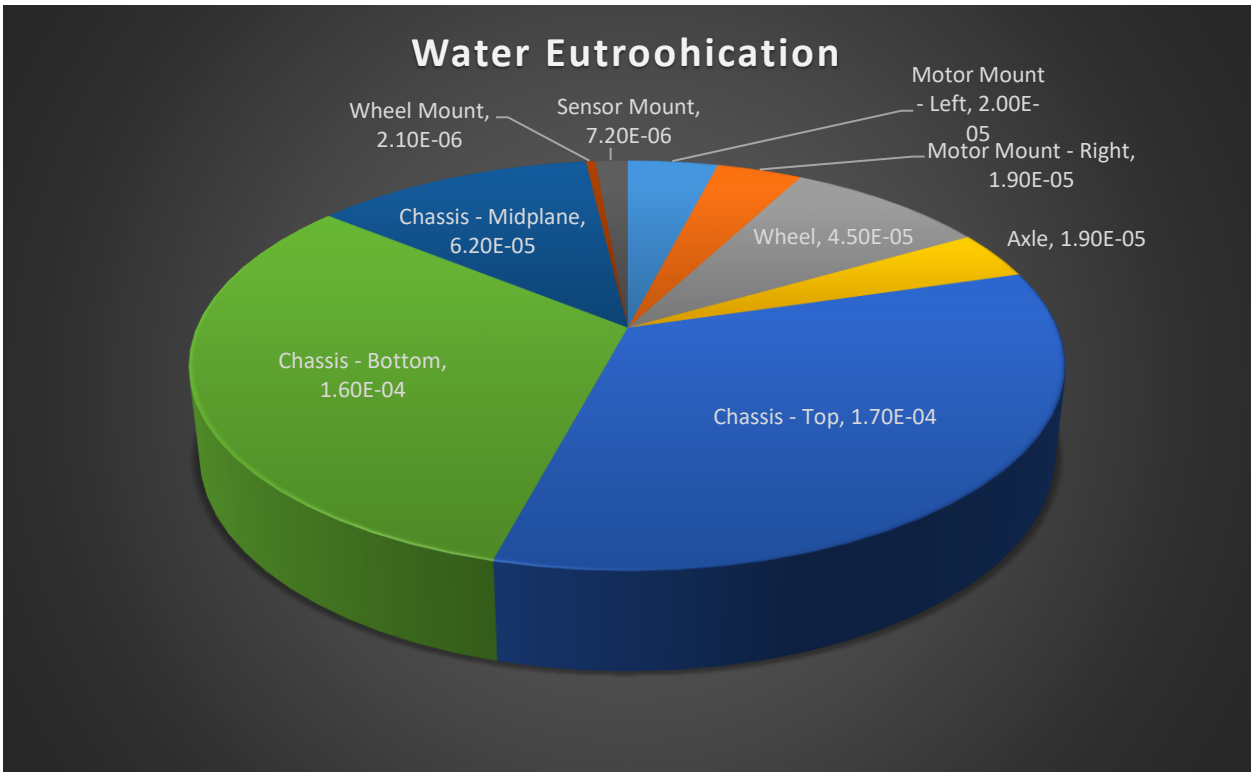
	Carbon Footprint	Energy Consumption	Air Acidification	Water Eutroohication
Motor Mount - Left	0.049	0.76	1.70E-04	2.00E-05
Motor Mount - Right	0.047	0.734	1.60E-04	1.90E-05
Wheel	0.111	1.7	3.90E-04	4.50E-05
Axle	0.054	0.714	2.60E-04	1.90E-05
Chassis - Top	0.406	6.4	1.40E-03	1.70E-04
Chassis - Bottom	0.376	5.9	1.30E-03	1.60E-04
Chassis - Midplane	0.15	2.4	5.20E-04	6.20E-05
Wheel Mount	5.10E-03	0.08	1.80E-05	2.10E-06
Sensor Mount	0.018	0.28	6.20E-05	7.20E-06
Total:	1.2161	18.968	4.28E-3	5.04E-4

From the above table and the corresponding charts, one can see that, from the fabrication of this robot, that a lot of energy has been consumed. The biggest consumption of energy and output of waste came from the fabrication of the three chassis plates. In fact, almost 75% of the consumed energy and output waste came from the fabrication of these. An alternative material for the chassis plates would clearly aid in the reduction of the energy consumed and waste emitted.


Carbon Footprint
Motor Mount - Left, 0.049
Sensor Mount, 0.018
Wheel Mount, 5.10E-03
Motor Mount - Right, 0.047
Chassis - Midplane, 0.15
Wheel, 0.111
Axle, 0.054
Chassis - Bottom, 0.376
Chassis - Top, 0.406


Energy Consumption
Sensor Mount, 0.28
Motor Mount - Left, 0.76
Motor Mount - Right, 0.734
Wheel Mount, 0.08
Chassis - Midplane, 2.4
Wheel, 1.7
Axle, 0.714
Chassis - Bottom, 5.9
Chassis - Top, 6.4


Air Acidification
Motor Mount - Left, 1.70E-04
Sensor Mount, 6.20E-05
Motor Mount - Right, 1.60E-04
Wheel Mount, 1.80E-05
Chassis - Midplane, 5.20E-04
Wheel, 3.90E-04
Axle, 2.60E-04
Chassis - Bottom, 1.30E-03
Chassis - Top, 1.40E-03


Water Eutroohication
Motor Mount - Left, 2.00E-05
Wheel Mount, 2.10E-06
Sensor Mount, 7.20E-06
Motor Mount - Right, 1.90E-05
Chassis - Midplane, 6.20E-05
Wheel, 4.50E-05
Axle, 1.90E-05
Chassis - Bottom, 1.60E-04
Chassis - Top, 1.70E-04

Team & Technical Analysis

Overall, our team was moderately functional: We were able to communicate well, we all worked hard to achieve, and every goal set was achieved before its respective deadline. The pitfalls of our team however, lay within the indecisiveness and stubbornness between the group members. Every decision made had to be agreed upon, by every member, as with any type of team project, but what made this project especially difficult for decision making was the sheer amount of choices available at our disposal. Despite the challenges faced, our team pulled through and we accomplished an effective and somewhat reliable robot that can complete the required task assigned, so in the end, our team can be considered functional, but know that we, like most other teams, had our difficulties throughout the design process.

Many issues arose throughout the semester, most of which our team was able to resolve. One of the first problems we faced was with our microcontroller. A communication error had occurred on the Arduino Romeo, a problem we now believe to be from the shorting of two wire coming in contact with the board. Another problem we faced was the incorrect placement of the sensors. This was the direct result of poor planning, the forgotten necessity of tread, in the SolidWorks modeling. This absentmindedness resulted in a higher front end on our robot, making the shields around the wheels ineffective, further resulting in the robot climbing the quarter round on the track instead of deflecting off it. This was resolved through the attachment of two small bearings on both sides of the robot as well as the lowering of the caster wheel in the back. Not only did this save us from going off the side of the track, but also saved the sensors from being smashed every time we hit the wall, giving us more consistent readings than before. Many minor and miscellaneous problems transpired as well, however almost all were fixed in a timely manner, with not too many repercussions.

For future design challenges, we would recommend that the class be changed to a 2-credit hour class. In addition to this change, each group should be comprised of 1 or 2 people instead of a group of 4 people. The advantages of this would be that each team member would be more involved in the fabrication, the design, and the coding aspects of the challenge. Not only will each team member be more involved, thus learning more in the process, but more would also be accomplished. No more indecision or communication errors between teammates and more ideas would be brought about and experimented with. The only downfall of this would be that each team member would have to do a little more work, not by much however because I guarantee that in each team now, there is at least one member who does nearly nothing.

Component selection was an important aspect of this design challenge. Our first consideration was the choice of motors that we should use. Research showed that the Jameco motors that the school offered were fairly lacking in power and speed. Some Pololu motors were found that were much faster and more powerful, thus these motors were chosen. The next major decision came from the choice between ultrasound and infrared. We chose the Sharp IR sensors for multiple reasons; they are not dependent on the materials used, ambient noise in different environment won't affect an IR sensor, and IR sensors are generally more consistent than ultrasound sensors. The next obvious choice was then the range of the sensor.

At first, we chose to use the shorter range sensors, a 4 to 30cm range because the mid-range sensors can't be used when up close to the quarter round on the track. However, as more and more success was seen on other robots with the longer range sensors, we switched over to these mid-range sensors with a range of 10 to 80cm and began to follow closer to the middle of the track than before.

Over the course of this semester, our team had to make many design decisions. Some of these decisions worked out well, other decisions did not. Of the inferior decisions made, the worst was the complicated design scheme. Not only was the assembly difficult, but the condensed space made disassembling required for some minor fixes, wasting some valuable time. Some positive improvements to our design were the addition of an easily adjustable angle on the IR sensors and the addition of molded tread to our wheels. The best decisions were made toward the latter end of the semester however, with the change from all short range sensors to all mid-range sensors and the gearing down from a 1:1 gear ratio to a 1:2 ratio. These provided the most reliable and consistent results thus far and vastly improved the functionality of our design.

Despite all the design choices made, good or bad, some modifications and alternative design decisions would be made if the project were to be repeated. Firstly, bumpers would be made and the robot would ride along the edge of the wall. Secondly, digital IR sensors would be used in conjunction with an IMU (inertial measurement unit) to precisely navigate the track. This would reduce the error dramatically within the system and allow for more control over the robot. Following the design principle used by many and coined by Kelly Johnson – Keep it simple, stupid, the design of the robot would be made much simpler and probably much smaller than it is now.

Overall, our team has worked hard to succeed, and despite some bad performances and the fact our design wasn't perfect in the end, the potential was still there. We have each learned a lot throughout this semester from this design challenge. This experience has been more than any one of us could have hoped for.