Project J: Open Source PVC Electric Wheelchair

Sponsor: Dylan Vaughn, CEO of Robomakery Inc.

Mentor: Dr. Jack Silberman, UCSD Mechanical and Aerospace Department

Team Members: Ibrahim Aldabbagh, David Lai, Darlene Montesano, Jooyoung Park

June 15, 2018

Executive Summary:

The project goal is to create an open source, PVC electric wheelchair, so that others can view the schematic and follow instructions in order to build their own electric wheelchair. From the previous quarter's work, we are trying to simplify the recreation of this project and minimize the costs and time required to build the project. As a result, we decided to do more research on the current parts and if possible find replacements that can increase functionality and cut down costs. We thought about possible improvements in the speed control of the wheelchair as well as the movement accuracy. We looked into the battery of the wheelchair and realized that improvements could be made for that too. Not only that, but we also got feedback from the previous quarter's team on other possible methods to improve the chair as well as issues that they faced while working on the project.

From our research, we decided to dive in and make changes to the project that we thought would improve the current design. First, we knew that a charging circuit was necessary so that we could reuse the same battery and recharge it for future use. We were able to get rid of the previous method of powering, which was using alligator clamps for a direct connection to the battery. We then proceeded to make more improvements with the battery by adding an OLED display that shows the battery life of the wheelchair. This allows the user to know when it is time to recharge their wheelchair.

We also made the decision to swap out the old microcontroller and use an Arduino UNO instead. Our reason behind this choice is because an Arduino is much more common and easy to find, and there is a lot more support and documentation for it, which makes programming our wheelchair much easier. We were originally using an MBED microcontroller, but made the switch because we realized we did not have a need for such a high level development board.

On top of this, we completely rewired the circuit that was previously designed. We did this because we found it difficult to follow the wires and as a result, it was hard to make progress. We took the time to optimize the wire connections and add clear labeling and documentation so that anyone would be able to understand how the wheelchair works.

A huge step towards the chair was our progress with the new brushless motors. We were able to successfully program and integrate these new motors as well as the hall sensors in order to control movement on the wheelchair. This allows for a more stable method of controlling speed as well as direction on the wheelchair. We can use the joystick to actually move the wheelchair around and control direction.

Overall, we were able to learn a lot from this project. We gained valuable technical skills and learned how to work on a project in a team environment, building up our communication skills. We also discussed ideas and received advice from our sponsor and mentor as we continued to grow throughout this project.

Table of Contents

Introduction

The PVC Electric Wheelchair project started as a capstone project at Brigham Young University, to provide a cheaper alternative and help kids whose guardians cannot afford an electrical wheelchair. The project was released as an open source project to give access to low income families in need of a cheaper, but fully featured PVC electric wheelchair. Currently, there are different teams working on perfecting the current prototype design. Our team is focused on improving the design of the previous ECE 191 UCSD team that designed the current model we are working on. We made changes to the wiring as well as added additional features to our open source wheelchair. We hope that this chair will be easy for others to rebuild on their own using common hardware tools while maintaining a low budget. Our goal is to make this design as cheap and as simple as possible so that most people can afford this and have the skills necessary to be able to build their own electric wheelchair.

Statement of Work

Our statement of work included the following: to design a charging circuit for the battery so that it can be rechargeable, display the battery life on an OLED, and update the current brushless motors . By designing a charging circuit for the battery, we will get rid of the old alligator clamps that were used to power the chair before. Also, the OLED display will allow the user to know when he/she needs to recharge the wheelchair. Updating the Motor controllers with hall sensors will allow better speed control. In addition to these major tasks we will also rewire the components and update the documentation as well as clearly label each part in order for future replication as an open source project.

Ethical and Safety Considerations

Ethical Issue: There are no major ethical issues with this project.

Safety Issue: Since this is an electric wheelchair, there is the danger of electric shock. If wiring is done incorrectly, there is the possibility that the circuit may short and the motors may malfunction. When we made the voltage divider circuit, this safety issue was an actual concern. We used 5Ω and 19Ω resistors to make the voltage divider for 24V battery. The resistors were burned down because too much current went through the resistors. We realized that we need to make the current lower by using higher values for the resistors.

When we disconnected our wires after working on the project, the Arduino was burned a little. The reason this happened is because we disconnected the wires out of order, even though the Arduino was not connected to power. We realized that we need to consider the order of how we disconnect the wires as well.

Technical Backgrounds

Some technical background is necessary.

- Basic understanding of circuitry and electrical wiring in order to follow the schematic provided.
- Basic understanding of programming in order to upload the code to the microcontroller.
- Basic tool knowledge in order to assemble PVC wheelchair parts.

Approach and System Design

Our approach involved extensive research for the different components that were in the previous design of the electric wheelchair, in order to help us on deciding how to implement further improvements and changes. After this phase, we decided to make the first change to the way the previous team had charged the battery. We got rid of the alligator clamps that was used and utilized a conventional charger instead. The second change that we had made was rewiring the circuitry, in order to have neater cabling and labels for a consumer to understand the functionality of the components and the order in which they are supposed to be connected. Additionally, a battery life display was added as a feature to the wheelchair to assist the user in determining when to recharge their battery. Most importantly, we updated the old brushed motor controllers to ones that were compatible with the Hall-sensor effect. Utilizing the sensor-equipped motor controllers made it possible for us to gain better control of the speed. Finally, we decided to make use of an Arduino Uno board, in place of the Mbed FRDM-K64F board which was used by the previous team. This decision was based on the ubiquitous nature of Arduino, as well as the accessibility of different resources and references to future users of the wheelchair. Below are more detailed explanations of our design changes.

Rechargeable battery

When we received the wheelchair at the first time, the battery was charged by alligator clamps only. The clamps are uncomfortable and dangerous. So, we removed the alligator clamps and changed it to plugging in charge system. This allows the wheelchair to be more versatile and self sustainable as a portable unit, which is an important feature to have.

Figure 1. Shows the old alligator clamps used to charge

Figure 2. Shows battery charger

<u>Rewiring</u>

This is a continuous project from the previous quarter. When we got the wheelchair, it was hard to understand and look at connections without clear documentations. The wires were touched each other and some wires were lost. We decided to rewire all connections for better understanding and making it clear. We labelled each wires, updated diagrams of the connections and also documented the functionality of each part.

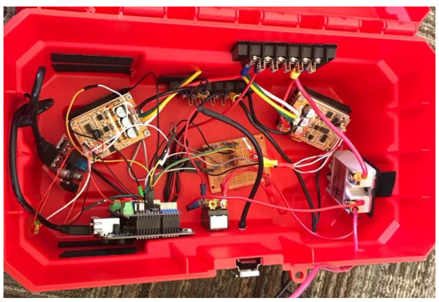

Figure 3. Initial Wiring

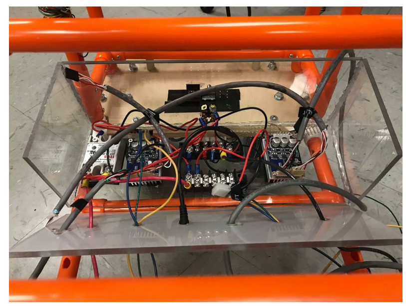

Figure 4. Rewired Circuit

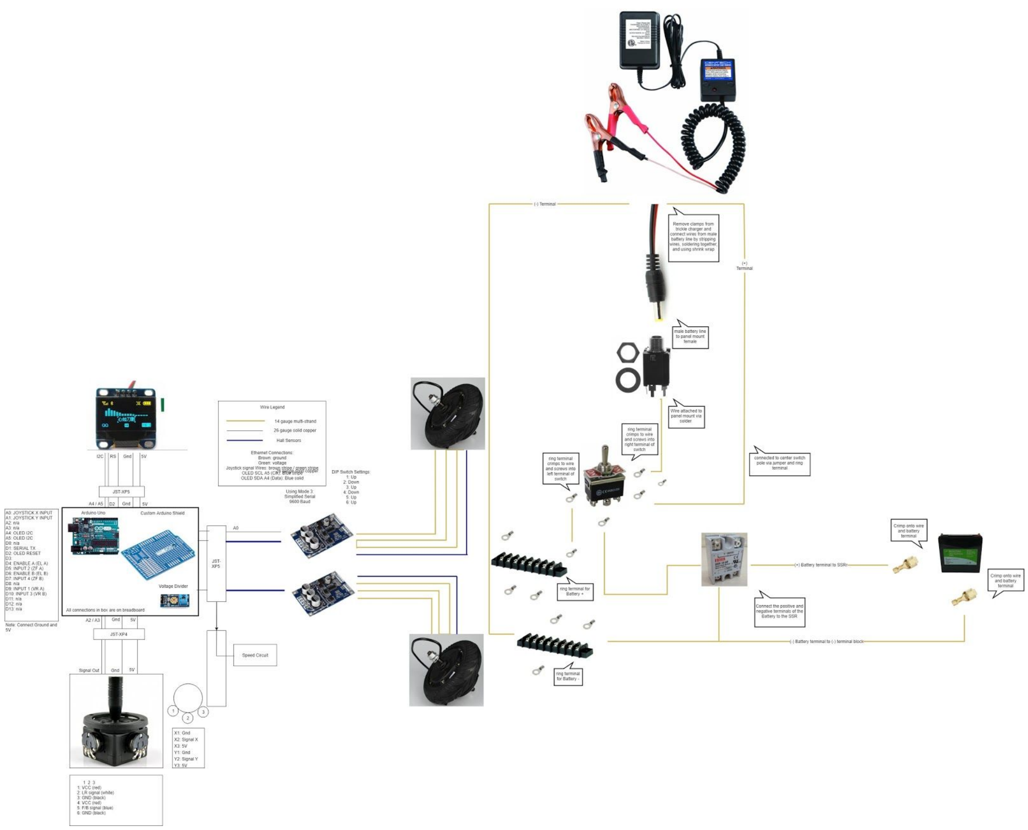

Figure 5. Shows the updated schematic the resembles the wiring done in fig.1

OLED Display

We decided to go forward with an LCD display first, due to recommendations that we found online. Then, we planned to modify the codes of LCD to apply into OLED what we are going to use. There was some obstacles for it. LCD has 16 pins, but OLED has 4 pins. We needed to research about all of the pins' role and rearrange every wire. Then we write a new code of battery indicator for OLED, because OLED has totally different libraries with LCD.

We faced problem in battery part too. The 24V battery can burn the Arduino because Arduino can only support a maximum of 5V. We needed a voltage divider in order to convert the 24V to 5V. 24V should be displayed to 5V, and 12V should be displayed to 2.5V proportionally because we wanted to display the battery with bars. If we indicate the battery life with bars, we just need the proportion from maximum battery. We used high power resistors, 5k Ω and 19k Ω for the voltage divider because we needed to make the current low to avoid burning the resistors. Through the voltage divider, the OLED display can read in the correct voltage, display battery life with bars and be powered accordingly.

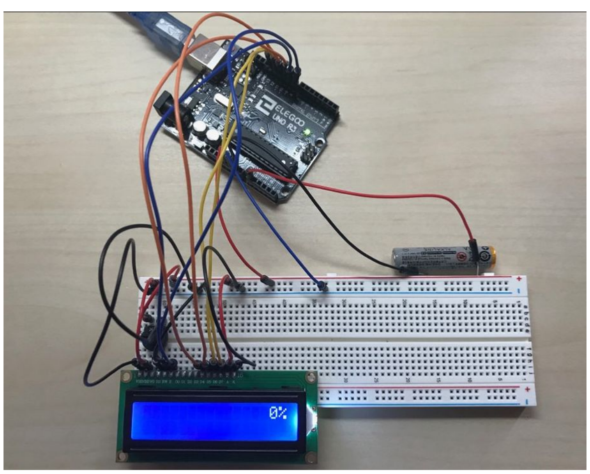

Figure 6. LCD displaying battery level

Figure 7. OLED displaying battery level

Motor controller and Joystick

Upon initially receiving the wheelchair, the motor did not provide enough power for the wheels to make it move. Upon investigating the issue, it turned out to be a problem with the wiring. However, even after fixing the wiring, the motor controller that the previous team had used were only able to move the wheelchair forward, and they also sent more voltage to the right wheel compared to the left one. Consequently, we decided to switch to brushless DC motor controllers which support hall sensors. For reference, brushless motors are typically 85-90% efficient whereas brushed motors are around 75-80% efficient. This difference means that more of the total power used by the motor is being turned into rotational force and less is being lost as heat. In addition, brushless motors don't have the friction and voltage drop that brushes create by dragging against the spinning commutator.

In a sensor-equipped BLDC motor, each Hall-effect sensor is combined with a switch which generates a logic "high" (for one magnetic pole) or "low" (for the opposite pole) signal. The commutation sequence is determined by combining the logic signals from the Hall-effect sensors and associated switches. At any time, at least one of the sensors is triggered by one of the rotor's magnetic poles and generates a voltage pulse. Figures 8 and 9 illustrate the commutation sequence of a three-phase BLDC motor driven anti-clockwise. The Hall-effect sensors are mounted at positions "a", "b" and "c". For each step in the commutation sequence, one winding (either "U", "V" or "W") is driven high by the MOSFET bridge while one is driven low and the third is left floating. For example, at the top left of the figure, U is high (forming an N pole), V is low (S) and W is floating. The resulting magnetic field moves the rotor anti-clockwise as its permanent magnets are repelled by one winding and attracted by the next. The second stage (below) shows winding U remaining high while V switches to floating and W switches low thus maintaining the 'rotation' of the magnetic field and moving the rotor with it. The remaining commutation steps, one electrical cycle, completes half a mechanical rotation of the rotor. On the coding side of reading and writing voltage signals, we used the AnalogRead and AnalogWrite commands on the Arduino terminals to control the speed of the wheelchair. The commands allowed us to make use of PWM (Pulse Width Modulation) signals, to get analog results using digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The code we used is discussed in more detail in the results section.

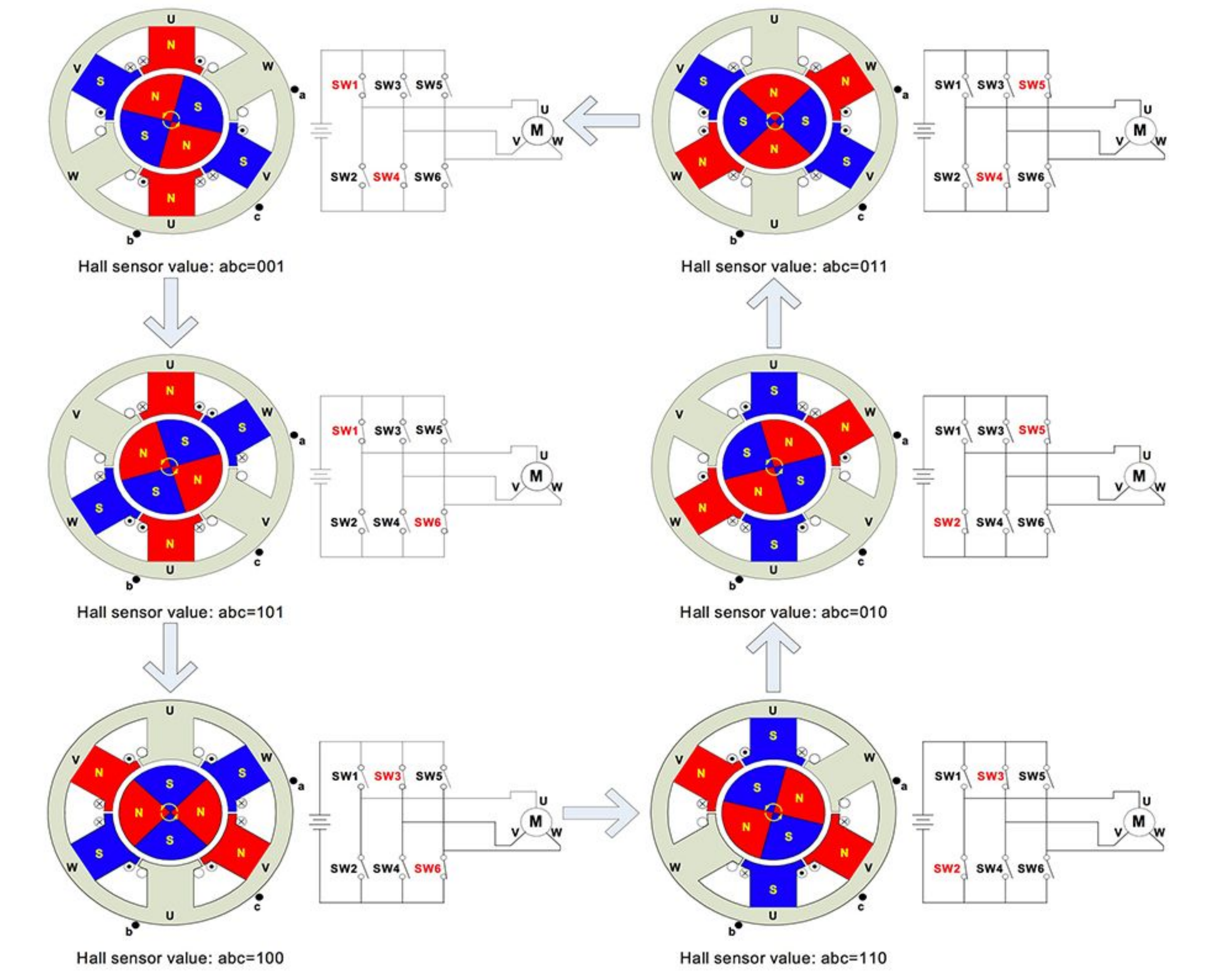

Figure 8. Electronic commutation sequence for three-phase BLDC motor using MOSFET bridge and Hall-effect sensors. In this case the rotor is driven anti-clockwise and the Hall-effect sensors ('a', 'b' and 'c') are mounted at 60° intervals. (Source: Digikey)

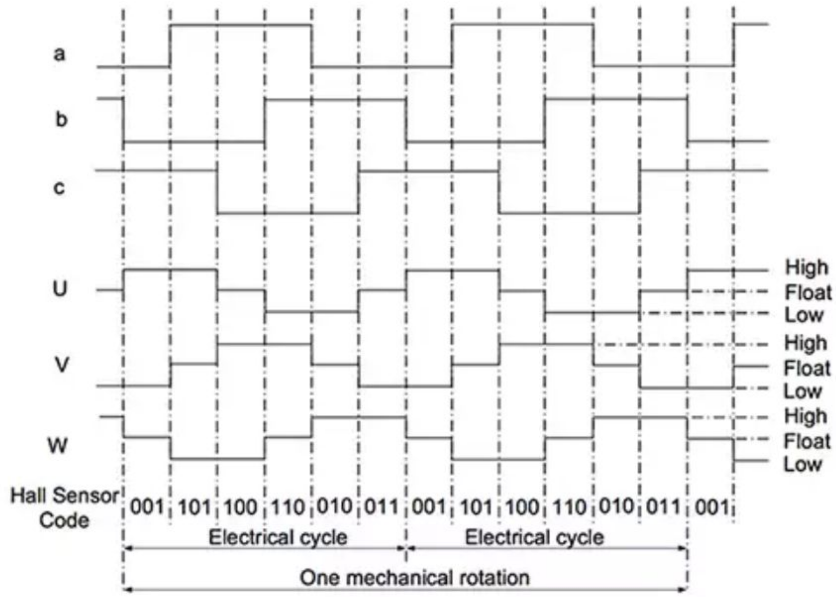

Figure 9: Hall-effect sensor logic switch output and winding status timing diagram for three-phase BLDC motor driven anti-clockwise. Note how at least one logic switch and winding changes status every 60°. (Source: Digikey)

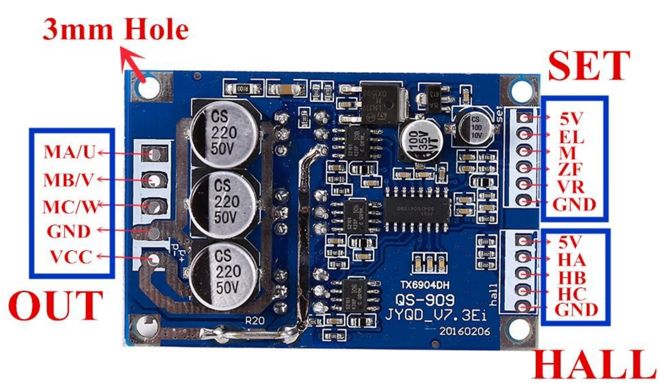

Figure 10. Shows the schematic of the new motor controllers

Results

For the rewiring of the circuitry of the wheelchair, we studied the previous wire design of the wheelchair and realized that it was unorganized and a bit difficult to follow.

As a result, we decided to take apart the rewiring and document each part so that there is a record for it. We swapped out the box and added newer wires that would not come off so easily. We also achieved to create neater wiring that allows an outsider to understand the circuity better, every wire is labeled and has a very specific purpose. Also, we made a schematic that has all the updated components used making it easier to follow and recreate the electrical wheelchair.

For the battery, we were able to successfully design a charging circuit in order to recharge the battery for reuse. This allowed for more efficient use of the battery because the battery would not just be a one time use, but can have an extended life as a result. Also, we were able to add an OLED display that showed the percentage of power remaining on the battery. Users are now able to know when is the appropriate time to recharge the battery of their electric wheelchair.

For the motor controllers, the new interface proved to be more successful than what was configured in the previous system. We were able to utilize the hall sensors to ensure that upon reading the commands from the user through the joystick, the motor controllers would send more accurate voltages. This makes the wheelchair more responsive to speed control and a lot less sensitive to the joystick commands. It also makes the wheelchair usable and eliminated the danger concern that was expressed to us upon receiving the chair. One issue worth noting is that the motor controllers are not sophisticated enough to fully accommodate the Hall-sensors. However, this issue can be fixed by replacing the motor controllers with more advanced alternatives.

In terms of the motor control code, we based it on the connections seen in Table 1. In our code, Y position means left and right and X position means forward and backward; the axes were switched and we read our coordinates on that basis. Each axis ranges from points 0 and 1024, with roughly 512 being the center. However, we allowed there to be a tolerance based on the joystick coordinate being read for each axis. Typically, this “tolerance” was a 30-50 point increment. Moreover, we used the map() function to map input voltages between 0 and 5 volts into integer values between -35 to +35, instead of 0 and 1024. The mapping allowed us to operate the chair at lower speeds with less sensitivity and much better response. This was not possible in the previous implementation.

Finally, we attempted to fix the wheel calibration problem by experimenting with different code implementations. We found that combinations of AnalogRead() and DigitalWrite() functions allowed to send 0 or 5V to the ZF terminal (see Figure 10) to control the direction better. This also allowed for smoother right and left turns, by turning the wheels in opposite directions to complete a turn.

Bill of Materials

PVC Parts	$187.73
Frame	$37.50
Drivetrain	$90.80
Electronics	$197.25
Grand Total	$513.28

This is our total budget for the electric wheelchair costs, broken up into multiple parts. We did not have additional costs to the PVC parts, but for our frame we bought additional parts and this brought up our total costs slightly. We also added to our drivetrain and electronics cost, thus bringing up the grand total to around $500. However, we are still staying within our budget of $750 in order to keep the electric wheelchair at a low and affordable cost for others. Below is a more detailed list for the materials.

PVC					
½ in Schedule 40 PVC	$6.18	6	F012FGP-OR-5	FORUMUFIT	www.forumufit.com

4-Way Tee Connect	$1.95	6	F0124WT-OR	FORUMUFIT	www.forumufit.com
5-Way Cross Connect	$3.38	4	F0125WC-OR	FORUMUFIT	www.forumufit.com
3-Way Elbow Connect	$1.69	6	F0125WE-OR	FORUMUFIT	www.forumufit.com
90 Degree Connect	$1.43	6	F01290E-OR	FORUMUFIT	www.forumufit.com
Tee Connect	$1.43	6	F012TEE-OR	FORUMUFIT	www.forumufit.com
Slip Sling Tee Connect	$1.80	9	F012STE	FORUMUFIT	www.forumufit.com
½ in Schedule 40 PVC	$6.18	6	F012FGP-OR-5	FORUMUFIT	www.forumufit.com
Fastener Pack	$14.95	1	FASTENER-PACK	Open Wheelchair	www.openwheelchair.org
Gear Box Plate	$14.95	2	GEARBOX-PLATE	Open Wheelchair	www.openwheelchair.org
Casters	$7.50	2	PRES5024ZN-TPR	A1-Casters	www.A1-Casters.com
Plexyglass*	$9.95	1	Plexyglass	Store	In store
DRIVETRAIN					
Actobotics 8mm Bore, 0.77”	$4.99	2	RB-Sct-628	robotshop	www.robotshop.com
Actobotics 1.50" x 0.770"	$4.99	2	RB-Sct-376	robotshop	www.robotshop.com
Actobotics 6" Heavy Duty	$9.99	2	RB-Sct-437	robotshop	www.robotshop.com
Takanawa 555 Gearbox	$14.18	2	2015021601	AliExpress	www.aliexpress.com
Brushless Motors*	$4.95	2	Motors	Store	In Store

ELECTRONICS					
Joystick Motor Controller	$149.90	1	JOY_MOTOR_CTRL	Open Wheelchair	www.openwheelchair.org
Joystick Enclosure	$27.35	1	http://shpws.me/HHRo	Shapeways	www.shapeways.com
Arduino Uno*	$19.95	1	Arduino	Store	In store

Conclusion

To conclude, we have successfully implemented major changes to the design of the PVC electric wheelchair. We made improvements to the overall design of the wheelchair as well as adding additional features and optimizing specific parts of the wheelchair. We successfully designed a charging circuit that allows us to recharge our battery for continual use rather than a one time use. We also created an OLED display that shows how much battery is remaining in the battery. We completely rewired all the old circuitry and created a design that was more intuitive to follow, as well as label each connection. We were able to integrate the new brushless motors using the hall sensors with the wheelchair, and have a functioning electric wheelchair be able to move forward and make turns.

However, there is still more work that needs to be put into the project. Although we were able to successfully have a functional wheelchair and add on multiple features, we were not able to develop an API for the wheelchair. Doing so will open up many additional possibilities to the project and an easier to use design. The chair also needs to continue to be calibrated for more accurate movements and directions. Another thing that the wheelchair needs is to be tested on multiple terrain, to further strengthen the functionality of the wheelchair.

Overall, we were able to learn a lot from this project. We gained valuable technical skills and learned how to work on a project in a team environment, building up our communication skills. We also discussed ideas and received advice from our sponsor and mentor as we continued to grow throughout this project.

References:

Github: https://github.com/robomakery/pvc-powerchair

DigiKey: https://www.digikey.com/en/articles/techzone

Arduino: https://www.arduino.cc/

HowToMechatronics:
https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/

YouTube: https://www.youtube.com/watch?v=7ph5ov56cfg

APPENDIX A - LIST OF PARTS

- ½ in schedule 40 PVC
- 4 Way Tee connect
- 5 Way Cross connect
- 3 way elbow connect
- 90 degree connect
- Tee connect
- Slip Sling Tee connect
- Gear Box Plate
- Casters
- Plexy-glass
- Brushless Motors
- Joystick Motor Controller
- Joystick Enclosure
- Arduino Uno

APPENDIX B - SAMPLE CODES (Arduino)

i) For Motor control

```
//JOYSTICK PINS
int xPosition = 0;
int yPosition = 0;
int xPin = A1;
int yPin = A0;


//MOTOR DIRECTION CONTROLS
int left_motor = 5;
int right_motor = 7;
//MOTOR SPEED CONTROL

int coord_y = 0;
int coord_x = 0;


int left_speed = 9;          //PWM Pins for sending to Driver Pin
int right_speed = 10;

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

  //activate Joystick pins
  pinMode(xPin, INPUT);
  pinMode(yPin, INPUT);

  //Set up Direction outputs for the motor driver inputs
  pinMode(left_motor, OUTPUT);
  pinMode(right_motor, OUTPUT);

  //Set up Speed outputs for Motor Driver
  pinMode(left_speed, OUTPUT);
  pinMode(right_speed, OUTPUT);
}

void loop() {
  xPosition = analogRead(xPin);
  yPosition = analogRead(yPin);

  coord_y = map(yPosition,0,1024,-35,35);
  coord_x = map(xPosition,0,1024,-35,35);

  Serial.print("Motor X: ");
  Serial.print(coord_x);
  Serial.print(" | Motor Y: ");
  Serial.print(coord_y);
  Serial.print(" | yPosition: ");
  Serial.print(yPosition);
```

```
  Serial.print(" | xPosition: ");
  Serial.println(xPosition);

  if (xPosition < 530 && xPosition > 500 && yPosition < 530 && yPosition > 500)
  {
    //Do Nothing
    motorStop();
    Serial.println("STOP");
  }
  else if (xPosition < 530 && xPosition > 470 && yPosition < 500)
  {
    //Forward
    motorBackward(left_motor,left_speed,abs(coord_y));
    motorBackward(right_motor,right_speed,abs(coord_y));
  }
  else if (xPosition < 530 && xPosition > 470 && yPosition > 500)
  {
    //Backwards
    motorForward(left_motor,left_speed,abs(coord_y));
    motorForward(right_motor,right_speed,abs(coord_y));
  }
  else if (yPosition < 530 && yPosition > 470 && xPosition > 500)
  {
    //Turn right
    motorForward(left_motor,left_speed,abs(coord_x));
    motorBackward(right_motor,right_speed,abs(coord_x));
  }
  else if (yPosition < 530 && yPosition > 470 && xPosition < 500)
  {
    //Turn left
    motorBackward(left_motor,left_speed,abs(coord_x));
    motorForward(right_motor,right_speed,abs(coord_x));
  }


}


void motorForward(int dir,int mot_spd, int spd){          //mot_spd = control which motor to move: left or
right
  digitalWrite(dir,LOW);
  analogWrite(mot_spd,spd);
}

void motorBackward(int dir,int mot_spd, int spd){
  digitalWrite(dir,HIGH);
  analogWrite(mot_spd,spd);
}


void motorStop(){
  analogWrite(left_speed,0);
  analogWrite(right_speed,0);
}
```

ii) Battery indicator on OLED

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define ADJ_PIN A0
#define OLED_RESET 4
Adafruit_SSD1306 Display(OLED_RESET);

int r = 0;
int i = 0;

void setup()   {

  Serial.begin(9600);
  pinMode(ADJ_PIN, INPUT);
  Display.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C (for the 128x32)

  Display.clearDisplay();
  Display.display();
  delay (1000);

  DrawTitles();

}
void loop() {

  r = analogRead(ADJ_PIN);
  r = r / 7.98;

  Display.setTextSize(2);
  // note set the background color or the old text will still display
  Display.setTextColor(WHITE, BLACK);
  Display.setCursor(0, 33);
  Display.println(Format(r * 7.99 / 204.6, 3, 2));

  //draw the bar graph
  Display.fillRect(r, 50, 128 - r, 10, BLACK);
  Display.fillRect(3, 50, r, 10, WHITE);

  for (i = 1; i < 13; i++) {
    Display.fillRect(i * 10, 50, 2, 10, BLACK);
  }

  // now that the display is build, display it...
  Display.display();
}
void DrawTitles(void) {

  Display.setTextSize(2);
```

```
  Display.setTextColor(WHITE);
  Display.setCursor(0, 0);
  Display.println("Bar Graph");

  Display.setTextSize(1);
  Display.setTextColor(WHITE);
  Display.setCursor(0, 19);
  Display.println("Measured Volts");
  //Display.println("Random number");
  Display.display();

}
String Format(double val, int dec, int dig ) {

  int addpad = 0;
  char sbuf[20];
  String fdata = (dtostrf(val, dec, dig, sbuf));
  int slen = fdata.length();
  for ( addpad = 1; addpad <= dec + dig - slen; addpad++) {
    fdata = " " + fdata;
  }
  return (fdata);

}
```