Omnidirectional Unmanned Manipulation Robot with Kinect Control

Progress Update 2

December 1st, 2018

Summary

In this progress update, we will show you the mechanics of our first prototype of the robot. We will also show you a working circuit that controls three DC motors and a gripper, as well as a circuit that utilizes an ultrasonic proximity sensor to control an LED. In addition, we will present code snippet for Arduino board, which controls the motors, and the code for Kinect, which reads human gesture and communicates with Arduino board. Finally, we will discuss the issue we encountered so far and things to be done before final presentation.

1) Mechanics

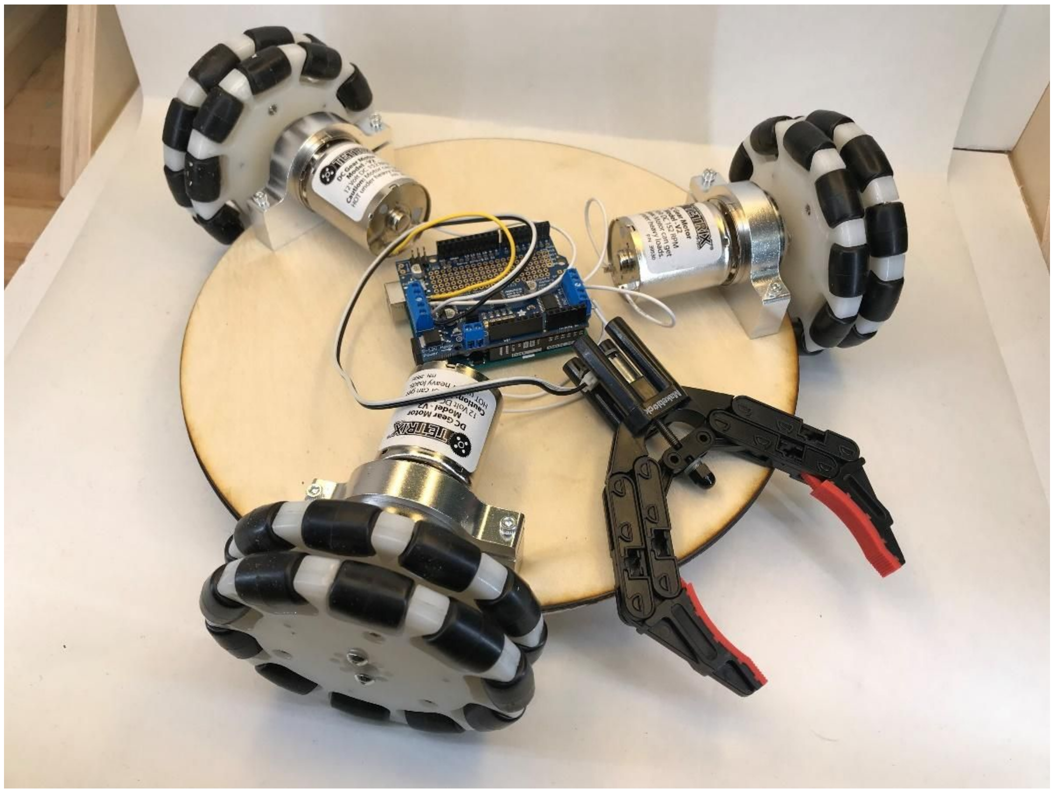


Figure 1: Isometric View of the Robot Assembled

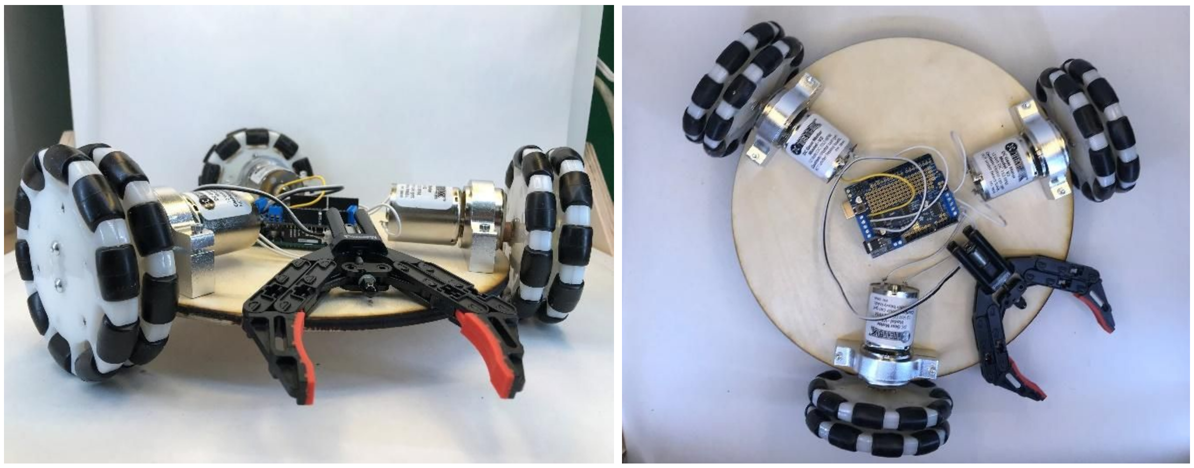


Figure 2: Side and Top View of the Robot Assembled

We require the center plate to be able to hold the motor-wheel sets and it should be lightweight. That's why we chose 6mm thick wood as the material for the center plate. In addition, we can use laser cutter to easily modify the shape of the wood plate.

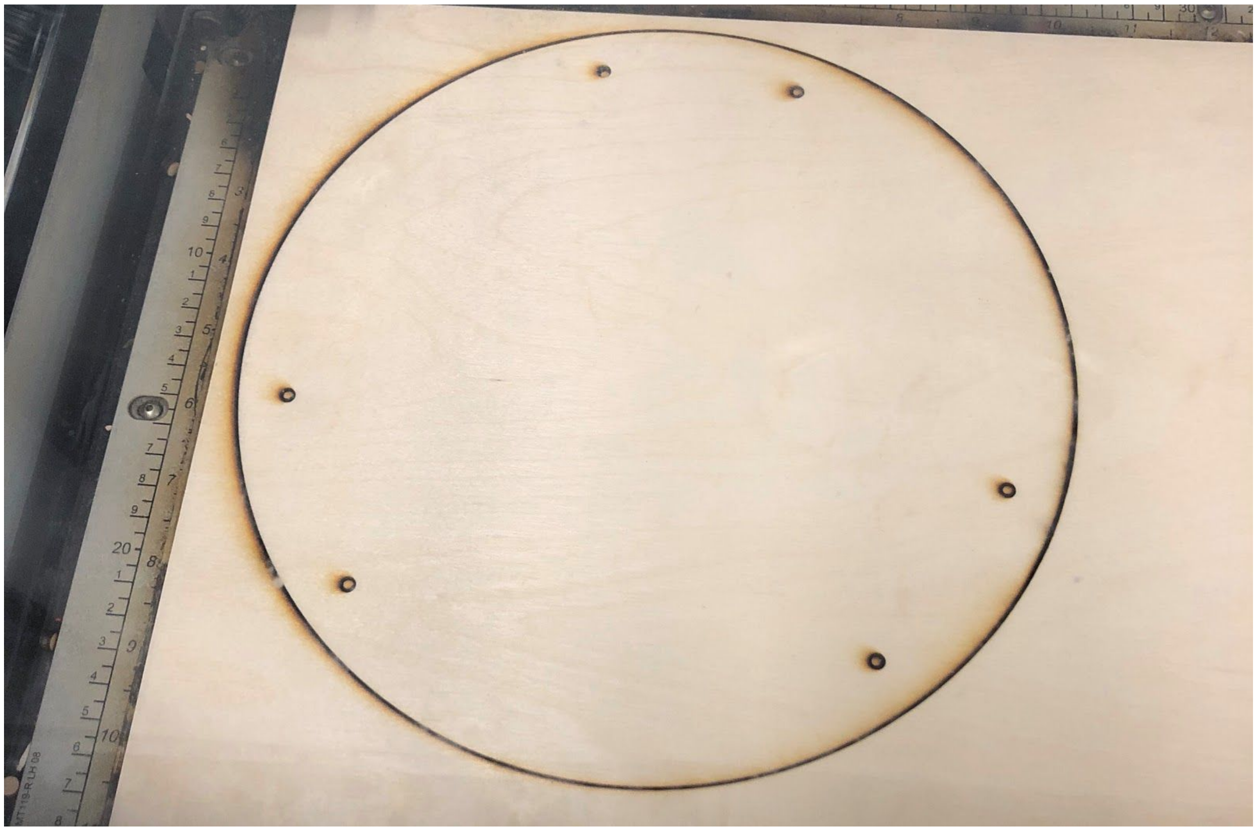


Figure 3: Laser-cutting the Center Plate

We used a metal DC motor mount to secure the motor-wheel set on the plate. The motor mount we used is thick and sturdy enough to ensure that the motors are inline with the plate.

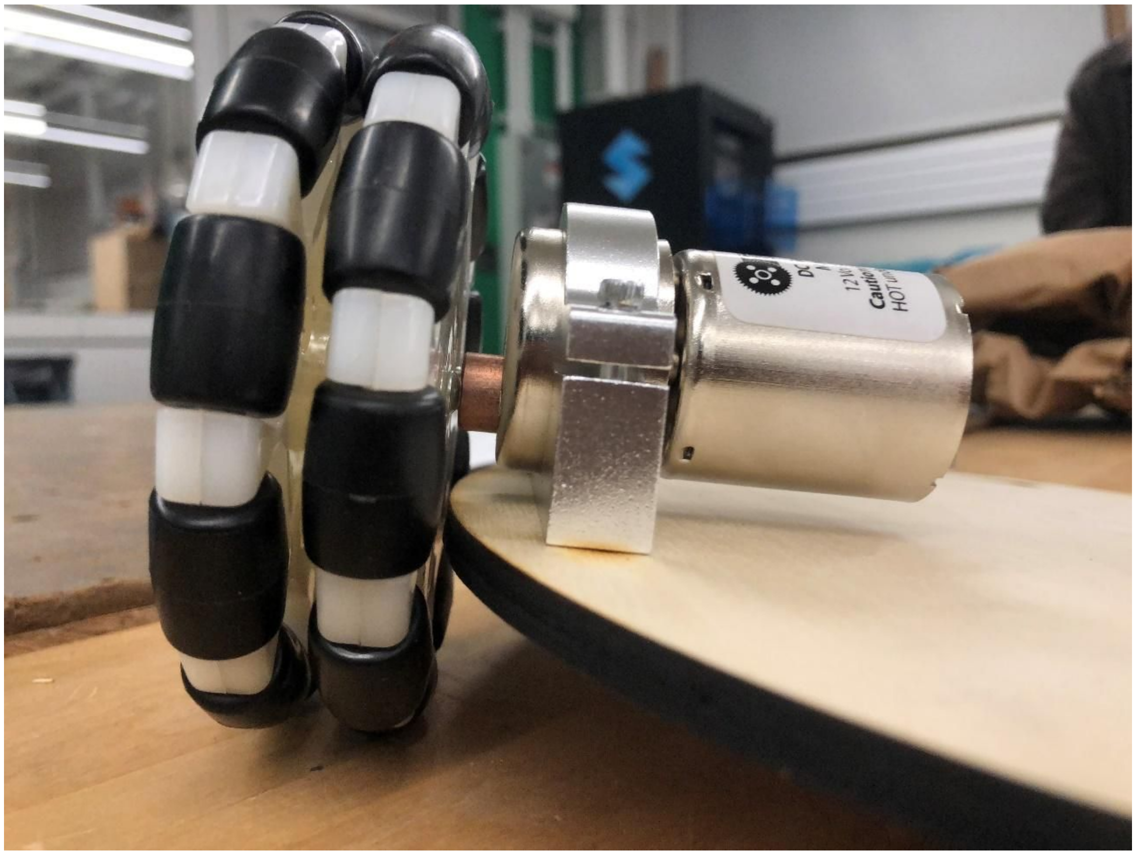


Figure 4: Motor Mount Side View

2) Electronics

Video demo is placed at the end of section 3) Code. For main body of the robot, we used one Arduino UNO and an Adafruit Motor Shield. We have three wheels connected to three DC motors and one gripper, which is also connected to a small DC motor.

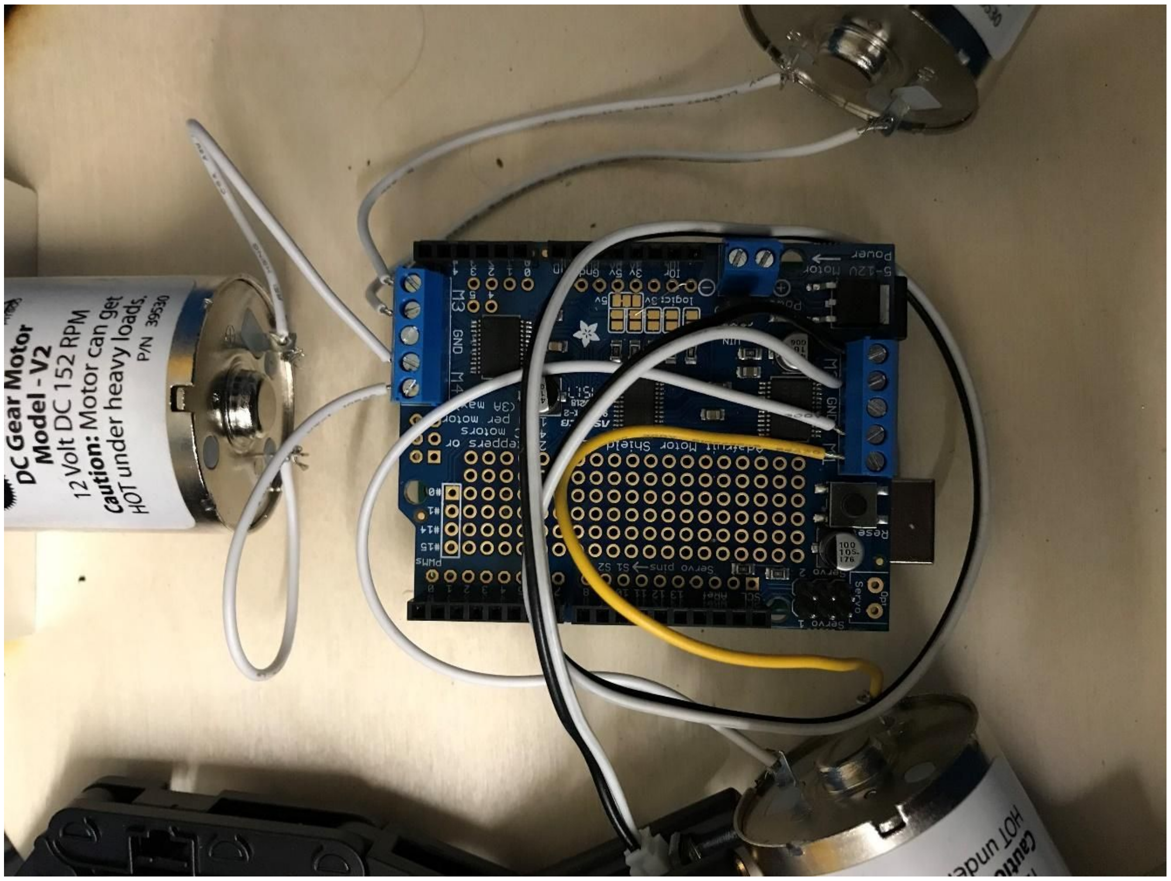


Figure 5: Wheels and gripper working circuit

One Adafruit motor shield supports at most 4 DC motors, so we used M1, M3, and M4 for the DC motors for wheels, and M2 for the gripper. The DC motors for wheels support 5V to 12V voltage. It has free spin RPM of 152, and stall torque of 2.25Nm. Even though we are powering the motors using 9V voltage, they can provide enough torque and speed to the entire robot, which we estimated to be no heavier than 2.5Kg.

The gripper is equipped with an N20 DC gear motor to allow the gripper to open and close.

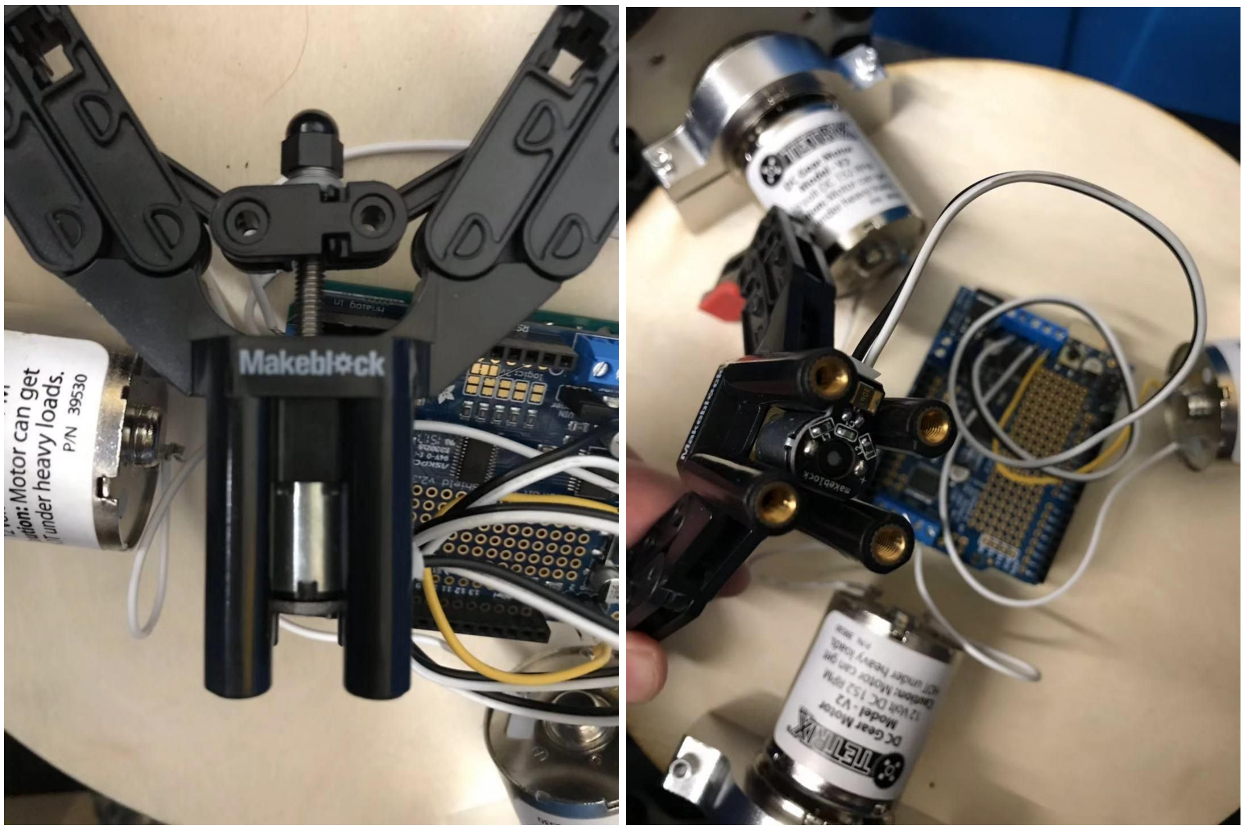


Figure 6: DC gear motor

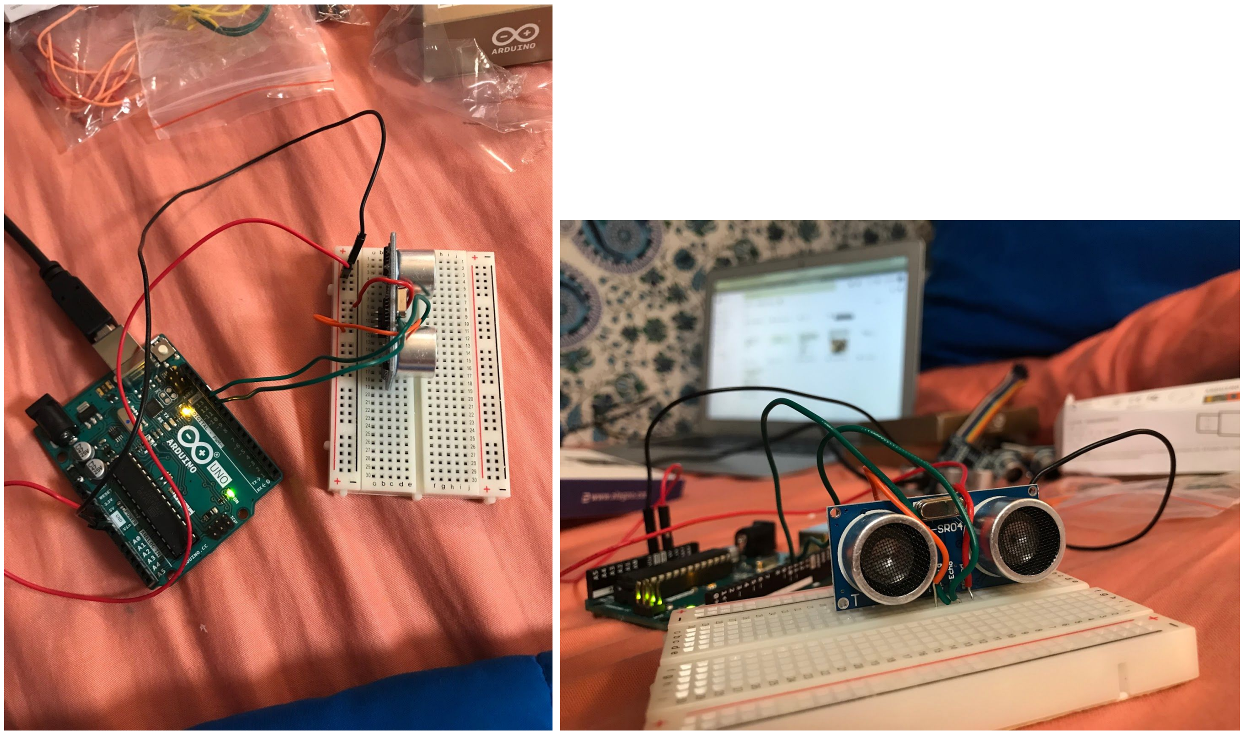


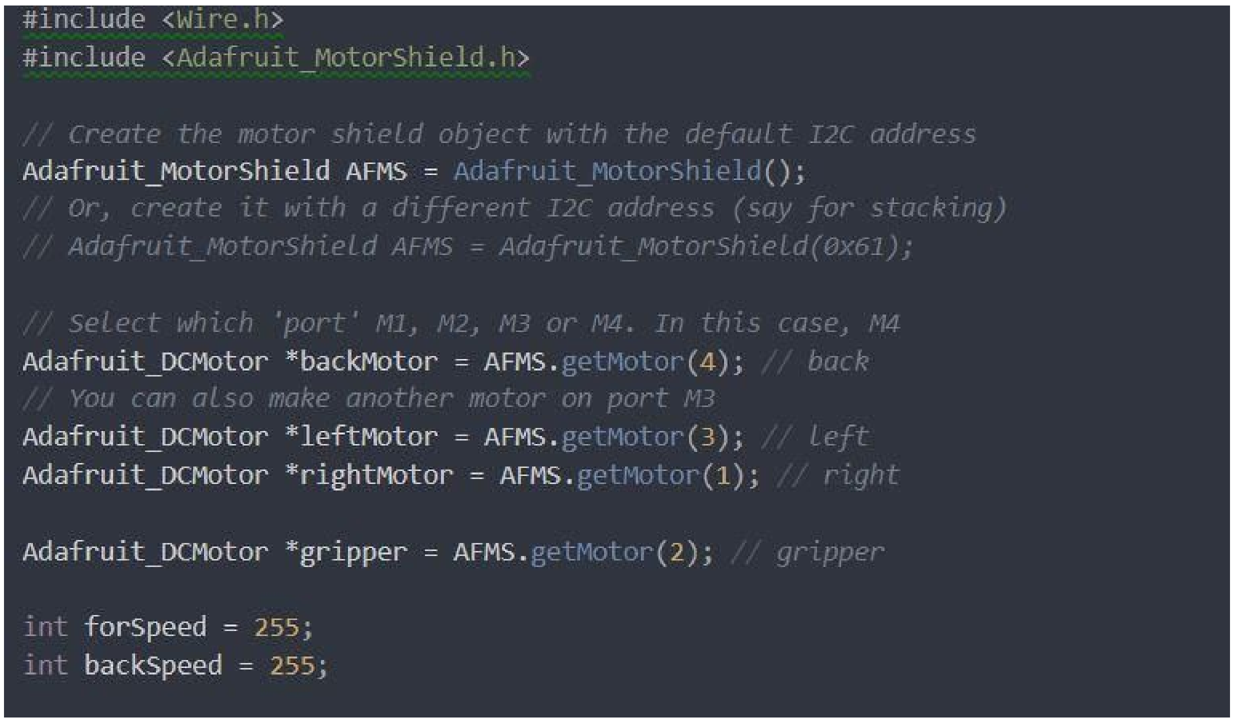
Figure 7: Proximity Sensor

3) Code

This time, we developed Arduino code to control three DC motors and python code to allow Kinect to send human gesture signal to Serial port of Arduino. We created a GitHub repo to help us manage the version of the code and allow us to collaborate easily. Link: https://github.com/shuy98/24354TP

(Comprehensive source code for Arduino can be found using the following link: https://github.com/shuy98/24354TP/tree/master/progress_2)

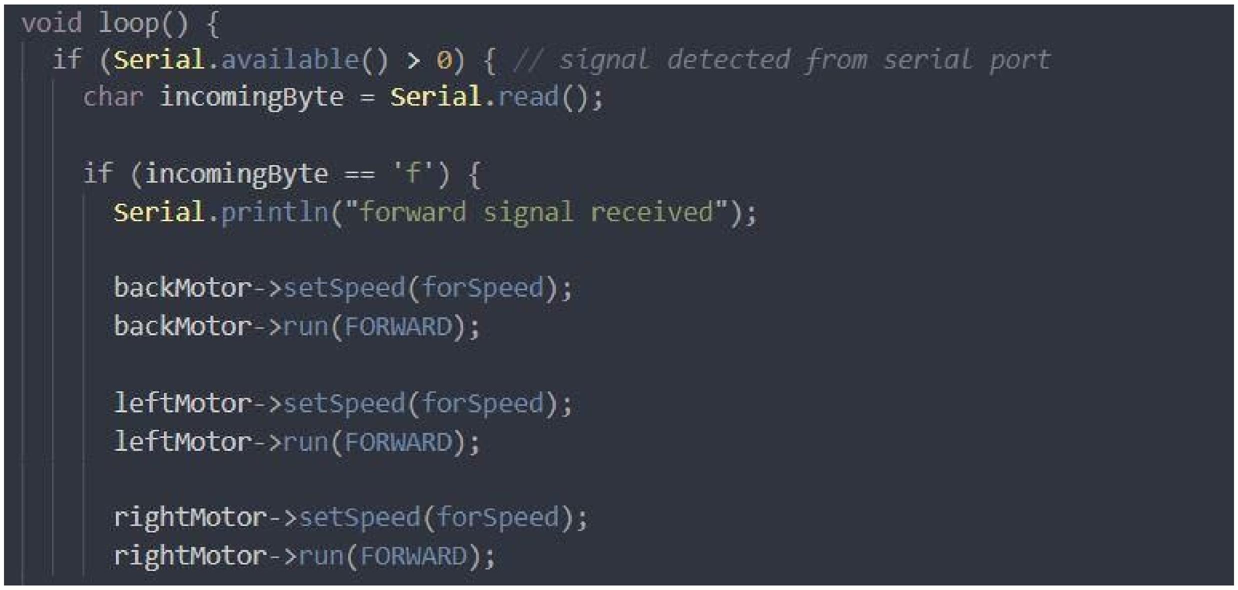
We used Adafruit motor shield library to control DC motors. Since we are using three DC motors for wheels and one DC motor for gripper, we initialized all these four motors.



```
#include <Wire.h>
#include <Adafruit_MotorShield.h>

// Create the motor shield object with the default I2C address
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
// Or, create it with a different I2C address (say for stacking)
// Adafruit_MotorShield AFMS = Adafruit_MotorShield(0x61);

// Select which 'port' M1, M2, M3 or M4. In this case, M4
Adafruit_DCMotor *backMotor = AFMS.getMotor(4); // back
// You can also make another motor on port M3
Adafruit_DCMotor *leftMotor = AFMS.getMotor(3); // left
Adafruit_DCMotor *rightMotor = AFMS.getMotor(1); // right

Adafruit_DCMotor *gripper = AFMS.getMotor(2); // gripper

int forSpeed = 255;
int backSpeed = 255;
```

Figure 8: Initialize DC motors from Adafruit Motor library

In this loop, it actively listens for signal from Serial port (Kinect interface sends signal to the Serial port). If Arduino receives the signal and it matches the condition, in this case it's character 'f', it will turn on the motor. We turn three motors in the same direction to make the robot turn clockwise/counter-clockwise. To make it move forward and backward, we fix the back wheel and turn the other two of the wheels in opposite directions.

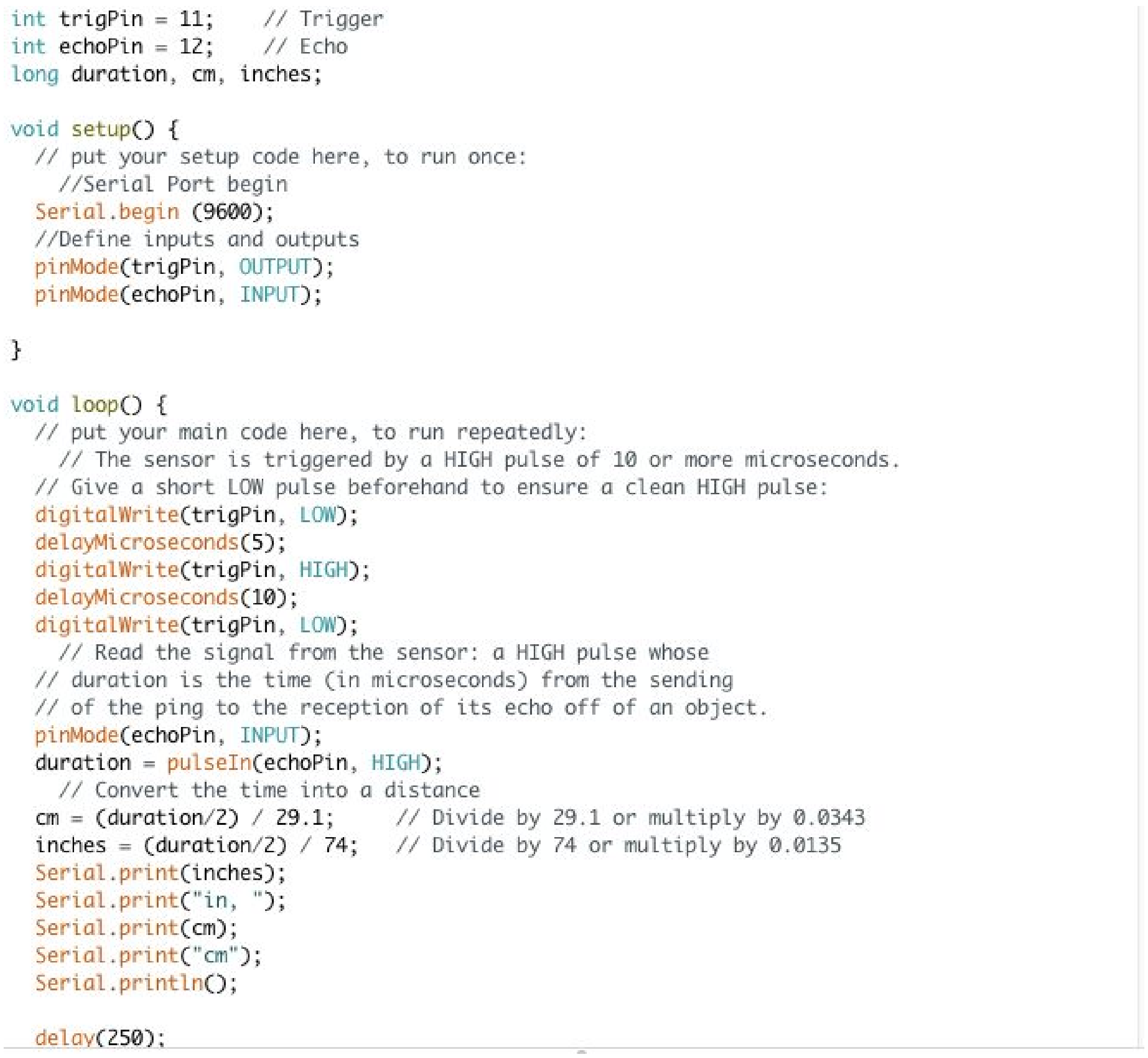


```
void loop() {
  if (Serial.available() > 0) { // signal detected from serial port
    char incomingByte = Serial.read();

    if (incomingByte == 'f') {
      Serial.println("forward signal received");

      backMotor->setSpeed(forSpeed);
      backMotor->run(FORWARD);

      leftMotor->setSpeed(forSpeed);
      leftMotor->run(FORWARD);

      rightMotor->setSpeed(forSpeed);
      rightMotor->run(FORWARD);
```

Figure 9: Listen for signal and turn on the motor

We didn't make any major changes to the Kinect side code except some modularity improvements and added gestures, details have been discussed in progress update 1. Check out the source code from this link:
https://github.com/shuy98/24354TP/blob/master/run.py

Using this code, we were able to instruct the robot using Serial port and allow it to move forward, backward, and turn clockwise/counter-clockwise.



```
int trigPin = 11;    // Trigger
int echoPin = 12;    // Echo
long duration, cm, inches;

void setup() {
  // put your setup code here, to run once:
    //Serial Port begin
  Serial.begin (9600);
  //Define inputs and outputs
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

}

void loop() {
  // put your main code here, to run repeatedly:
    // The sensor is triggered by a HIGH pulse of 10 or more microseconds.
  // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
    // Read the signal from the sensor: a HIGH pulse whose
  // duration is the time (in microseconds) from the sending
  // of the ping to the reception of its echo off of an object.
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
    // Convert the time into a distance
  cm = (duration/2) / 29.1;     // Divide by 29.1 or multiply by 0.0343
  inches = (duration/2) / 74;   // Divide by 74 or multiply by 0.0135
  Serial.print(inches);
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();

  delay(250);
```

Figure 10: Code which prints out distance an object is away from the proximity sensor

Serial monitor displays inches and cm away an object is.

Video demo:

- Robot moving forward and backward controlled by PC. https://drive.google.com/file/d/1mRkVVeUIwBoWFXJ1S_hFO90CJ1UFN2j4/view?usp=sharing

- Robot turning clockwise/counter-clockwise controlled by PC.
  https://drive.google.com/file/d/1EuiUJG095ITJh76e2yqfx4a8CCoWp9M_/view?usp=sharing
- Gripper open and close to grip an object
  https://drive.google.com/file/d/1xRY6WJ7tjuOVHeXRk6ILYwNfiMLbZ2xo/view?usp=sharing
- Proximity Sensor detecting distance in inches and cm

  https://drive.google.com/open?id=1_KX3vb1ToSKIsPZffzp7Y9HCyz7Yo1x4

4) Reflections

One unexpected issue we encountered in this prototype assembly is that a single 9V battery does not have enough energy to drive three large 12V DC motors and one small DC motor simultaneously. It's not an issue if we power the motor shield using the cable, which draws current either from USB port or power outlet. However, since we will make the robot wireless (remote control) eventually, we have to find a way to power these motors with batteries. Therefore, we are considering using six 1.5V AA battery pack, which will provide 9V in total to power the motor shield. For the Arduino UNO itself, we will keep using a single 9V battery since it only powers one single Bluetooth module, which does not draw much energy compared with four DC motors.

One expected issue is that the robot does not move perfectly straight in forward and backward motion. Possible solutions are:

- Add weight to one side to make the robot balanced.
- Tweak the speed of the DC motors.
- Mechanical changes on the structure.

Things to be done before Demo day:

- Address the issues mentioned above.
- Incorporate a Bluetooth module to allow remote control.
- Gripper attachment mechanical design.
- Kinect gesture improvements.
- Proximity sensor equipped gadgetry.