Yihao Qian

Team A: Aware

Teammates: Amit Agarwal Harry Golash

Menghan Zhang Zihao (Theo) Zhang

ILR01

Oct.14, 2016

Individual Progress

For sensors and motors lab, I was in charge of the servo and force sensor, also I was in charge of integrating part of the code with teammates, moreover, I wrote the code and found out the transformation function of the distance sensor. The circuit for our system could be found in Figure 1.

Servo and force sensor

What I did is to let the force sensor to control the rotation of the servo motor. The main idea is to let the force sensor detect a force and output an analog voltage that can be input to the Arduino pin, then averaging the value and map the voltage to 0~180 degree. Finally, the Arduino board sends the command in order to let the servo motor to rotate to the corresponding degree.

I will introduce 3 main parts in the following report:

Force Sensor:

Force Sensor's resister decrease as the force increase (almost linear). I designed a circuit, that series sensor, the 200 Ω resister with the 5V power. Then read the analog signal from the pin, then use the map function to map the input value to the value used for motor. The circuit can be found in Figure 2.

Mean Value:

The input from the force sensor is vibrate, which means that the servo has to respond to the signal vary frequently. In order to smooth the control signal, I tried to average the value of input from the force sensor.

The main idea is to create a buffer, for example the size of the buffer is around 10 size. For each time, we are going to read the input, and abandon the oldest input. Then we average all the value in the buffer, providing a smooth output control signal.

Servo Motor:

Arduino has already existed libraries for servo motor, however in order to accurately control the servo motor, I have to fine tuning my program. Instead of using the not accurate command myservo.write(val), I use the function called myservo. writeMicroseconds to precisely control the angle of the Servo motor. In my experiment, I find out when the digital command output val is 500, the servo motor's angle is 0 degree, when the output val is 2400, the servo motor's angle is 180 degrees. Mapping the input signal to 500-2400 would achieve a much more precise control of the servo motor.

Integrating the force sensor and potentiometer

The other thing I did for our project is to integrate the code Menghan with mime. The main idea is to use a bottom to control the system state, and switch between different functions. I used the knowledge I learnt from the last Arduino assignment, debounce the bottom. Also the code adopts a switch function, so it would be much easier for integrating the code from other team members.

Calculating the transform function of distance sensor

I chose to test the transform function of the distance sensor. I would like to compute the relationship between the output voltage and the ground truth detect distance. The circuit is quite simple, the 5v voltage power, the ground, and the signal output that is linked to the pin on the Arduino board. I just use the analog read to read the voltage and use the ruler to measure the corresponding distance.

Challenges:

Resister missing

When combing the circuit with Menghan, she added a new button on the board in order to switch the state of the system. After I combined the code and download it to the board, the switch didn't work. The system state changed almost randomly. At first I thought it was due to the bug of the program (maybe something wrong with the debouncing program), I changed two ways of debouncing, the basic one and the interrupt one. No matter what I tried, the bottom didn't work. So I started to think this problem might be caused by the hardware and software. So I download the last assignment code to both the TA board and our board, the TA board works perfectly fine, however our board still didn't work. It was at that time, I started to check whether the wire is correct or not. After thoroughly checking the board, I found out one of the resister that should be linked to the switch bottom is missing. After adding the resister, the bottom worked perfectly.

Teamwork:

Since there are 5 people in our team. We split the sensor and motor equally to each team mates. Amit is responsible for the DC motor, and the distance sensor. I am in charge of the force sensor and servo. Menghan and I combined our code together. I wrote the code for to test the transfer function. Zihao is responsible for the stepper motor and IR sensor. Menghan is responsible for Potentiometer and servo motor. Harry is in charge of the GUI.

Figures:

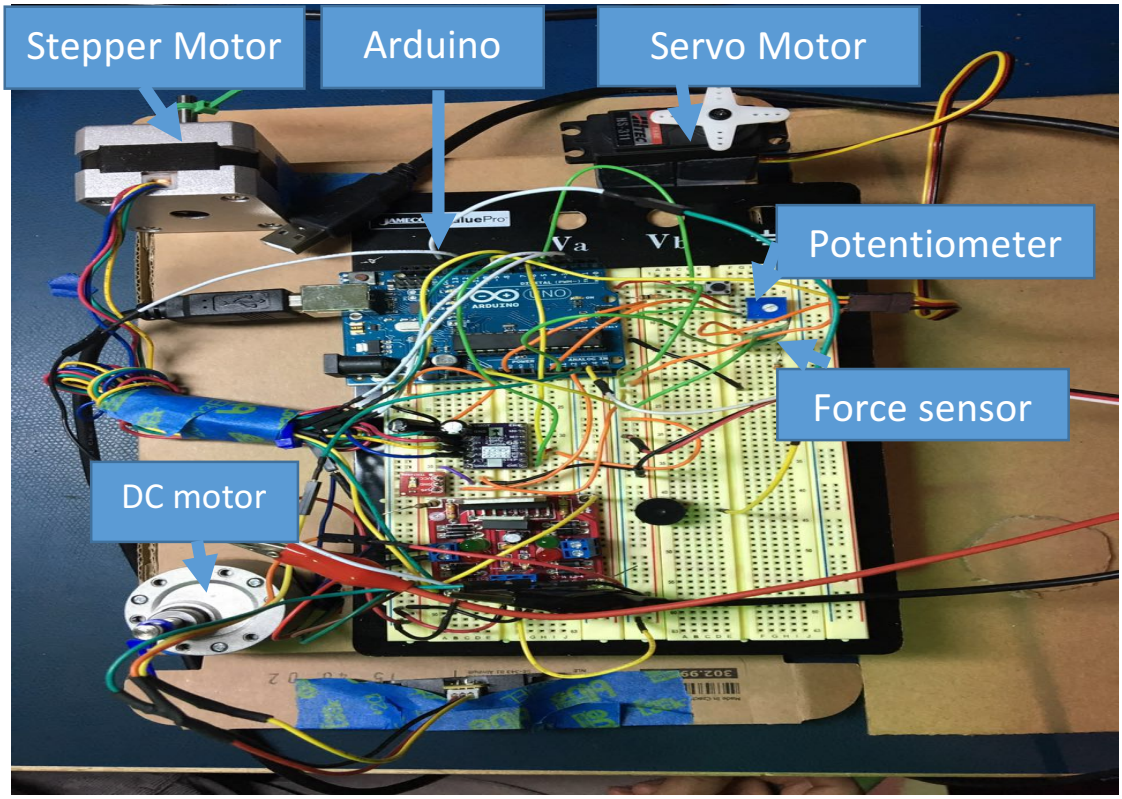

Figure 1. The final board

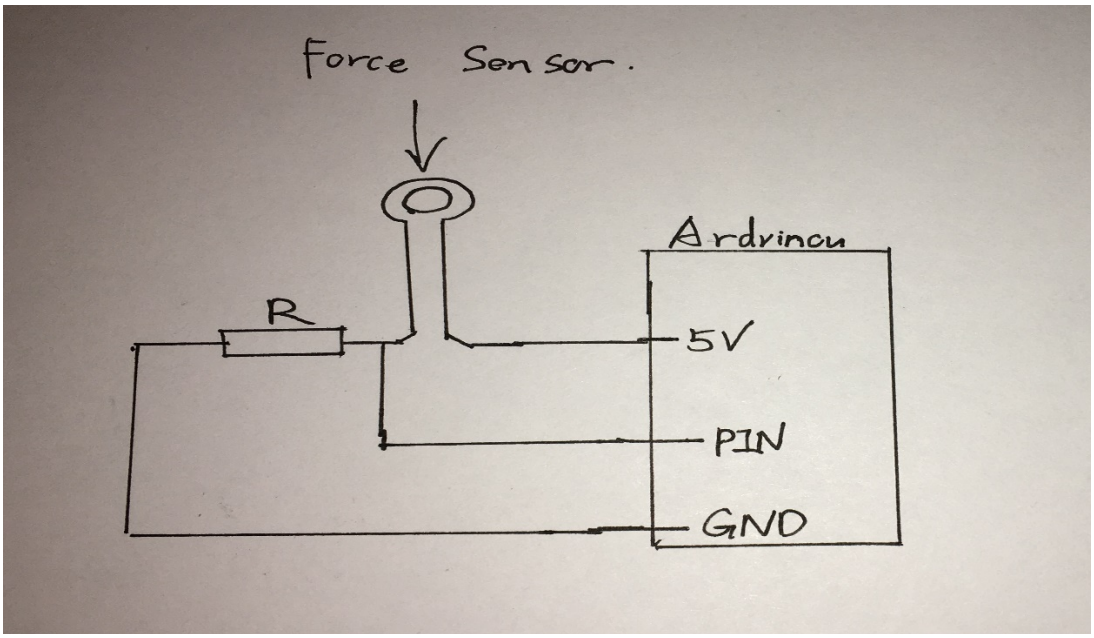

Figure 2. The circuit for the force sensor

Plans:

From now on, we are going to begin to build our own perception system.

Harry and Amit will work on the hard-ware design, specifically, on designing the rigid mounting rat that can fixed our sensors.

Zihao and Menghan is also contributing their time to test the performance of the camera and radar. They are going to test the performance of these sensors in different environment, aiming to find the strength and weakness of those sensors.

I already know something about the stereo camera 3-D reconstruction, so it may help for our project. Moreover, I already read some paper about the object detection, a method called faster r-cnn, which would take 0.2s to detect the interest object in the image could be used to build the prototype of the project. Also, I am going to implement this network in C++ version. Combined with the high performance GPU, the algorithm is hopefully work fast enough to work it real time, I am trying to learn how to write the code in Caffe through reading the tutorials online.

The potential challenges in the future is as follow:

I can't accurately estimate how much we depend on the hardware. I know how to do the 3-D reconstruction in the static or low speed environment, however, I am not so sure whether we could accurately build the 3-D map on a car with 80km/h speed. Moreover, 3-D reconstruction need the support from the GPU, object detection needs the support from the GPU, I am afraid that our hardware system can't handle such a demanding task.

Code:

```
//Force sensor and potentiometer control the servo motor

#include <Servo.h>

Servo myservo;

// create servo object to control a servo

const int numReadings = 10;

int potpin = A0;  // analog pin used to connect the potentiometer

int val;    // variable to read the value from the analog pin
```

```
int readings[numReadings];      // the readings from the analog input
int readIndex = 0;              // the index of the current reading
int total = 0;                  // the running total
int average = 0;

int potpinf=1;
int valf;
int i;
const int numReadingsf = 10;
int readingsf[numReadingsf];      // the readings from the analog input
int readIndexf = 0;              // the index of the current reading
int totalf = 0;                  // the running total
int averagef = 0;



//debouncing
int buttonPin = 2;    // the number of the pushbutton pin


// Variables will change:

int buttonState;             // the current reading from the input pin
int lastButtonState = LOW;   // the previous reading from the input pin

// the following variables are unsigned long's because the time, measured in miliseconds,
// will quickly become a bigger number than can be stored in an int.
unsigned long lastDebounceTime = 0;  // the last time the output pin was toggled
unsigned long debounceDelay = 50;     // the debounce time; increase if the output flickers
```

```
int sysstate=0;




void setup() {
    //pinMode(buttonPin, INPUT);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
Serial.begin(9600);
pinMode(potpinf,INPUT);
pinMode(buttonPin, INPUT);
  myservo.attach(9);

  for (int thisReadingf = 0; thisReadingf < numReadingsf; thisReadingf++) {
    readingsf[thisReadingf] = 0;}

 for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings[thisReading] = 0;
  }

}




void portential()
{

  total = total - readings[readIndex];
  val = analogRead(potpin);              // reads the value of the potentiometer (value
between 0 and 1023)
```

```
  readings[readIndex] = map(val, 0, 1023, 500, 2400);      // scale it to use it with the
servo (value between 0 and 180)
   // add the reading to the total:
  total = total + readings[readIndex];
  // advance to the next position in the array:
  readIndex = readIndex + 1;
   if (readIndex >= numReadings) {
    // ...wrap around to the beginning:
    readIndex = 0;
    Serial.println(average);
    myservo.writeMicroseconds(average);
  }

  // calculate the average:
  average = total / numReadings;
  // send it to the computer as ASCII digits



  //Serial.println(val);

  //myservo.write(val);               // sets the servo position according to the scaled value
  delay(10);                        // waits for the servo to get there
  }


  void force()
  {
    valf=analogRead(potpinf);
```

```
  Serial.println(map(valf, 0, 120, 0,180));
  valf = map(valf, 0, 120, 500, 2400);
  totalf = totalf - readingsf[readIndexf];
  readingsf[readIndexf] = valf;
  totalf = totalf + readingsf[readIndexf];
  readIndexf = readIndexf + 1;
  if (readIndexf >= numReadingsf) {
  readIndexf = 0;
 }
 averagef = totalf / numReadingsf;
 myservo.writeMicroseconds(averagef);

  delay(2);
   }


void loop() {
 int reading = digitalRead(buttonPin);


 // check to see if you just pressed the button
 // (i.e. the input went from LOW to HIGH),  and you've waited
 // long enough since the last press to ignore any noise:

 // If the switch changed, due to noise or pressing:
 if (reading != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
 }
```

```
  if ((millis() - lastDebounceTime) > debounceDelay) {
    // whatever the reading is at, it's been there for longer
    // than the debounce delay, so take it as the actual current state:

    // if the button state has changed:
    if (reading != buttonState) {
      buttonState = reading;

      // only toggle the LED if the new button state is HIGH
      if (buttonState == HIGH) {
        sysstate=(sysstate+1)%2;
      }
    }
  }
 lastButtonState = reading;

switch(sysstate)
{case 0:
force();
Serial.print("force");
break;
case 1:
portential();
Serial.print("portential");
break;
default:
break;
}
```

```
delay(2);


}

//Test the transfer function of distance sensor
int sensorPin = A0;
double sensorValue = 0;  // variable to store the value coming from the sensor
void setup() {
Serial.begin(9600);
}

void loop() {
sensorValue = analogRead(sensorPin);
Serial.print(sensorValue);
delay(2);
}
```