Sensors and Motor Control Lab

Individual lab report #1 October 16, 2015

Menghan Zhang

TeamA
Amit Agarwal
Harry Golash
Yihao Qian
Zihao Zhang

Individual progress

a) Used potentiometer to motive the servo motor ,
b) Controlled the position of servo motor by rotating the potentiometer to get the max and min degree of servo:
    i. I searched online and then found that there are tutorials and library for potentiometer and servo motor. After learned about that then change the certain pin and to satisfy teams arrange.
    ii. To increase the control accuracy, I choose to use writeMicroseconds() to instead of write.
    iii. To smooth and decrease the impact of noise , I made an average every 100ms then export one result to control the motor
c) Combined the potentiometer with force resistance sensor using button to change the state then control the servo motor to rotate to a certain position. Because of needing further integration with other's sensors and codes, so I discuss with the team member first about which pin they used for their sensors and define the variable in understandable and specific name to avoid using same pin and same name for variable, try to encapsulate the function of sensors in a function and use switch to change the state

Challenges

a) The problem when I design the circuit and test for the results is that the servo motor keep spinning in a certain degree and can not be controlled by the potentiometer, then I go over the circuit again and change the position of the potentiometer on bread board then it works. The reason must be the problem with the bread board, they supposed be connected but actually the connection is broken.
b) There was also a problem when I combine the circuit with Yihao's force sensor, we use the button to change the state and because of the several mistakes and problem of hardware, it took us long time to get the correct results.
c) The problem here is the button can not work decently, at first we thought it's because of the debouncing did not work, then we go over several methods to debounce but still had no effect. Then we tested the voltage of the button and find out it have some problem here. Then I realized that's because I forgot the 10k ohm pull down resistor. After I add the resistor it can work decently.
d) Challenge in the future

The problem is that I need to be more careful about the circuit, I thought I was finished the circuit after being interrupted by something else but actually, I need double check the circuit before I start coding and debug.

Teamwork

a) Combined the circuit and code with Yihao.(Appendix)

b) Measured and calculated the transfer function of the Infrared Proximity Sensor with Zihao and Yihao

Figures

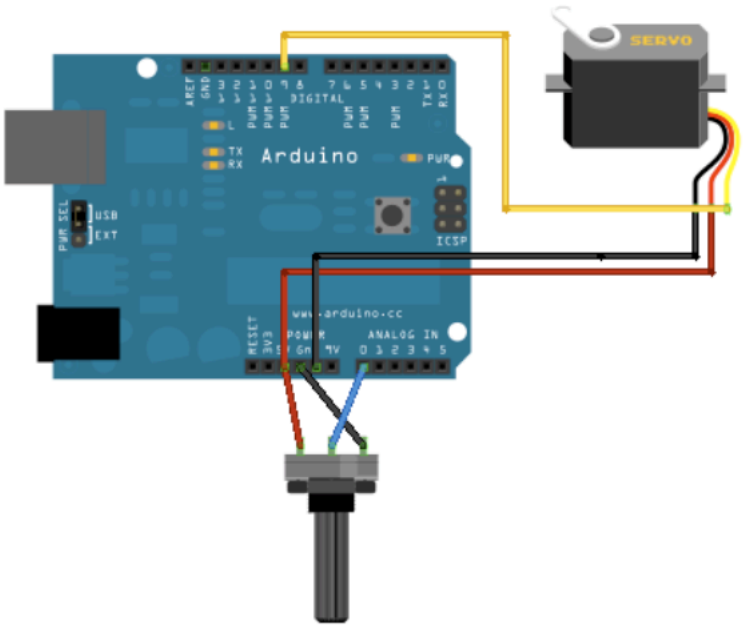


Figure1 : circut

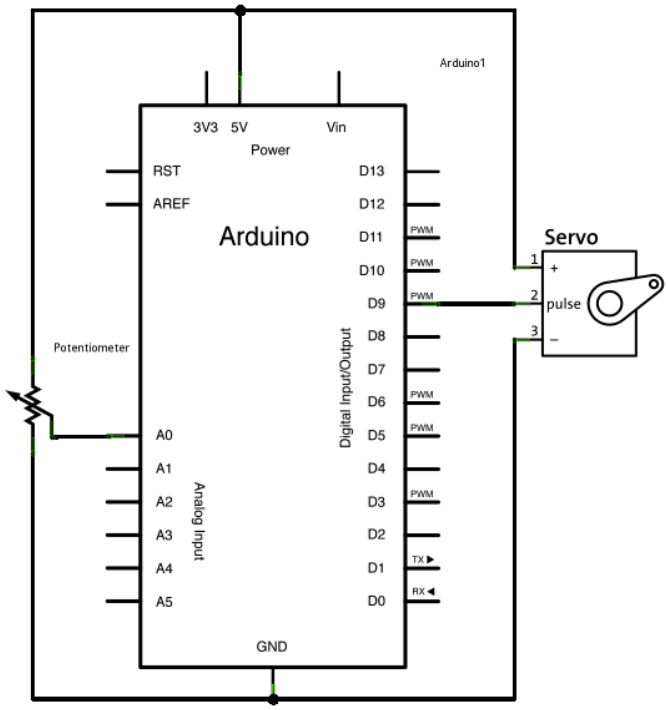


Figure2 Skematic

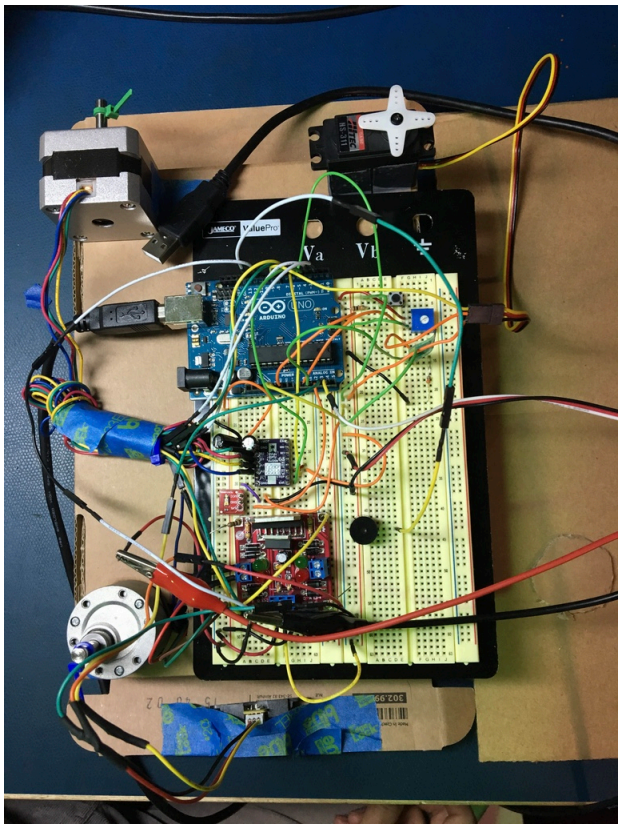
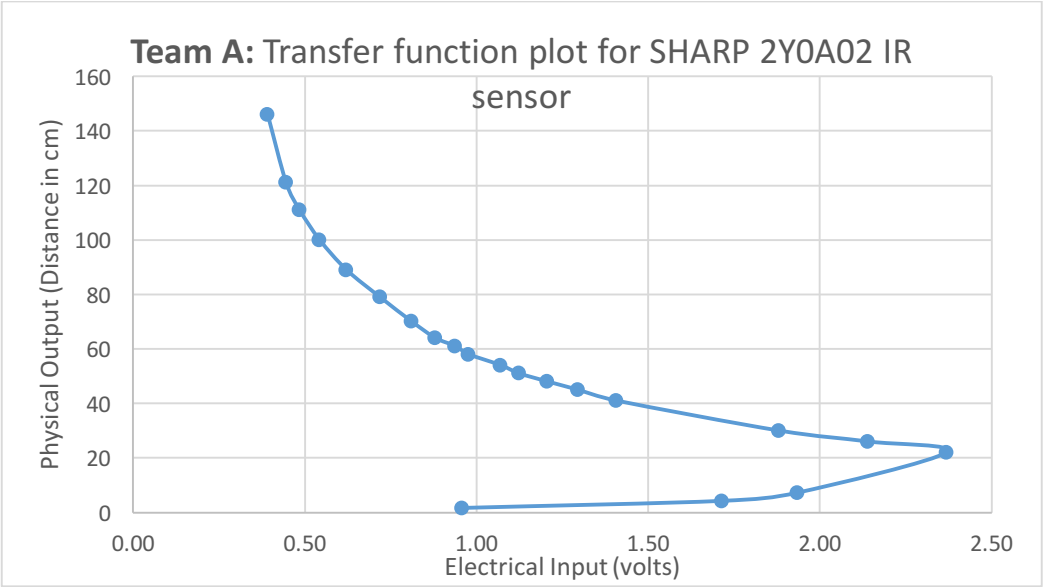


Figure 3 Final version of the circuit



Plans

a) From this sensors and motors lab, I realized that my knowledge about the motor and circuit is limited, lacking of the ability to find the problem in the short time, so from now

on I need to review the knowledge were learned in undergraduate school and make more practice about that.

b) For object detection, I need to learn some basic method for machine learning first, then try to understand how Faster R-CNN work so that I can implement that in the future.

c) For coding ability, I need to learn Python and C++ again, practice more and understand the truly meaning behind class and Inherit in C++.

Appendix : Code combined potentiometer and force resistor

```
#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);

}
```