Dawn Haken
Yuchen Li
Quoc Pham

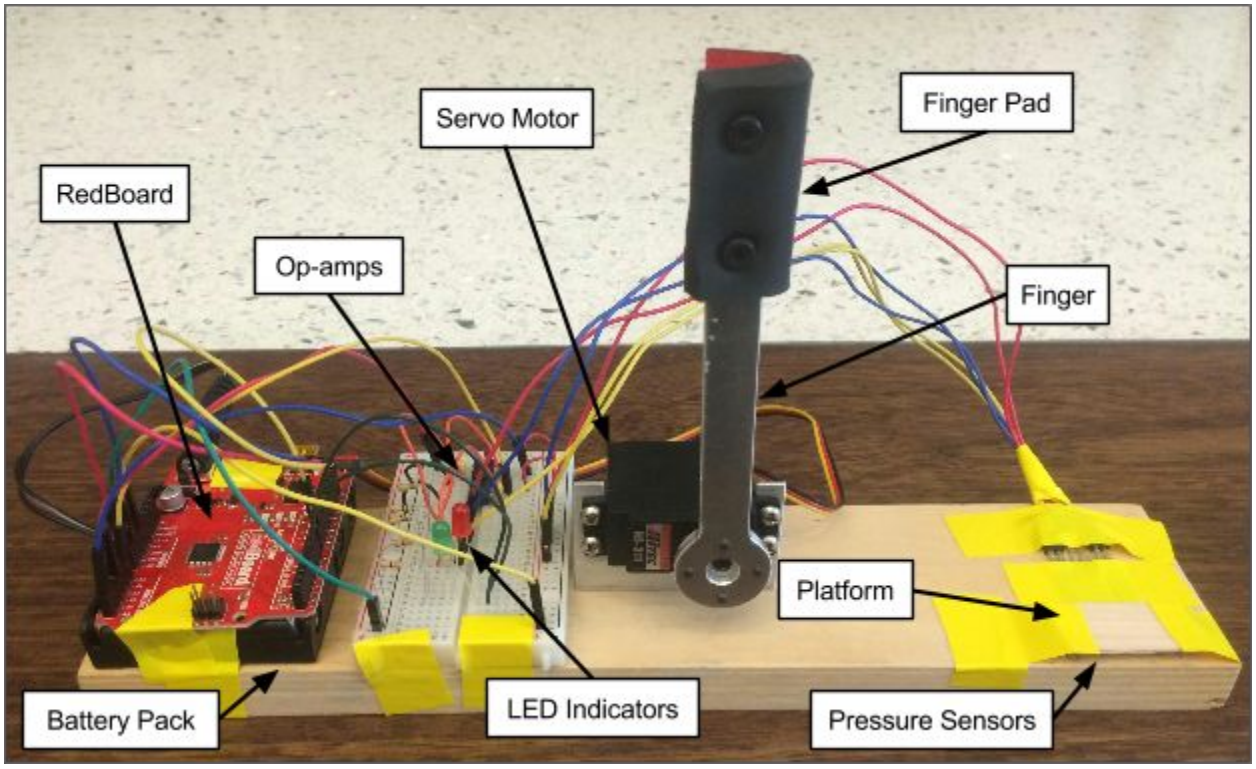
ECE 110 Honors Section Project:
Robotic Finger with Pressure Feedback

Introduction

The inspiration and motivation for our project was the human finger. We wanted to try and mimic how the brain determines the grip on the large variety of objects which allows it delicately hold a grape as well as carry heavy objects. Much like the pressure sensors in our finger, we decided to use a force sensor to feedback information about the object's weight along with the pressure coming from the robotic finger. The robotic finger in this setup was a servo motor with an arm attached to it. Our design solution to mimic the human finger was to program the RedBoard to control the servo so that changes in the weight caused by the servo's movement will impede the servo's next movement. This way the servo will grip only to an extent determined by the weight of the object. The next step would be to have the setup continually read the forces on the sensors as we shift the robotic finger from a horizontal position to a vertical position. Vertical positioning would cause a decrease in pressure from the weight of the object, and in turn activate the servo to "grip" tighter, so that the readings from the force sensor are held constant. This is an interesting way to keep a steady grip on an object without the use of an accelerometer.

Design

For our servo motor we chose the HiTEC HS-311 Standard for it's reasonable maximum torque and availability at the Electronic Services Shop. The Machine Shop manufactured for us an aluminum finger for the servo, and a bracket to hold the servo in place. To measure pressure we used three of SparkFun's Small Force Sensitive Resistors. The sensors were placed below a platform with three small "feet". All of the weight of objects placed on this platform was directed into the sensors, which have very small surface areas. A neoprene "finger pad" was attached to the aluminum finger, allowing objects to be gripped without slipping.

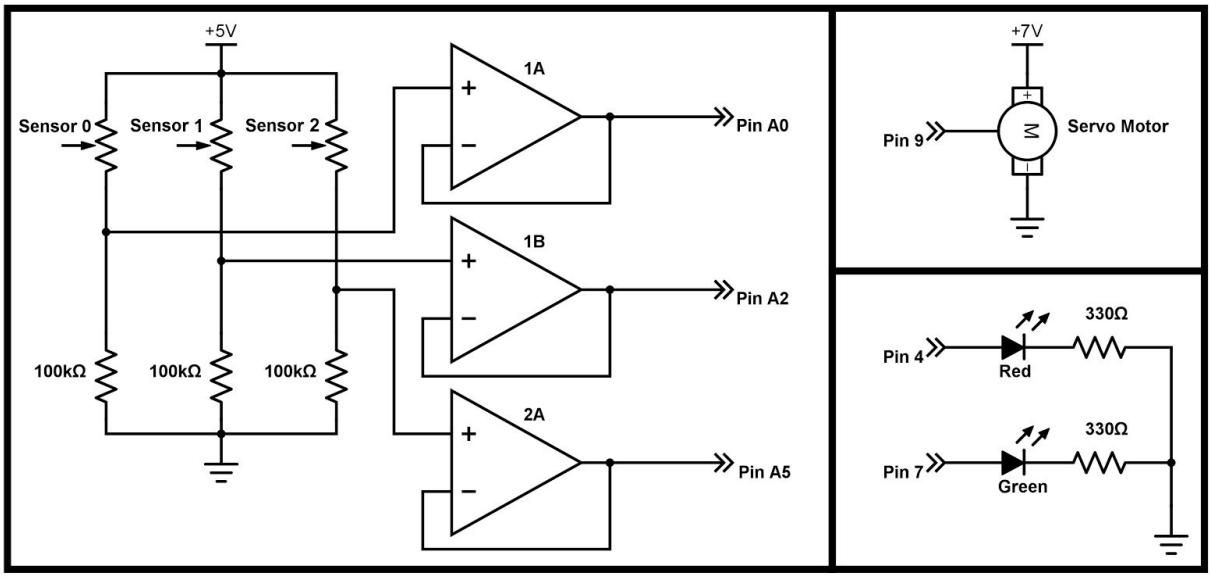


Block Diagram

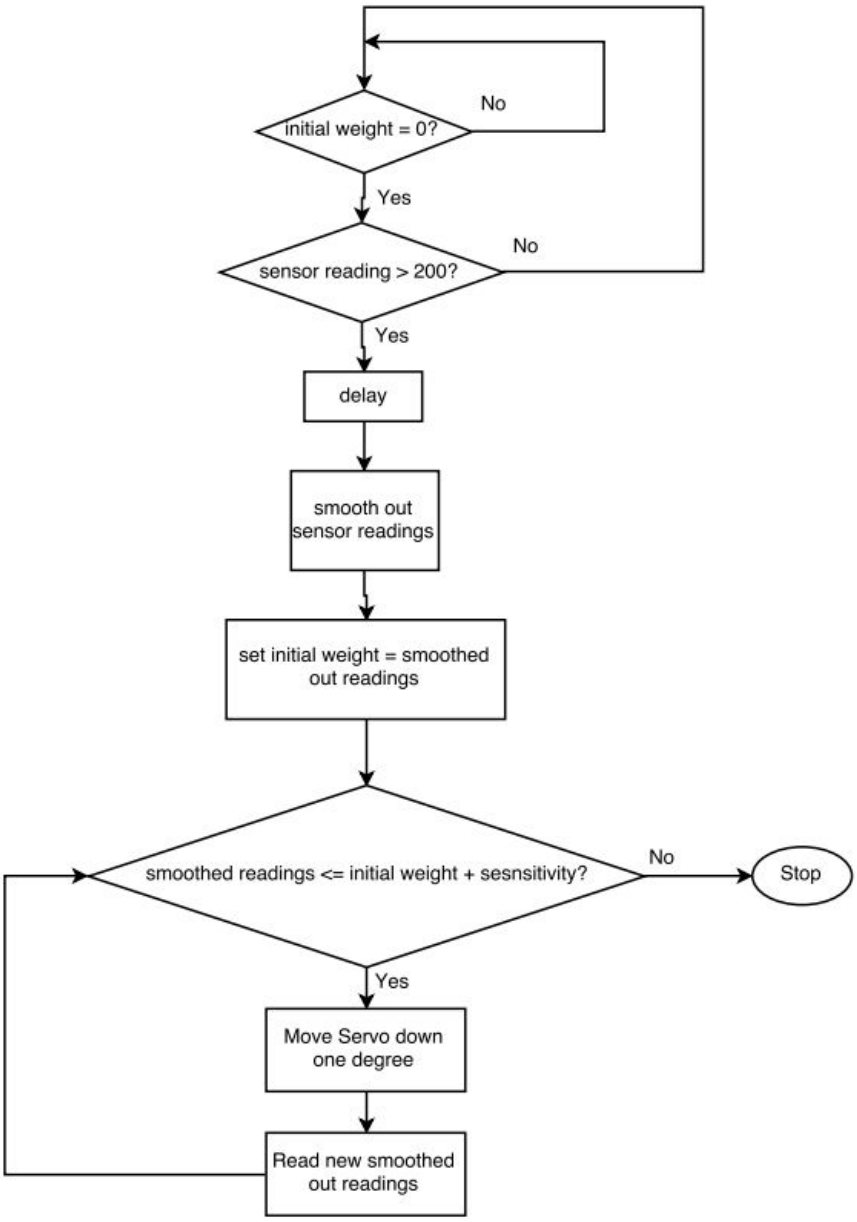
A 7 volt battery pack was used to power the RedBoard, which in turn provides power to the force sensitive resistors and servo motor. The RedBoard reads the voltage from the force sensor circuit, controls the servo motor movement, and turns on and off the LEDs.

Circuit Diagram

We set up the force sensitive resistors in voltage dividers with 100kΩ resistors and the 5V voltage output from the RedBoard. After having a lot of trouble reading consistent values, we set up LM358N Op-amps as buffers to prevent any interference in the voltage divider from the RedBoard when it reads in the pins. The input impedance to the op-amp is very high, and effectively draws no current from the voltage divider, and therefore does not interfere with it. The voltages from the op-amps are read by analog inputs A0, A2, and A5 on the RedBoard. The servo motors is controlled by pin 9 from the RedBoard, and powered by the RedBoard's VIN, 7 volts. Pins 4 and 7 control two LEDs to show the current state of the code.


+5V
Sensor 0
Sensor 1
Sensor 2
100kΩ
100kΩ
100kΩ
1A
1B
2A
Pin A0
Pin A2
Pin A5
+7V
Pin 9
Servo Motor
Pin 4
Red
330Ω
Pin 7
Green
330Ω

Code Flow Chart



Code

```
#include <Servo.h>

Servo servo1;                    //defines servo1 as a servo class
const int forcePin0 = 0, forcePin1 = 2, forcePin2= 5;    //initializing and defining variables
const int servoPin=9, ledPinW = 7, ledPinS= 4;
int delta = 100, baseValue = 0;
```

```
int initialForce, initialWeight = 0;
int forceReading0, forceReading1, forceReading2;
int totalReading, servoPosition = 0;
int timeAveragedReading;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  servo1.attach(servoPin);                    //fixes the servo control pin to digital pin 9
  servo1.write(0);                  //sends out a value for the position for the servo
  pinMode(ledPinW, OUTPUT);       //sets the digital pin for the weight LED as OUT
  pinMode(ledPinS, OUTPUT);       //sets the digital pin for the servo LED as OUT
}

void loop() {
  // put your main code here, to run repeatedly:
  forceReading0 = analogRead(forcePin0);          //reads the value at the analog pin 0
  forceReading1 = analogRead(forcePin1);          //reads the value at the analog pin 2
  forceReading2 = analogRead(forcePin2);          //reads the value at the analog pin 5

 //this sums the separate readings to find the total pressure across all three sensors
  totalReading = forceReading0 + forceReading1 + forceReading2;

  Serial.print("Force sensor reading: ");      //prints the reading for debugging purposes
  Serial.print(totalReading);

  Serial.print("   Servo position reading: ");          //prints the servo position for debugging
  Serial.println(servoPosition);

  digitalWrite(ledPinW, LOW);       //sets weight LED to a low voltage
  digitalWrite(ledPinS, LOW);       //sets servo LED to a low voltage

//this first “if” statement will store the weight being placed on the sensor
//it also allows for the program to automatically start once a weight is detected
 if (initialWeight == 0) {
    if (totalReading > 200) {             //a threshold to avoid false starts
      digitalWrite(ledPinW, HIGH);
      Serial.print("I am in the loop");             //debugging the logic
      delay(2000);                  //delay allows for the weight to settle and stay consistent
```

```
    smoothing (30);              //removes some noise from the sensor

    initialWeight = timeAveragedReading;          //stores the approximate weight
    Serial.println(initialWeight);                //this is for debugging
    stepperMove();               //calls the servo to move and stop based on change in pressure
   }
 }

  delay(250);
}

void stepperMove () {
  smoothing(15);

//this "while" loop sets a threshold for the servo to overcome before it should stop moving
  while (timeAveragedReading <= initialWeight + delta) {
    digitalWrite(ledPinS, HIGH);       //turns the servo LED on
    servoPosition += 1;                //moves servo down if the pressure isn't high enough
    servo1.write(servoPosition);       //debugging the servo position to determine delta
    delay (10);

    smoothing(15);

  Serial.print("Force sensor reading: ");
  Serial.println(timeAveragedReading);

  Serial.print("Servo position reading: ");
  Serial.println(servoPosition);
  }
}

void smoothing (int x) {
      for (int i=0; i<x; i++) {    //grabs X amount of readings and takes a running average

      forceReading0 = analogRead(forcePin0);
      forceReading1 = analogRead(forcePin1);
      forceReading2 = analogRead(forcePin2);
```

```
      totalReading = forceReading0 + forceReading1 + forceReading2;

      if (i==0) {            //initial average is set to the first reading
        timeAveragedReading = totalReading;
      }
      if (i>=1) {            //new average is then based on the old one and new reading
        timeAveragedReading = (totalReading + timeAveragedReading) / 2;
      }
     }
  Serial.println("Smoothed");
}
```

Results

By the time of demonstration, our project could successfully recognize the weight of an object placed on the platform, move the finger down at a regular speed, and recognize the slight increased pressure from the finger contacting the object, and stop its movement. We wish we could have moved further with this project, but we had so many issues with the sensors and servo that this was not possible. In order to better present our project, we added a red and green LED to indicate what the servo was doing at the time.

We showed that the robotic finger worked with two objects, an 18g plastic container and a 66g aluminum ingot. 18g is approximate the activation mass for the force sensors. Lighter objects would not register, even after the implementation of the platform. The platform consistently allowed all three sensors to read numbers within 20% of one another, a big improvement from often reading 0 from one of the sensors. The neoprene finger pad gave good contact and prevented objects from slipping away from the finger.

Our irregular noise values from the sensors with no force on them were always below 40. We set our threshold for activation as 200, as the lowest readings we could point to as from an object rather than noise were approximately 400-430 . The sensors appear to have an activation weight of about 16g, and after overcoming this they can clearly read changes as small as 3g. This is not ideal, but it means it may be possible to hold objects delicately enough and achieve our original plan.

Future Work

Although we managed to achieve the first half of our experiment, there are many things we can improve on this project to achieve a more human-like grip. Firstly, we still need to write in additional programming to accommodate the vertical shifting of the robotic finger. Secondly, the limitations on our sensors didn't allow us to grip squishier, more delicate objects so we can continue to experiment with a different sensor for future

revisions. Thirdly, the simplistic servo arm is easy to work with, but does not capture the smaller pressure distribution of a human fingertip. In this regard, we would require a better mechanical design. Fourthly, if we manage to achieve all of these goals, we can move to loftier goals and build another prototype with additional movement capabilities, additional sensors, and additional programming to have the robotic finger automatically pick up and move objects around.

Conclusion

Our project consists of design, produce, and a long process of debugging.

I. What Worked

The neoprene provided enough friction to hold objects. We succeeded in arranging the three very small, sensitive sensors in a "v" shape and combining the resistance values they output. We were able to tune the servo motor so that it can reach 0 to 180 degrees. And thanks to the staffs in Machine Shop, who helped us building an arm and fixed the servo in place with servo motor brackets. We wrote a correct programming algorithm to reduce noise by doing average over time, with a counter variable. It succeeded to limit the fluctuation within a range which was small enough for us to deal with. In fact, we could make the fluctuation as small as desired by including more variables to do average. Also note that fluctuation is not equivalent to noise, although they are somewhat related. What we did, strictly speaking, only aimed at reducing unintended fluctuation in data processed by our program, not necessarily noise itself or the cause of it. We found that this process successfully met our needs.

II. Went Wrong But Solved

At first we built the force sensor circuit with a simple VDR circuit. Due to very inconsistent readings, we tried the recommended op-amp circuit found in the datasheet, which we believe cleaned up our readings. Another issue was that often the weight of the object was distributed on only one or two sensors, rather than all three. This caused the resistance data input to Arduino to be ridiculously high and unable to use when we at first connected three sensors in series. But after we re-arranged them in parallel, this issue no longer was a big problem. We finally succeeded in making the force distributed on all sensors by using a board, which gave us even more consistent readings for a given object.

Debugging the code took a long time. At the beginning of November, our code made the arm go down regardless of whether there was object on the sensors. This is because forceReading variable was not read and overridden in each loop containing the movement of the arm. However, exactly the same mistake occurred again, for a long time even till the

last day before the demo. Luckily we found it and corrected it in time. We also debugged a problem about sensor reading, which read a few large numbers at the beginning of each run of program, while it should have read zeros constantly. It was because when the servo started to draw a current, which was pretty large, it affected other parts in the circuit that shared the same voltage source (because the source is not "ideal" defined by electric terms). After we made them use separate sources, the bug was fixed.

III. Still need to improve

We need a more powerful servo motor. The current one cannot give enough force to allow the arm to hold a heavier object in position once the board is tilted. Also, the sensors might not be as sensitive as described in datasheet. They won't register objects that are too light. And if we change the circuit to make them more sensitive (by making the set resistance equal the working resistance of the flex sensor in series to it, as we proved mathematically), then too much noise value would show up, and we could not tell between noise and light-weight objects. In fact, we finally found a better method to build the circuit and reduce noise, but the issue of insensitivity still exists as long as we want to try lightweight objects. In addition, a few points in the Future Work section were what we planned to achieve this semester if time allows. Although time turned out to be too limited, at least we have gained some broad picture of how those goals would be achieved, based on what we achieved now as well as what we experienced in the process.

IV. Lessons Learned

We learned the correct way to build the circuit for force sensor and one way to build the physical setup of the finger-like device. We learned how to control servo motor, and specifically in our project, we made it move one degree at a time and checking whether it was time to stop before each move. We wrote and applied an average filter with a counter. We explored a way to physically make the weight plus force to be distributed over all of three sensors. And we also learned the fact that for an not-so-ideal source, when the servo started to draw a current, which was pretty large, it affected other parts in the circuit that shared the same voltage source.