Inmoov Robotic Finger

Scalable Designs

Ehren Chan || Anda Su

Computer Engineering 12 - TEJ4M

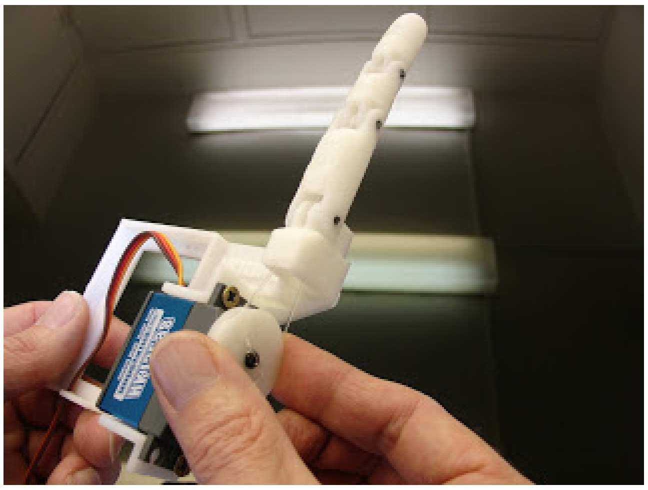
J. Radulovic

17/06/2019

TABLE OF CONTENTS

1) Background Information

We will be making a robotic finger which can be scaled to later become a whole arm (and even a whole body). The final project should look like this:


(Taken from InMoov's website).

This project will require a Raspberry Pi, an Arduino, a servo motor, and 3D printed finger parts (along with the necessary cables/wires to power and the devices).

This robotic finger will be controlled remotely. The host will be connected to the Raspberry Pi, which processes an input signal, and sends another signal to the Arduino to modify the code that the Arduino runs. The Arduino will be solely responsible for controlling any movement of the motors, and the Pi will be responsible for communicating with the host, processing input signals, and outputting data to the Arduino.

2) Proposal

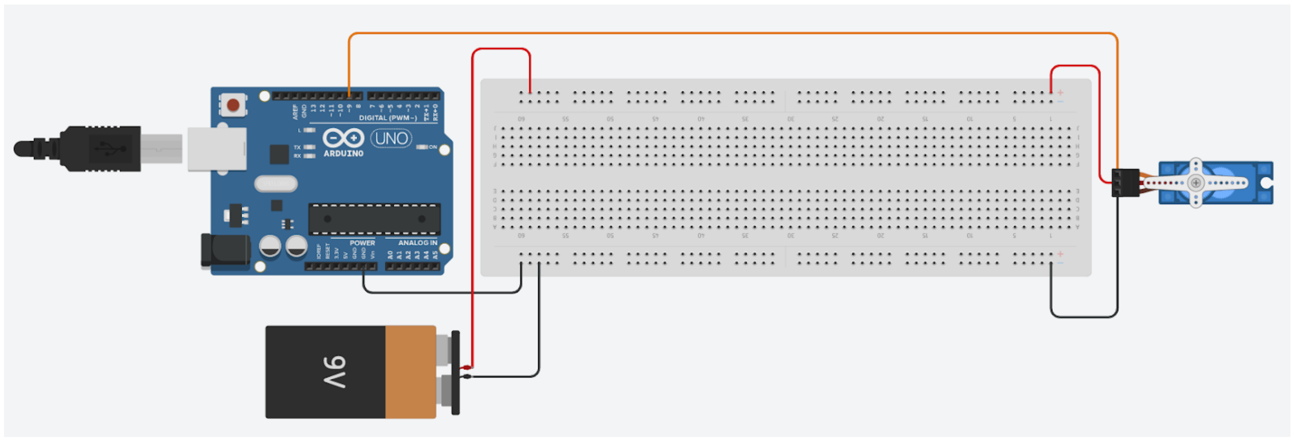
Task	Plan to Accomplish Task	Person Responsible
Setup (Wireless Mesh) Network	Define the router's IP (LAN), add Ubiquiti APs to the network using the Unifi controller, connect any necessary switches to the router (for LAN connections)	Anda
Setup Pi	Download the necessary programs (Python), and set a static IP address	Anda
Test Arduino Code	Using a DC motor, test how to send signals to output to change the speed of the motor (PWM)	Anda
Test Pi Code	Using pygame, assign keypresses to output certain signals (in this case, printing in the console)	Ehren
Research & test code between Pi and Arduino	Learn how the Pi code can be written to send a signal to the Arduino through the USB/ACM interface, referring to documentation made by other people online, and modifying the code to send our desired signals, using a DC motor to test the final code	Anda & Ehren
Research how a servo motor works, and how code with an Arduino can control the movement	Use online resources, and the specification sheet of the servo motor to figure out what duty cycle will turn the motor a certain direction, and search up code from other people (using Servo.h library)	Anda
Construct robotic finger	Follow the video tutorial to put the finger together	Anda
Write code for Arduino and Pi	Modify the previous test codes to apply to the servo motor	Anda (Arduino) & Ehren (Pi)
Modify Pi code so that the finger is controlled via holding the button instead of requiring individual key presses	Have the code set a state when the button is held and resets when released instead of sending a signal on button presses. The signal sent will be determined by the current state.	Ehren

3) Design

Schematics

The 3D Printed parts can be found on InMoov's website:
http://inmoov.fr/inmoov-stl-parts-viewer/?bodyparts=Finger-starter-kit

The Arduino and servo motor will be wired as follows:



The Arduino and Pi will be connected via USB, the Arduino will simply get its power from the USB port of a Pi.

Pseudocode

Arduino

```
signal_recieved = serial.read() // read incoming data
if(signal_recieved == HIGH){
     Turn motor CW incrementally
}
else if(singnal_recieved == LOW){
     Turn motor CCW incrementally
}
```

Pi (Python)

```
if(KeyPressed == KEY_UP){
     signal_send == HIGH
}
else if(KeyPressed == KEY_DOWN){
     signal_send == LOW
}
serial.write(signal_sent) // send data via USB interface
```

4) Implementation of Design

Costs + Materials

Arduino: $22.00
Raspberry Pi: $46.10
Servo Motor: $44.99
9V Battery: $2.99
Fishing line: ~$10.00
Total Cost: $126.08 (Not including 3D printer/printed parts or wires)

Steps

1. Setup the network:
    a. Define router's LAN IP
    b. Enable DHCP
    c. Enable WLAN and create SSID and password
2. Setup the Pi:
    a. Enable SSH using sudo raspi-config or the GUI
    b. Set up static ip, using the GUI
3. Wire the hardware according to the diagram above
4. Connect Arduino to computer and write Arduino code (following the attached code) using the Arduino IDE
5. Upload the code onto the Arduino
6. Write Python code for Pi, following the attached code
7. Connect the Arduino to the Pi via USB
8. Construct finger with motor following this video: https://www.youtube.com/watch?v=0t2uhAyf2-c

Code

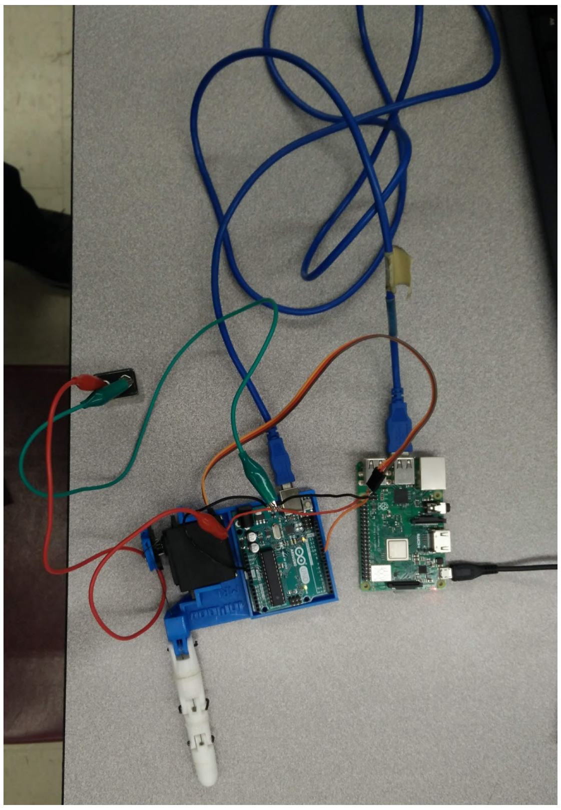
Arduino

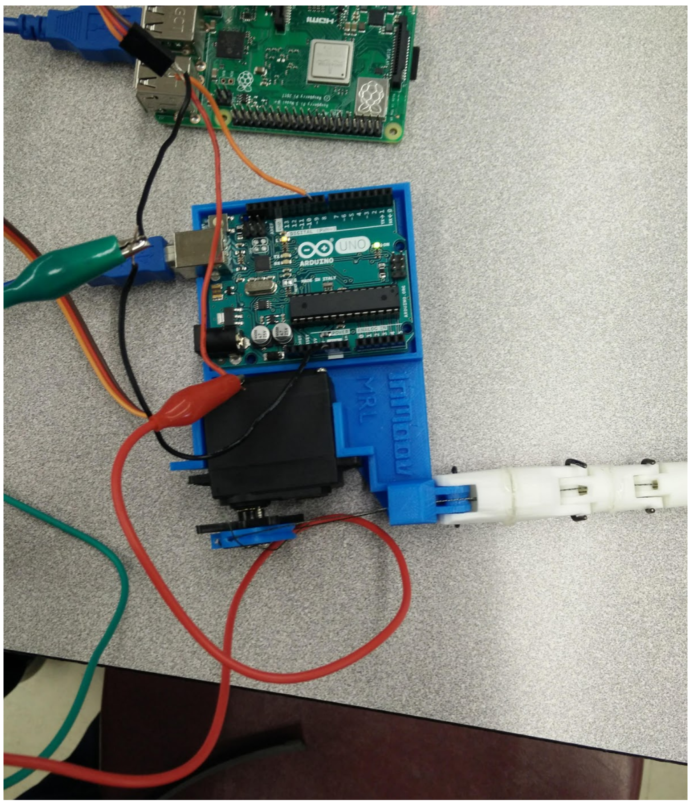
```
#include <Servo.h>

Servo servo;  // create servo object to control a servo

int pos = 90;  // variable to store the servo position
int change = 0;
int incoming_state = 0;

void setup() {
  Serial.begin(9600);
  servo.attach(9);
  servo.write(90);
}

void loop() {
  if(Serial.available() > 0){
     incoming_state = Serial.read() - '0'; // Reads input
     if(incoming_state == 1 && pos < 180){ // Checks for a signal
sent
          change+=5; // Increments if the correct signal was sent
(CW Motion)
     }
     else if(incoming_state == 0 && pos > 0){ // Checks for a
signal sent
          change-=5; // Increments if the correct signal was sent
(CCW Motion)
     }
     if(change != 0){ // If there was a change in position
          pos = pos + change;
          servo.write(pos); // Turn the motor
     }
  }
  change = 0;
}
```

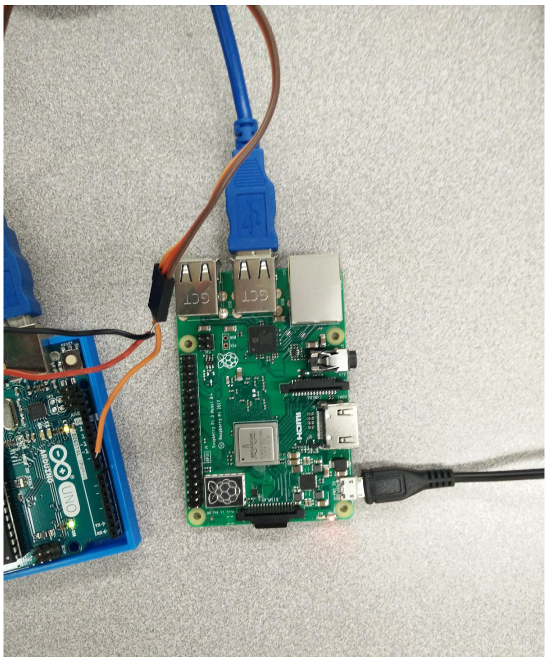
Pi (Python)

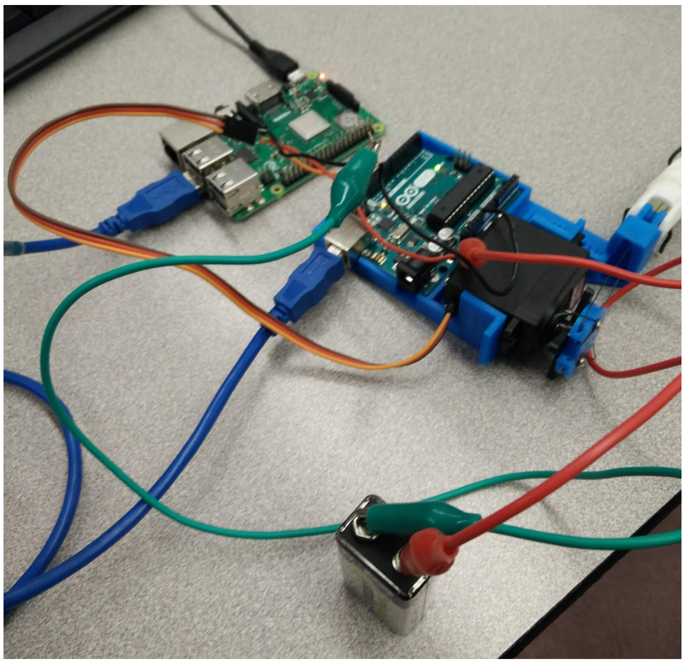
```
import serial //Importing packages required for the code
import pygame
import time
ser=serial.Serial("/dev/ttyACM0", 9600) // Initializes the usb
interface to send the data to and from
pygame.init() // Starts the pygame screen for controls
screen=pygame.display.set_mode((200,200))

while True:
     for event in pygame.event.get():
     if event.type == pygame.KEYDOWN: // Detects keypress down
          if event.key==pygame.K_w: // Detects keypress "w"
               ser.write(b'1') // Writes an output "1"
          if event.key==pygame.K_s: // Detects keypress "s"
               ser.write(b'0') // Writes an output "0"
          if event.key==pygame.K_9:
               break
```

Images









5) Discussion of Results

At the end, the finger did work successfully. However, there were some inconsistencies and inconveniences due to the implementation of the code and hardware limitations.
At times, the connection was dropped, and the motor didn’t move upon sending the signals. This was due to loose connections, and could have been fixed by soldering the wires together to ensure a connection. Also, the way the code was implemented prevented a press-and-hold action to continue to turn the motor. This made the user press a certain button over and over again just to turn the motor in one direction all the way. This could have been resolved by adding extra code within the Python program to detect a press and hold.

6) Resources

InMoov's website, documenting how to construct the finger:
http://inmoov.fr/finger-starter/
Electronics Hobbist's website, documentation on how to communicate to and from an Arduino and Pi:
https://electronicshobbyists.com/control-arduino-using-raspberry-pi-arduino-and-raspberry-pi-serial-communication/
Communication from the Pi to the Arduino, encoding a message with "b" before the actual value:
https://classes.engineering.wustl.edu/ese205/core/index.php?title=Serial_Communication_between_Raspberry_Pi_%26_Arduino
Guide to Pygame:
https://www.pygame.org/docs/ref/key.html

7) Designer’s Notes

In chronological order of events:

What was done? What issues were present?	Next steps
AP (wireless network) setup, APs weren’t being adopted originally	Try setting a static IP for the AP (The AP was eventually adopted automatically)
Testing Python and Arduino code on a motor (sending signals and changing duty cycle for the arduino, detecting signals and printing them out for the pi)	Get Arduino to detect signal from Pi
Found code to send signals over the USB interface (using Serial.read()), but the arduino wasn’t detecting the signal from the Pi	Research more sample code and modify to fit our needs
Found out the problem with sending signals from the Pi: using serial.write(b’1’), the ‘b’ was the key, as it “encoded” the value (in this case, 1) and sent it over to the Arduino to be read properly, all this testing was done with a DC motor	Research how to operate a servo motor
Testing how to control motor with Arduino, tried some code from online sources but nothing seemed to work	Research duty cycle of the servo motor and how to control rotation of motor with Arduino
Discovered that servo motor was broken due to overloading (used power source that was too big)	Replaced broken motor with fully functional one, learned how to properly wire a servo motor (power doesn’t go directly to servo in order to turn it! It needs PWM signals from the arduino as well)
Finally tested code with a working servo motor, using Arduino and Pi, It worked perfectly fine	Begin construction of the finger
Built finger according to InMoov’s video demonstration	Run the code with the complete finger
Finished testing with the finger constructed! Working fine, but sometimes connections become loose, also wanted to revise the code to add a press-and-hold action to continue to turn the motor in a specific direction, but ran out of time	Submit!

Note: The finger can be scaled up to have multiple fingers, and eventually, an entire arm. The code and hardware can be used over and over again.