Fall | 14

Weekly Status Report

Wesley Myers

October 28th - November 3rd, 2014

wesley.y.myers@gmail.com

Table of Contents

Executive Summary

Appear to be on track with requirements. This week's objective was to have a website up and running for the most part.

Interface Board

- Narrowed Bug from Last Week
- Board Revision for Power Issues

Website Development

- Basic Website Stood Up

Arduino Serial Code Development

- Reads Analog Pin and Sends to Pi
- No luck on serial communication to LRF

Interface Board

A strange thing was happening during last week's testing was that every time the Raspberry Pi communicated with the Arduino to move a servo, the Arduino would reset. What I believe was happening was that there was a sudden current spike due to moving the servo, causing a drop in voltage, and thus caused the Arduino to turn off briefly and reset. In order to counteract this we need to supply the servos via a separate source. The following is a diagram of this test configuration that I had set up. I believe the laptop could not handle the initial current.

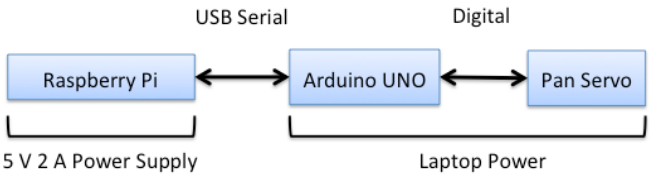

Figure 1 - Test Set Up

Two weeks ago, work was done to document an interface board for the Arduino, Raspberry Pi, and Laser Range Finder. This interface board makes it such that a user could swap out any component easily and not have to worry about any hard connections.

The LRF will draw 150 mA maximum. The Arduino UNO can draw a maximum of 500 mA before the on-board fuse fires. The UNO can also output a maximum of 200 mA. At normal operation, each servo can draw around 250 mA.

The 5V rail of the Raspberry Pi rail is passed straight from the USB power supply[1]. Therefore the maximum current draw is limited by the current from the power supply. Therefore, the Arduino and LRF can easily be powered by the Raspberry Pi.

In order to support all devices, there is a chance that we could use a 3 Amp power supply to plug in directly to the Raspberry Pi. However to be on the safe side, it was determined that brown outs were not desired in the case of a sudden jump in current draw. In order to support this, some more components were ordered.

- DC Barrel Jack Adapter[2]
- 5 V DC 2 A Wall Adapter[3]
- Two Sided Proto Board[4]

This is also unfortunately will cause another revision to the board layout. The new layout is as follows. The goal is to supply the servos with their own power source. This will eliminate any chance of a brown out for the system.

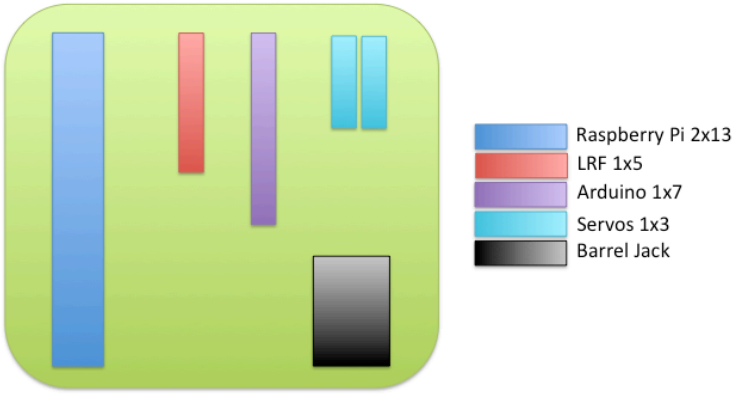

Figure 2 - Interface Board Layout

[1] http://raspberrypi.stackexchange.com/questions/9298/what-is-the-maximum-
[2] https://www.sparkfun.com/products/10811
[3] https://www.sparkfun.com/products/12889
[4] https://www.sparkfun.com/products/8811

Website Development

So a big issue I've been facing is trying to figure out how to stream the video feed to a webpage. Shouldn't this be simple? It is fairly straightforward[5] to stream the video to an IP address, but to embed it in a webpage is a bit more complicated.

I ended up going with a package I found that is apparently tried and true. It is simply called Raspberry Pi Webcam Interface[6]. All of the code can be sourced off of the referenced github account[7].

First things first, you'll want to make sure your Raspberry Pi is up-to-date.

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo rpi-update
sudo reboot
```

You'll now want to copy the files over to your Pi. So download the software off of the github account and *scp* them over. After you unzip them on the pi, then it is straightforward to install.

```
cd /home/pi/
unzip RPi_Cam_Web_Interface.zip
cd RPi_Cam_Web_Interface
./RPi_Cam_Web_Interface_Installer.sh install
./RPi_Cam_Web_Interface.sh start
```

Now you should be up and running! Go to the website and you should be able to see an interface with many options.

```
192.168.1.110/index.php
```

All of your files are located in */var/www*. The files in that directory were copied over when the install was done. If your IP address is different, then you may want to do an *ifconfig* and double check.

---

[5] http://www.raspberrypi.org/turn-your-pi-into-a-low-cost-hd-surveillance-cam/
[6] http://elinux.org/RPi-Cam-Web-Interface
[7] https://github.com/silvanmelchior/RPi_Cam_Web_Interface

After much headache learning HTML and CSS, I came up with the following as a basic interface. Right now it doesn't engage the servos to actuate. The code can be found in Appendix B.

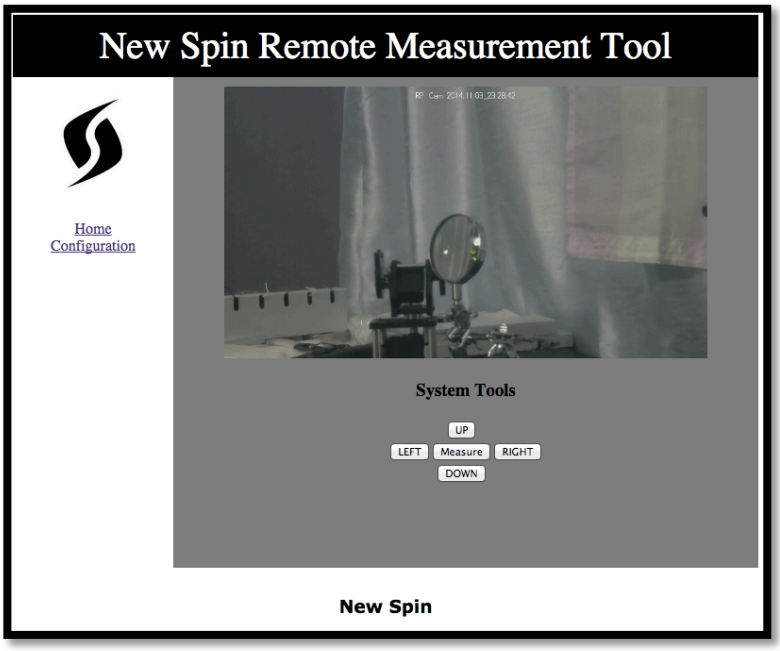

Figure 3 - Website Screen Capture

Some further things I need to work on are how to still take images for OpenCV to crunch on, save the servo configuration and laser reading into a database, etc.

Arduino Serial Code Development

Integrated the various programs into one source file to handle all serial communications for the Arduino. This current version just samples the Analog pin on the Arduino, low pass filters the samples, and then sends the result back to the Raspberry Pi. The code can be found in Appendix A.

Currently I've had no success getting the communication between the Arduino and LRF to work. This might be shelved for the end of the semester at this point to try to debug this problem.

Appendix A

```
#include <Servo.h>

//#define DEBUG

//tilt
#define TILT_LEVEL 90
#define TILT_MAX_DOWN 75

//pan
#define PAN_MIDDLE 90
#define PAN_MAX_LEFT 50
#define PAN_MAX_RIGHT 130

// laser defines
#define SF01_ANALOG 0
#define SF01_0_0V_DISTANCE        0.0
#define SF01_3_3V_DISTANCE        33.00
int analog;
float analog_voltage;
float analog_distance_meters;
float slope = (SF01_3_3V_DISTANCE - SF01_0_0V_DISTANCE)/3.3;

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

int pan_pos;    // variable to store the servo position
int tilt_pos;
int new_pos;

char command;
int i;

void setup()
{
  panServo.attach(5);  // attaches the servo on pin 9 to the servo object
  panServo.write(PAN_MIDDLE);

  tiltServo.attach(6);
  tiltServo.write(TILT_LEVEL);

  pan_pos = panServo.read();
  tilt_pos = tiltServo.read();

  Serial.begin(9600);
}


void loop()
{
  if (Serial.available())
  {
    //get command (i.e. read laser, move pan-tilt)
    command = Serial.read();

    //if there is a number trailing, then get that too
    new_pos = Serial.parseInt();

    if(command == 'p')
    {
      // we want to move the pan servo

      if(new_pos >= PAN_MAX_LEFT && new_pos <= PAN_MAX_RIGHT)
      {
        #ifdef DEBUG
        Serial.print("Moving Pan Servo: ");
        Serial.print(new_pos);
        Serial.print(" -> ");
        Serial.println(pan_pos);
        #endif
```

```
    if (new_pos > pan_pos)
    {
      while (pan_pos < new_pos)
      {
        pan_pos++;

        panServo.write(pan_pos);
        delay(50);

        #ifdef DEBUG
        Serial.println("right!");
        #endif
      }
    }
    else if(new_pos < pan_pos)
    {
      while (new_pos < pan_pos)
      {
        pan_pos--;

        panServo.write(pan_pos);
        delay(50);

        #ifdef DEBUG
        Serial.println("left!");
        #endif
      }
    }

    new_pos = panServo.read();
  }
  else
  {
    #ifdef DEBUG
    Serial.println("===== Bad Pan Servo input =====");
    Serial.print(" - Servo: ");
    Serial.println(command);
    Serial.print(" - Position: ");
    Serial.println(new_pos);
    #endif
  }
}
else if(command == 't')
{
  // We want to move the tilt servo

  if(new_pos >= TILT_MAX_DOWN && new_pos <= TILT_LEVEL)
    {
      if (new_pos > tilt_pos)
      {
        while (tilt_pos < new_pos)
        {
          tilt_pos++;

          tiltServo.write(tilt_pos);
          delay(50);

          #ifdef DEBUG
          Serial.println("up!");
          #endif
        }
      }
      else if(new_pos < tilt_pos)
      {
        while (new_pos < tilt_pos)
        {
          tilt_pos--;

          tiltServo.write(tilt_pos);
          delay(50);
```

```
            #ifdef DEBUG
            Serial.println("down!");
            #endif
          }
        }

        new_pos = tiltServo.read();
      }
      else
      {
        #ifdef DEBUG
        Serial.println("===== Bad Tilt Servo input =====");
        Serial.print(" - Servo: ");
        Serial.println(command);
        Serial.print(" - Position: ");
        Serial.println(new_pos);
        #endif
      }
    }
    else if(command == 'r')
    {
      // We want to read the LRF

      analog = 0;

      for(i = 0; i < 5; i++)
      {
        // Read the ADC value of the analog input pin
        analog += analogRead(SF01_ANALOG);
        delay(150); //refresh time for laser is 125
      }

      //LPF the value
      analog = analog/5;

      // Convert this into a voltage
      analog_voltage = analog * 0.0049;

      // Convert the voltage into a distance using the SF01 settings
      analog_distance_meters = analog_voltage*slope + SF01_0_0V_DISTANCE;

      //test response
      Serial.println(analog_distance_meters);

      #ifdef DEBUG
      Serial.println("===== Read LRF =====");
      #endif
    }
    else
    {
      #ifdef DEBUG
      Serial.println("===== Bad Input =====");
      Serial.print("Command: ");
      Serial.println(command);
      #endif
    }
  }
}
```

Appendix B

```
<!DOCTYPE html>
<html>
  <head>
    <title>New Spin Remote Measurement System</title>
    <script src="script_min.js"></script>
    <link rel="stylesheet" href="styles.css">

 </head>
  <body onload="setTimeout('init();', 100);">
      <div id="header">
      New Spin Remote Measurement Tool
      </div>

      <div id="nav">

      <center><a href="http://www.newspin.com"><img src="ns_logo.jpg"
width="100" height="120"></a></center><br>
      <a href="clean.php">Home</a><br>
      <a href="config.php">Configuration</a><br>

      </div>

      <div id="section" style="display:table;">

      <img id="mjpeg_dest" />

      <h3>System Tools</h3>
      <center>
      <table style="width:10%">
        <tr>
          <td></td>
          <td><center><button type='button' id='tilt_up'
onClick='window.location="";'>UP</button></td>
          <td></td>
        </tr>
        <tr>
          <td><center><button type='button' id='pan_left'
onClick='window.location="";'>LEFT</button></center></td>
          <td><center><button type='button' id='measure_lrf'
onClick='window.location="";'>Measure</button></center></td>
          <td><center><button type='button' id='pan_right'
onClick='window.location="";'>RIGHT</button></center></td>
        </tr>
        <tr>
          <td></td>
          <td><center><button type='button' id='tilt_down'
onClick='window.location="";'>DOWN</button></center></td>
          <td></td>
        </tr>
      </table>
      </center>
      </div>

      <div id="footer">
      <p><h3>New Spin</h3></p>
      </div>

  </body>
</html>
```