

Operation Manual

Team No. 19

Construction Marking Robot



Members:

Kelsey Howard (knh12d)

Justin Gibbs (jrg13f)

Brandon Roberts (bdr12)

Derrick Portis (dp11d)

Christian Baez (cob11b)

Advisor: Dr. Nikhil Gupta

Sponsor: PSBI – Mark Winger

Instructor: Dr. Nikhil Gupta

April 1, 2016

Contents

Abstract.....	ii
Functional Analysis	1
Project Specification	3
Project Assembly	6
Operation Instruction	7
Software Instructions	7
Hardware Instructions	10
Troubleshooting	11
Regular Maintenance	11
Spare Parts	12
Acknowledgement	12

Abstract

The operation manual includes the complete functional description of the project, the specifications of individual parts used in the project, the 3-D mesh of how the parts are put together, and the step by step instruction set in order to put the project together and replicate its functionality. In addition, this document provides trouble shooting instructions, information for regular maintenance for specific parts as needed and a list of spare parts that should be kept on hand in case of failure or degradation.

Functional Analysis

The function of this project is to design an accurate robotic marking system that will read in a floor plan and print it out on concrete at full scale. The project works by first taking the floor plan as a CAD file of type dxf and running this file through the Pointor program created by Ransen Software, this will be run on the Raspberry Pi 2. This software will analyze the floorplan file, and determine all of the individual lines that make up the floor plan. From there, the program will output a text file containing the endpoints of all the lines in the floorplan. This text file of endpoints will then be read into a density propagation program that will take the endpoints and add intermediate coordinate points along the line that are half inch apart. This program will also include a bit for the current marking status, as well as calculate the angle (theta) necessary for the robot to turn in order to move to the next point. The Pioneer 2-DX, a differentially steered robot the team is using was donated for use by the CISCOR lab. This robot will read in the updated text file and pass the velocity, linear velocity (which are both constant) and the theta as parameters for the Pioneer's movement function; this will be done by the Raspberry Pi 2 calling Pioneer's function stored on its own operating system and will be physically connected to it via an Ethernet cable. Attached to the Pioneer is the gantry system designed by the team. The gantry consists of two linear actuators run by two NEMA 23 stepper motors that are controlled by SmartLynx bipolar motor drivers. The code controlling the stepper motor drivers is written on the Arduino UNO microcontroller. The marker holder is connected to the gantry system and run by a NEMA 17 stepper motor; it will be able to switch between a few different markers in order to vary the marking color. The marking holder will also have a standby position which will be used when the robot is not marking. A tracking prism will be mounted on top of the linear guide rail and will be tracked by Trimble's robot total station. The total station will be controlled by Trimble's 2.4GHz external radio and will transmit the prism's location back to the external radio which is connected to the Raspberry Pi 2 on the robot. If the prism's location does not exactly match the text file of coordinates as the robot moves, the difference in distance will be sent from the Raspberry Pi 2 to the Arduino UNO board's linear guide rail movement function. This function will convert the distance to stepper motor steps so that the marker will shift in order to compensate for any accuracy errors. As the robot moves from point to point, the position of the prism will constantly be sent to the Raspberry Pi 2, so that the marker placement can constantly

be adjusting to ensure at least half inch accuracy of the printed floorplan. Instead of having to manually set the gantry's platforms to the center, the raspberry pi will send a soft-stop to the stepper motors to move the platforms on the X and Y axis to the center to prepare for marking. The marking robot will have a LiDar LMS200 (Laser Measurement System) used for object detection. The LiDar will have a function that will be called by the Raspberry Pi to check to see if any obstacles are range and close enough to knock the robot off its course. The LiDar will be mounted on an angled slope to ensure that the LiDar can see shorter objects as the LiDar only has a horizontal movement and not vertical. The mount give the LiDar a 5 foot range before it reaches the ground.

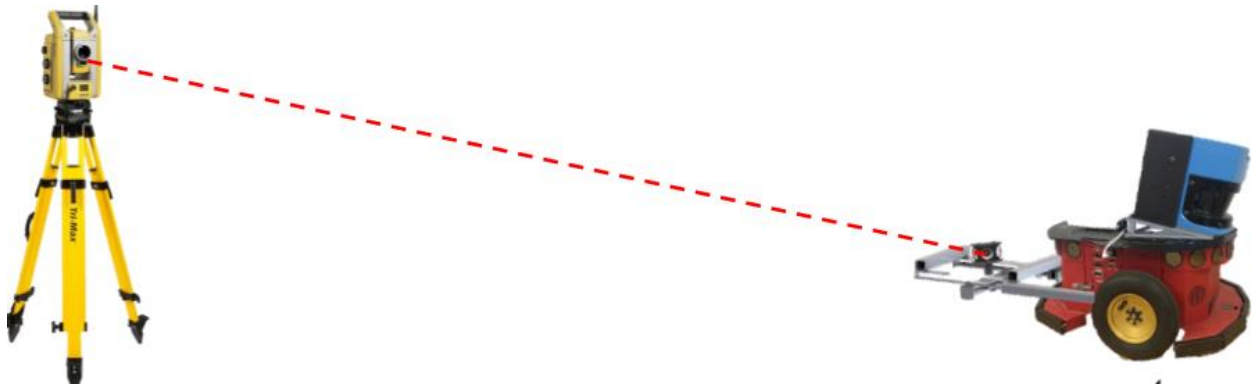


Figure 1: RTS tracking Pioneer

Project Specification

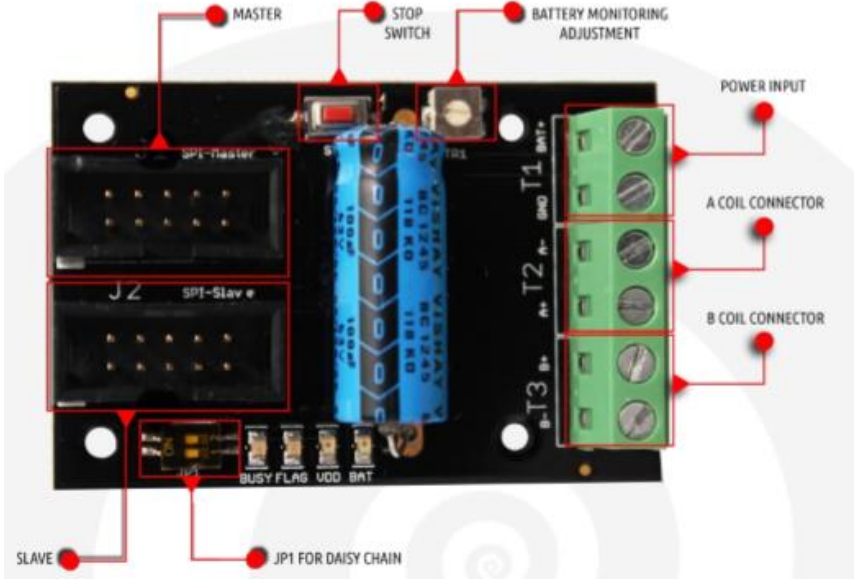


Figure 2: SmartLynx motor driver – Top view

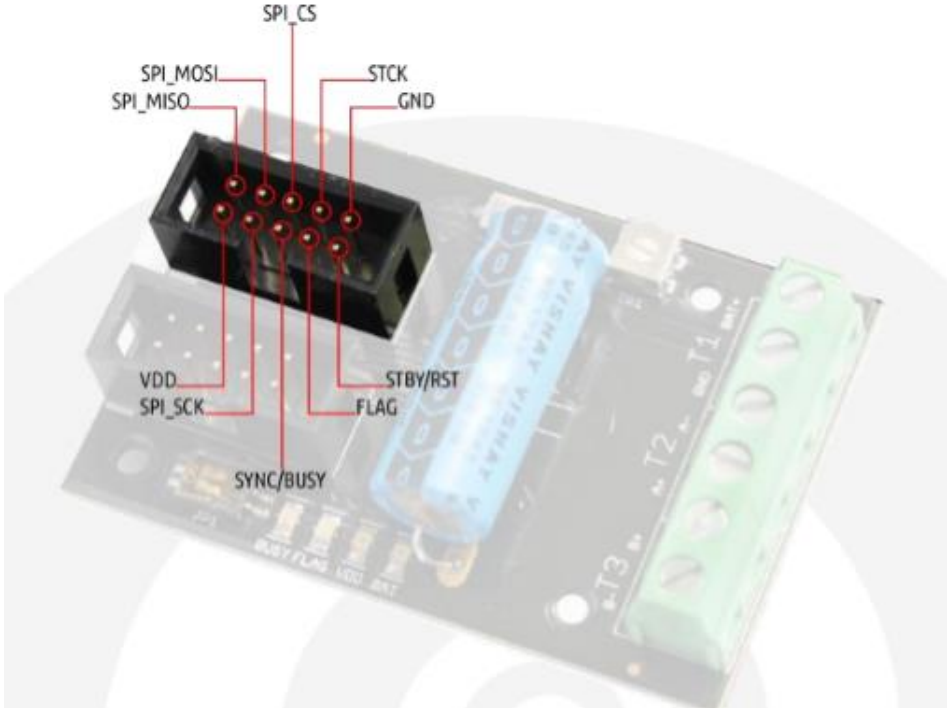


Figure 3: SmartLynx J1 header pin layout

**NEMA size 23 1.8°
2-phase stepper motor**



Mechanical Specifications

Dimensions in inches (mm)

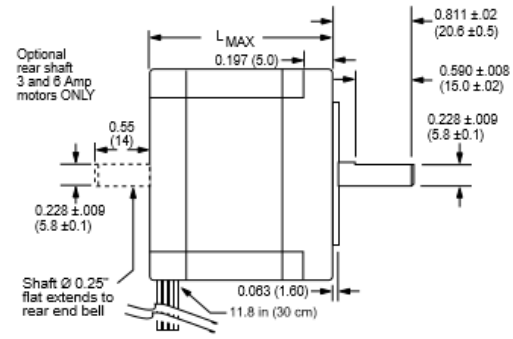


Figure 4: NEMA 23 Mechanical Specifications

Specifications

2.4 Amp motors		Single length	Double length	Triple length
Part number		M-2218-2.4 S (1)	M-2222-2.4 S (1)	M-2231-2.4 S (1)
Holding torque	oz-in	90	144	239
	N-cm	64	102	169
Detent torque	oz-in	3.9	5.6	9.7
	N-cm	2.7	3.9	6.9
Rotor inertia	oz-in-sec ²	0.00255	0.00368	0.0065
	kg-cm ²	0.18	0.26	0.468
Weight	oz	16.9	21.2	35.3
	grams	480	600	1000
Phase current	amps	2.4	2.4	2.4
Phase resistance	ohms	0.95	1.2	1.5
Phase inductance	mH	2.4	4.0	5.4

(1) Only available with single shaft.

Figure 5: NEMA 23 Technical Specifications

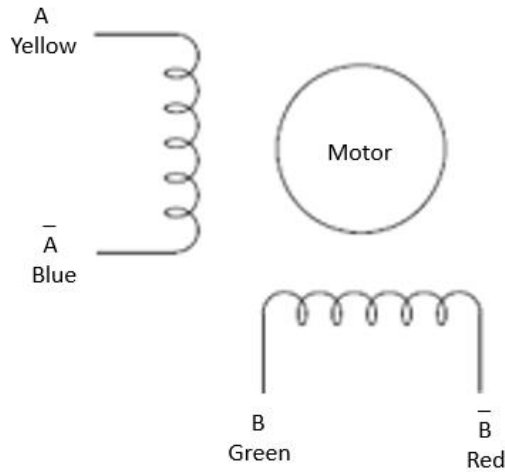


Figure 6: NEMA 23 Wiring Diagram

**NEMA size 17 1.8°
2-phase stepper motor**



Mechanical Specifications
Dimensions in inches (mm)

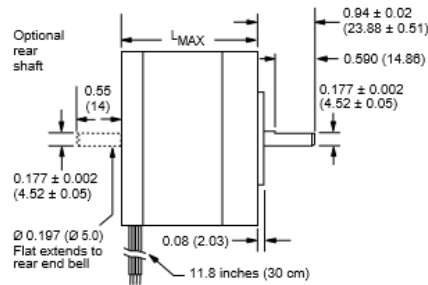


Figure 7: NEMA 17 Mechanical Specifications

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Figure 8: Arduino Mega2560 Technical Specifications

Technical Specifications:

- Broadcom BCM2836 Arm7 Quad Core Processor powered Single Board Computer running at 900MHz
- 1GB RAM
- 40pin extended GPIO
- 4 x USB 2 ports
- 4 pole Stereo output and Composite video port
- Full size HDMI
- CSI camera port for connecting the Raspberry Pi camera
- DSI display port for connecting the Raspberry Pi touch screen display
- Micro SD port for loading your operating system and storing data
- Micro USB power source

Figure 9: Raspberry Pi 2 Model B Technical Specifications

Project Assembly

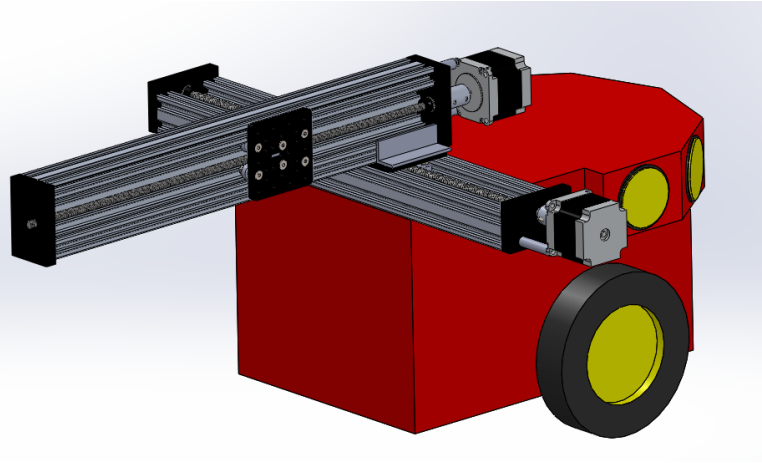
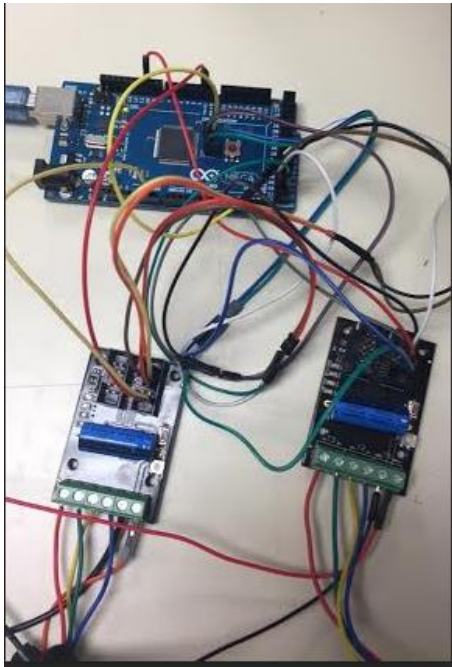


Figure 10: Circuit for SmartLynx Motor drivers connected to Arduino Mega2560 with 3D gantry system

Table 1: SmartLynx Motor drivers to Arduino Mega2560 Pin Connections

SmartLynx pin	MOSI	MISO	SCK	SS	Reset
Arduino Mega pin	51	50	52	53	1
Arduino Mega pin (2nd driver)	ICSP-4	ICSP-1	ICSP-3	10	ICSP-5

Operation Instruction

Software Instructions

To convert a dxf file into a txt file of coordinates, you first run the Ransen Software program Pointor. You select file > Open DXF file... and then select your CAD file. Once you open the desired file, the program will analyze the line types and their amounts based on the file.

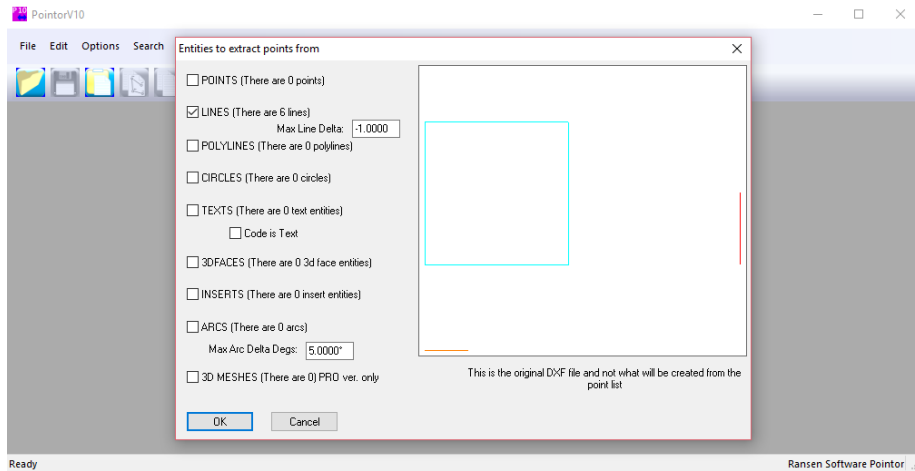


Figure 10: Ransen Software Pointor's DXF analysis

After clicking OK, you will see a screen of coordinate points where the line endpoints were. Next click on the View Pointlist icon to see the list of coordinate points relative to the dxf file.

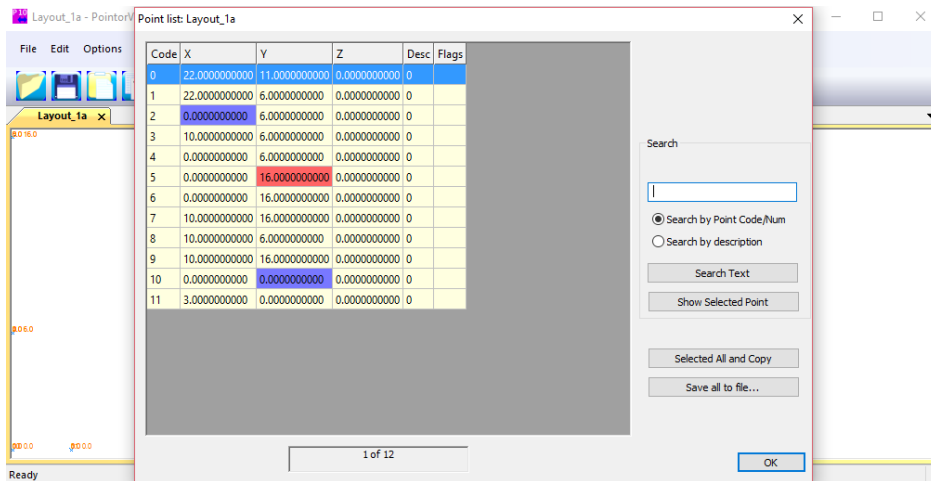


Figure 11: Ransen Software Pointor's Pointlist

Click Save all to file... for the format select Code X Y, select comma <,> for Decimal Separator, and for the Delimiter select space. Type in 1 for "Start writing from row", then select OK and choose a name for your file.

Next compile reader.cpp with the command “g++ reader.cpp” this will create an executable file a.out. We then pass the txt file of coordinates through the executable using the command “a.out < Layout_1a.TXT > output.txt” this will create a new txt file output.txt while will be sent to the robot in order to properly call its movement function. (Note: Layout_1a.TXT is what I called the coordinate file created by the Pointor software, you will enter in the name of your file instead).

In order to control the gantry system download the Arduino software and open the NEMA23 Arduino file created by the team. This code will run a function that takes in an integer to represent the distance and will turn the stepper motor appropriately in order to shift the marker holder as needed according to prism location.

To use the Trimble external radio, connect it to a USB to Serial converter and plug it in a USB port. Run the C# code on a .NET framework and change the Port on your device to match the com port in the code if needed. So far the code will turn on and off the radio. Sending a break for 100ms will turn the radio on, sending a break for 1500ms will turn the radio off. Refer to Trimble’s documentation on the 2.4 GHz external radio for more information.

Connecting Raspberry Pi to Arduino via I2C

While using Raspbian, Go to Terminal and type following commands to download the proper libraries used to connect the two:

- sudo apt-get install python-smbus
- sudo apt-get install i2c-tools

Now you must install I2C support and kernel. Type:

- Sudo raspi-config

A screen should pop up. Go to 8 Advance Options, then go to A7 I2C. Click yes to enable the ARM I2C interface, and yes to the I2C kernel module to be loaded by default. Reboot Raspberry Pi. After reboot, go back to terminal and type:

- sudo nano /etc/modules

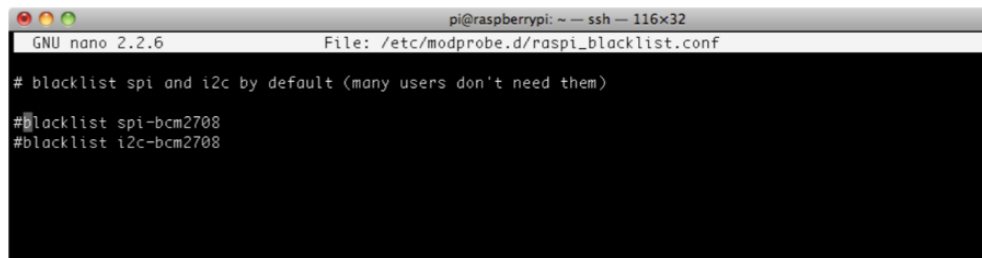
If file exists, go to the very end of the file and comment out these lines by putting # in front:

- i2c-bcm2708
- i2c-dev

Exit the file (CTRL-X and Y) then type:

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

Add the lines:



```
pi@raspberrypi: ~ -- ssh -- 116x32
GNU nano 2.2.6 File: /etc/modprobe.d/raspi_blacklist.conf
# blacklist spi and i2c by default (many users don't need them)
#blacklist spi-bcm2708
#blacklist i2c-bcm2708
```

Figure 12: Raspberry Pi 2 Blacklist code

Afterwards the configuration file needs to be updated. Add:

- sudo nano /boot/config.txt

Add the following lines to the bottom of the text file:

- dtparam=i2c1=on
- dtparam=i2c_arm=on

note that the '1' in 'i2c1' is a one not an L.

You are finished if completed correctly, not you have to reboot:

- sudo reboot

After reboot go to terminal and type:

- i2cdetect -y 1

This pulls up the address block of the Raspberry Pi. This is how you check to see if the Arduino is sending its address to the Raspberry.

Hardware Instructions

In addition to the software component of the robotic total station, it will need to be physically set up and leveled.

In order to control the NEMA23 stepper motor you will need six male to female jumper wires, and two male to male jumper wires for each board. The connection is as follows:

Table 1: SmartLynx Motor drivers to Arduino Mega2560 Pin Connections

SmartLynx pin	MOSI	MISO	SCK	SS	Reset
Arduino Mega pin	51	50	52	53	1
Arduino Mega pin (2nd driver)	ICSP-4	ICSP-1	ICSP-3	10	ICSP-5

The stepper motor wires will be connected according to the NEMA23 wiring diagram. Be sure to connect the SmartLynx ground pin to another ground pin on the Arduino board. On the Smartlynx you will connect the power supply's positive to Battery, and the negative to the ground wire slot next to Battery.

The external radio is connected to a USB port via a Serial to USB converter. Ultimately this should be connected to the USB port on the Raspberry Pi 2. The Raspberry Pi 2 will be plugged into the Pioneer's operating system via an Ethernet cable, and will be connected to the Arduino board via I2C, which allows the Raspberry Pi to send values that will be used in functions on the Arduino's code.

The Raspberry has built in pull-up resistors on the GPIO pins, but they are only available on pins 3 and 5 or the GPIO0 and GPIO1, they are called the SDA and the SCL respectively. The Arduino Uno's I2C pins are A4(SDA) and A5(SCL); Mega's are 20(SDA) and 21(SCL)

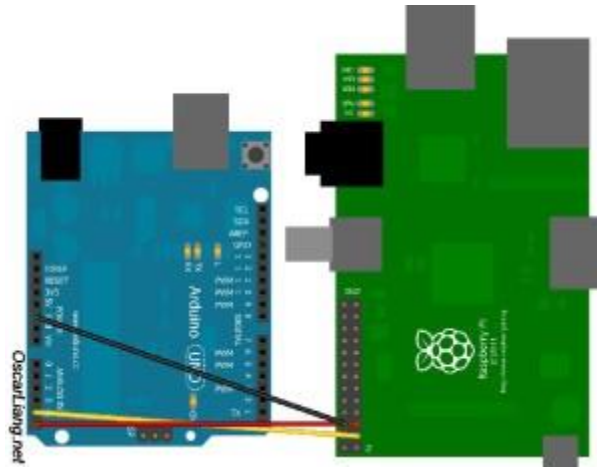


Figure 13: Raspberry Pi 2 to Arduino Connection

Troubleshooting

Some of the current issues with the project include being able to fully utilize Trimble's external radio in order to operate the robot total station so that it can track the prism. The documentation on the radio includes some pseudocode as well as the functions the radio contains, but actually accessing the functions with code is not given and is proprietary to Trimble. An application will have to be developed in order to use the radio, our group can currently turn the radio off and on but this will have to be added to. In addition to this, the radio's hardware seemed very inconsistent. For example, the radio would occasionally not respond to the same code being run under the same parameters.

Another issue we had is to utilize the Pointor software on the Raspberry Pi 2 because it is an exe file and the Pi operates on a Raspbian operating system. Our group downloaded Windows IoT operating system on the Pi in order to run the Pointor software that way, however we found it was very limiting and took away much of the Pi's functionality so Raspbian was redownloaded instead.

Regular Maintenance

The LiPo batteries will have to regularly charged, as well as Trimble's tablet, Trimble's external 2.4 GHz radio, and the robotic total station's battery. The LiPo batteries must be at the same charge, preferably fully charged. The batteries try to balance out their voltage level, and if

they are voltage levels are at least close, the batteries will short. If batteries begin to swell, discontinue using immediately.

Spare Parts

- Arduino Mega – 1
- SmartLynx Stepper Motor Driver - 2
- Jumper Wires – MM MF FF – 10 each
- LiPo Battery – 1
- Markers – 2 of each color
- Raspberry Pi 2 – 1
- Flat-head Screwdriver – 1
- Stepper Motors - 2

Acknowledgement

We would like to thank Mark Winger of PSBI for being an amazing sponsor. Dr. Gupta for guidance and advice throughout the project. The CISCOR lab for donation of the mobile robot. Finally, we would like to thank Rob Miller of Florida Building Point and Trimble for the donation of the total station suite.