Quadrilateral Assembly / Usage Guide

This document serves as an initial user guide for the quadrilateral project from the CHROME Lab

Assembly

Parts list:

A. Pololu Force-Sensing Linear Potentiometer (FSLP), cut tabs off – 4
B. Rotary Potentiometer – 1 (there are many models that can work. Device may need modification to fit different models – see appendix)
C. Arduino Uno – 1
D. Breadboard and jumper wires
E. 3D printed quad pieces – 3 regular inner lengths, 3 regular outer lengths, 1 inner length with potentiometer modifications, 1 outer length with potentiometer modifications

Figure 1: Pololu FSLP

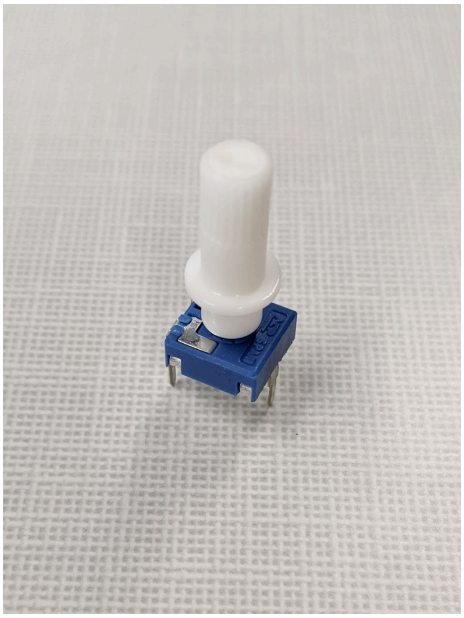


Figure 2: Rotary Potentiometer

Figure 3: 3D printed lengths

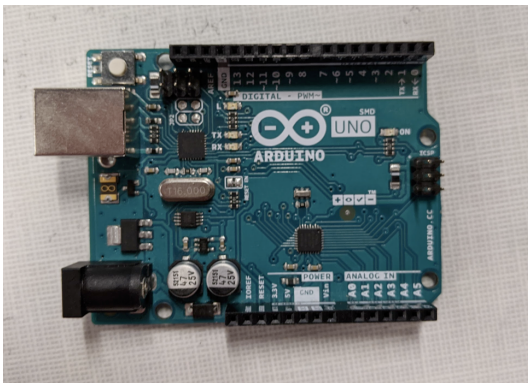
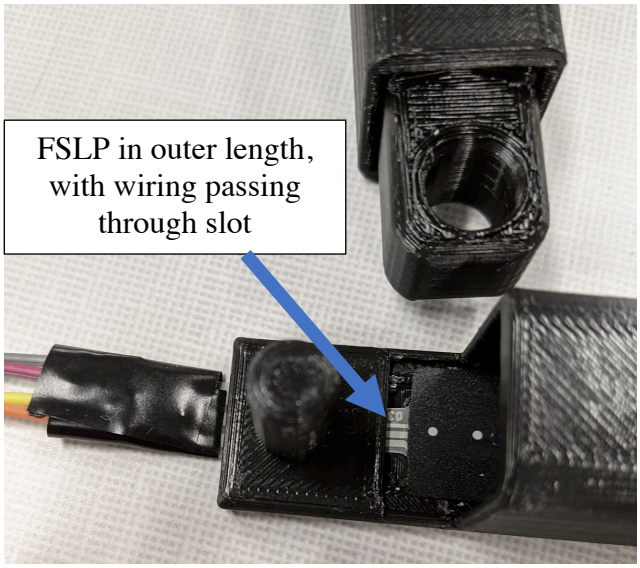


Figure 4: Arduino Uno

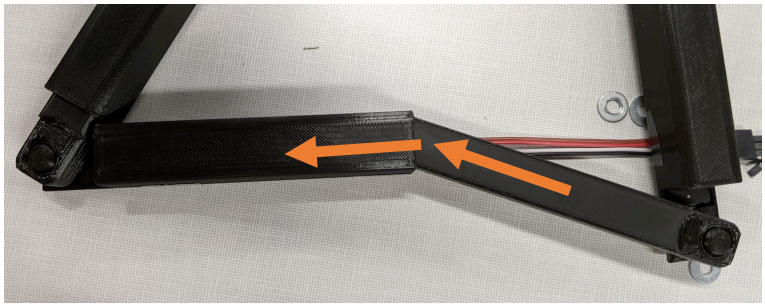
Assembly:

1. Ensure that all parts are collected and printed
2. Apply one FSLP to the bottom side of each outer length, using the adhesive on the FSLP itself
    a. There is a slot in the outer length for the wiring to pass through

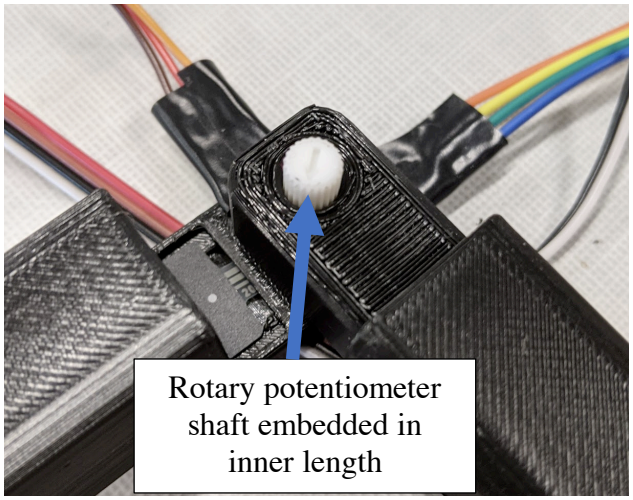


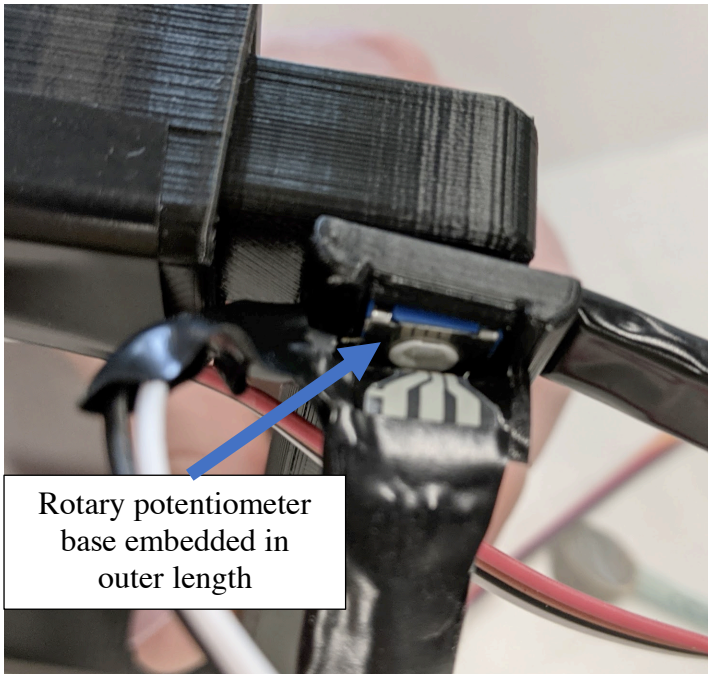
b. Firmly press down to ensure attachment

3. Attach the inner and outer lengths at corners and insert inner into outer
    a. Both have a small notch to keep the inner from falling out of outer, might require an angled entry

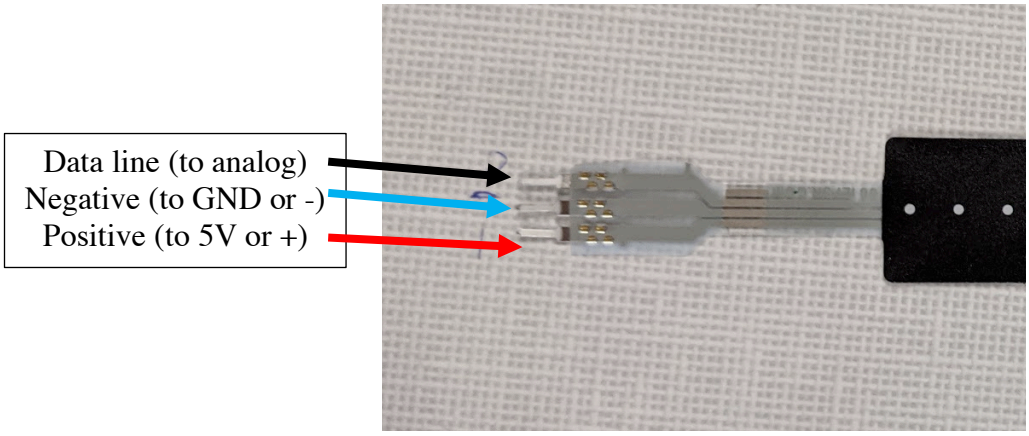


4. With the potentiometer-adjusted lengths, slot the base into the outer length and the shaft into the inner length





5. Wiring: each potentiometer (length and rotary) require power – wire 5V and GND to "+" and "–" of breadboard. Using jumper wires and electrical tape when necessary, wire the FLSPs and rotary potentiometer to power and the analog inputs of the Arduino



a. Rotary potentiometers generally have the same wiring schematics, positive and negative and data. Wire them accordingly

6. This wiring can be messy – some can be taped to the underside to clean up the appearance

Software

Files:

A. Arduino IDE with quad.ino

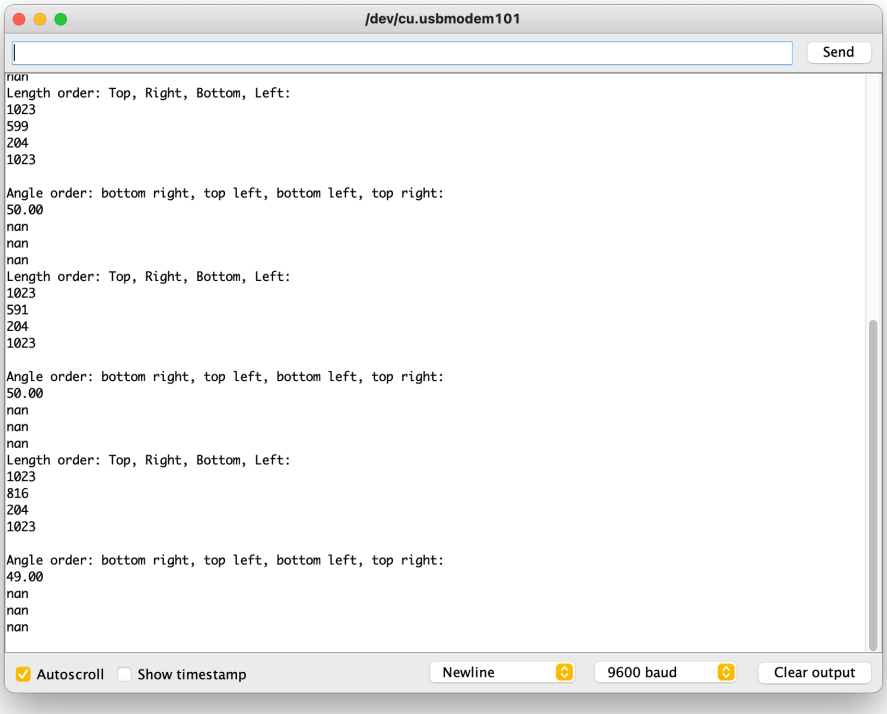
B. Python IDE (such as visual studio code or jupyter notebook) for speak.py

Setup:

1. Open quad.ino and change input values as needed

```
//Initialization of four length sensors (left, bottom, right, top) and FSR for control of data
//THESE WILL BE DIFFERENT FOR YOUR WIRING, change the value as needed
const int left = A3;
const int bottom = A5;
const int top = A2;
const int right = A4;
const int button_fsr = A1; // ignore if no FSR is wired as a action-button
const int pot_pin = A0;
//const int ledPin = 9;
```

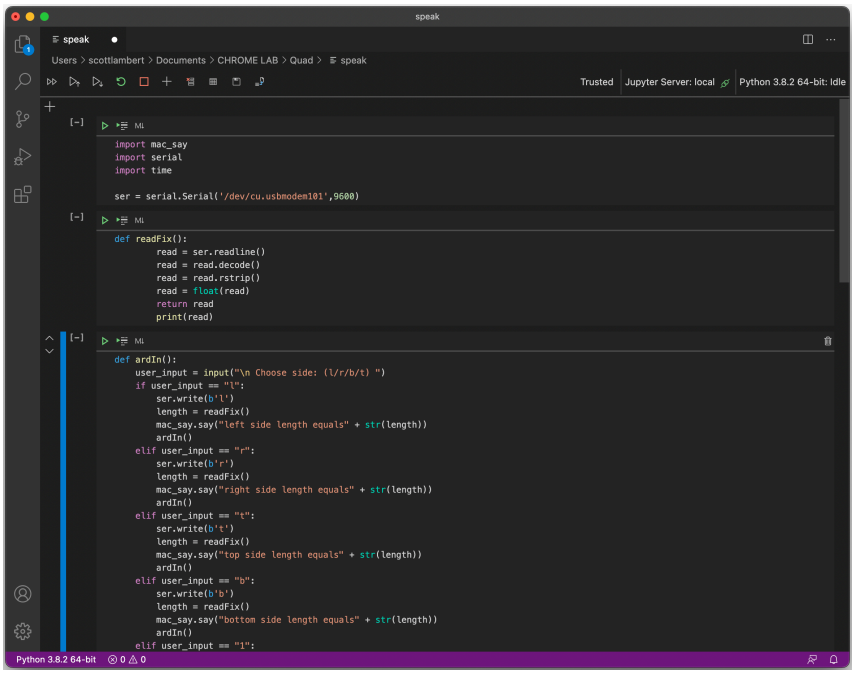
2. Uncomment wiring validation (directly after `void loop()`) and upload to Arduino board. Open serial monitor to observe results of validation



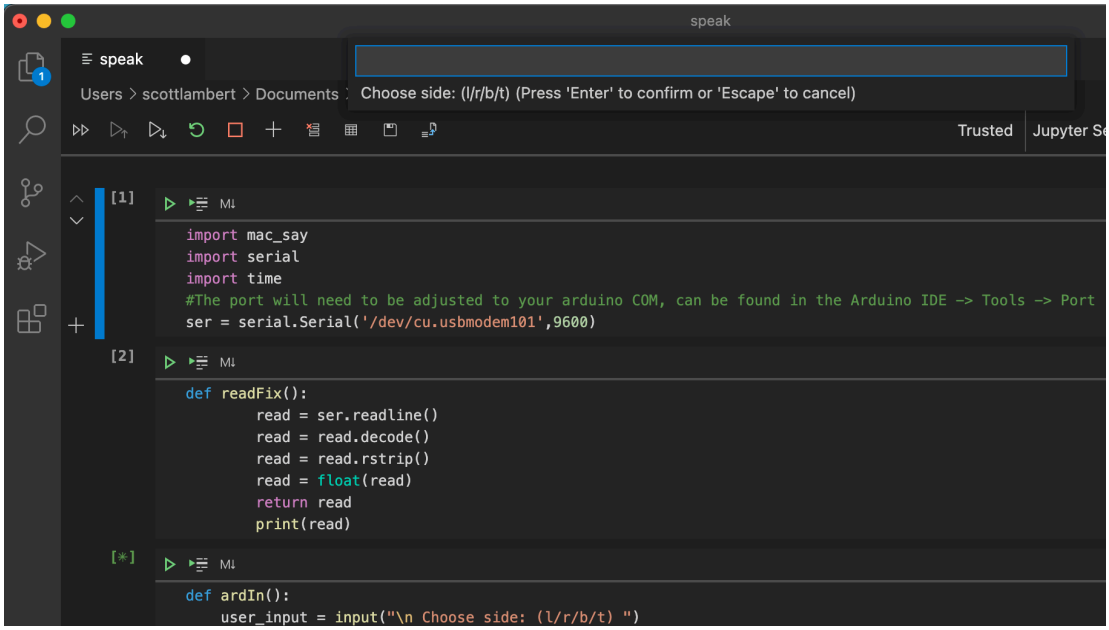
a. Some angles might be reported as “nan” as the mapping and angle calculations need adjustment for each device – see appendix

b. Re-comment out the validation after calibration and re-upload quad.ino

3. Open speak.py in any python IDE, jupyter notebook in visual studio code (VSC) is used for this tutorial



a. Adjust the serial port in the first block to the Arduino port (can be found in Arduino IDE -> Tools tab -> Port
b. After quad.ino has been uploaded, run the speak.py (can be run at once or in steps)
c. If using VSC, an input box will appear. Inputting a selection should announce that data value (left side, right side, angle 1, angle 2, etc.)



Appendix

Rotary Potentiometer Fitting:

1. The files “quentin_inner_pot.ipt” and “quentin_outer_pot.ipt” may need modification to fit the chosen pot.
2. Open the files in Inventor (or other 3D modeling software) and adjust the square slot in the outer to fit the measurements of the base.
3. Adjust the radius of the circle extrusion in the inner to fit the shaft of the potentiometer.
    a. The inner should be snug, such that the shaft rotates with the inner length

Potentiometer Calibration:

FSLP Calibration:

1. Measure length of each side at smallest size in inches or cm
2. Record Arduino output (0-1024) from validation
3. Extend all lengths to maximum size and record
4. Record new Arduino output (0-1024, but should be higher than step 2 value)
5. Use mmap function in quad.ino to calibrate (need to change values in multiple places in code):
    a. At beginning of loop, after validation, the following lines:

```
top_len = mmap(analogRead(top),213,964,5.26,8.34);
right_len = mmap(analogRead(right),375,957,5.30,8.2);
bottom_len = mmap(analogRead(bottom),230,970,5.27,8.21);
left_len = mmap(analogRead(left),220,986,5.25,8.32);
```

    Change values to the values from the above steps in this order:

    Mmap(analogRead(*side*), step2 value, step 4 value, step 1 value, step 3 value)

    b. Copy the four modified lines and paste them over the similar lines after *if (button > 150)*, near line #100

Rotary Calibration

1. Create a 180° at the potentiometer corner
2. Record value from inputVal (0-1024) from validation
3. Create smallest angle at potentiometer corner (about 40°, varies by print and model)
4. Record value from inputVal (0-1024)
5. Change map values in the same way as FSLP calibration, in the same locations