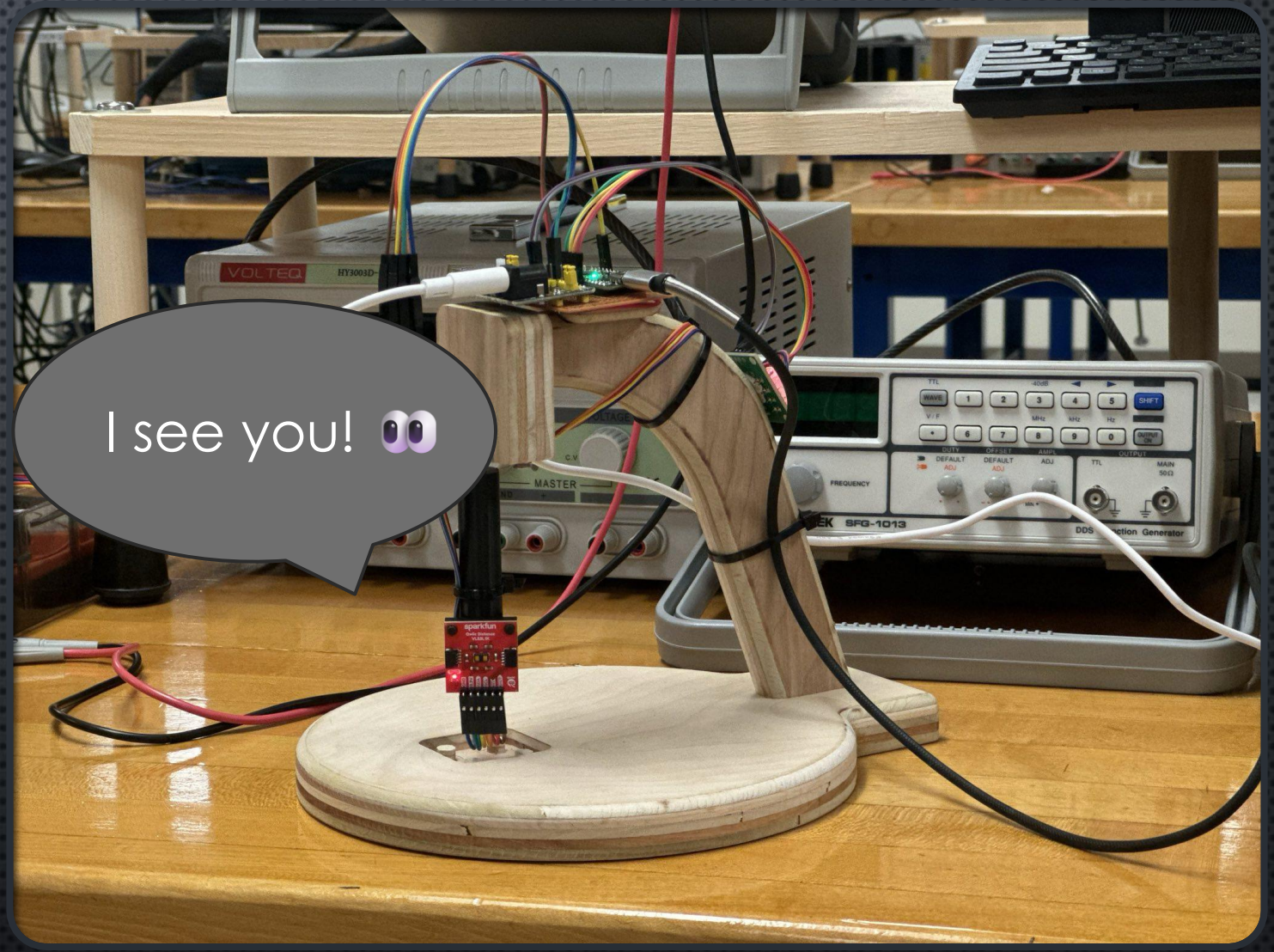
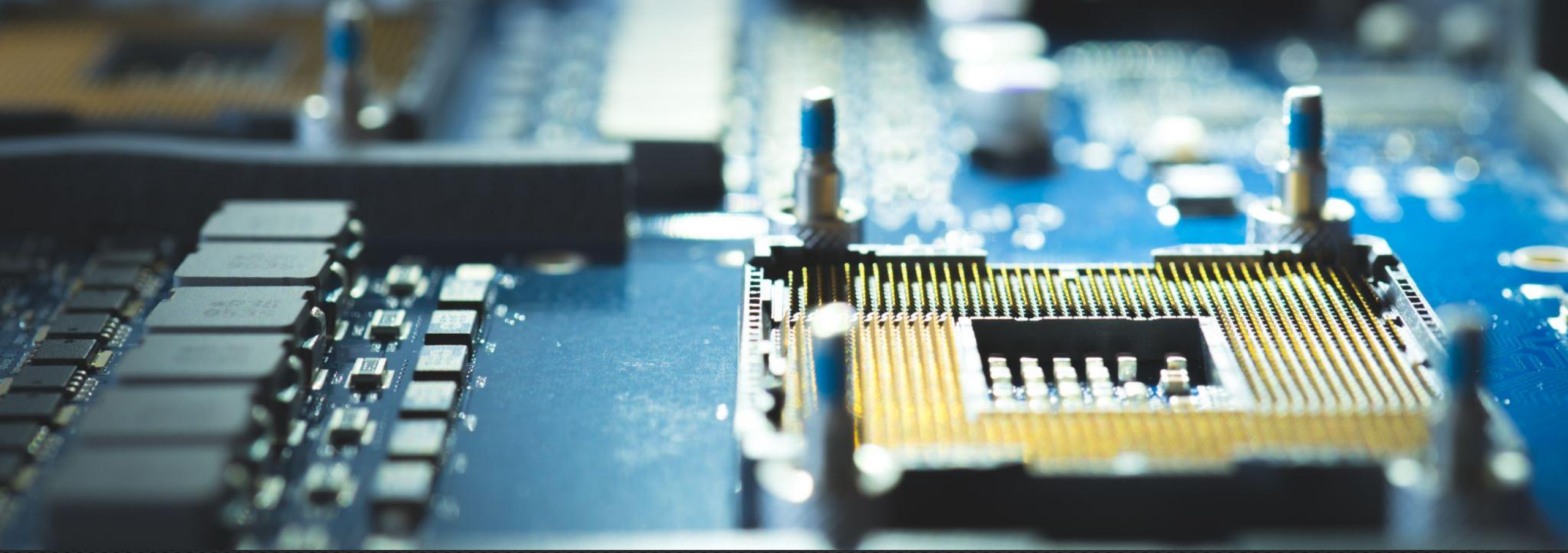
TIME OF FLIGHT SCANNER

DANIEL NALEPA, NATE SEIBOLD

AGENDA

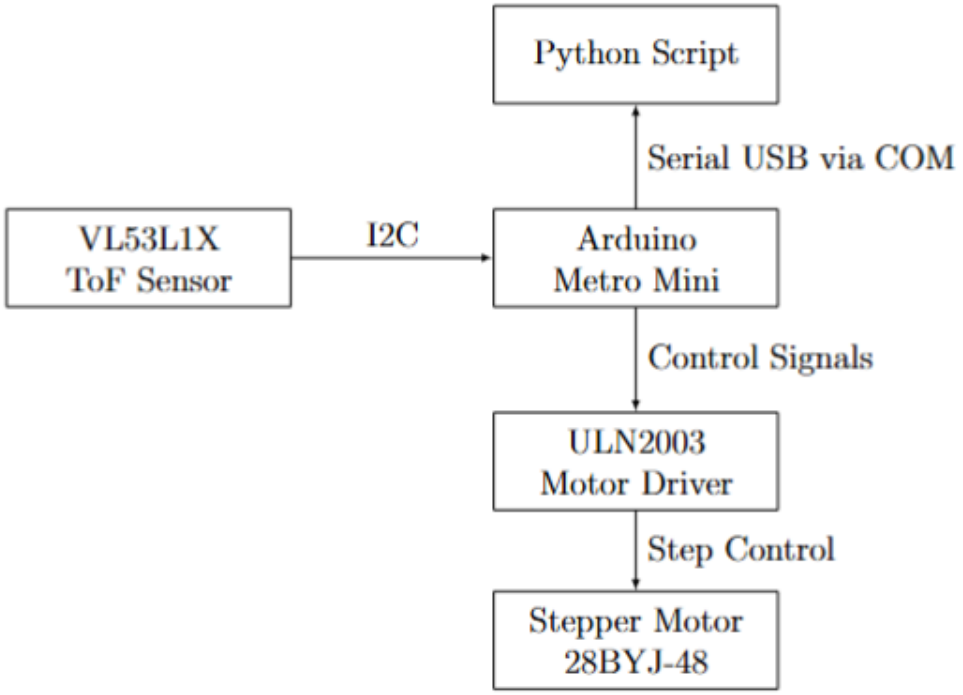
- Design Overview
- VL53L1X Sensor
- Design Iterations
- Assembly Design
- Code
- Demo





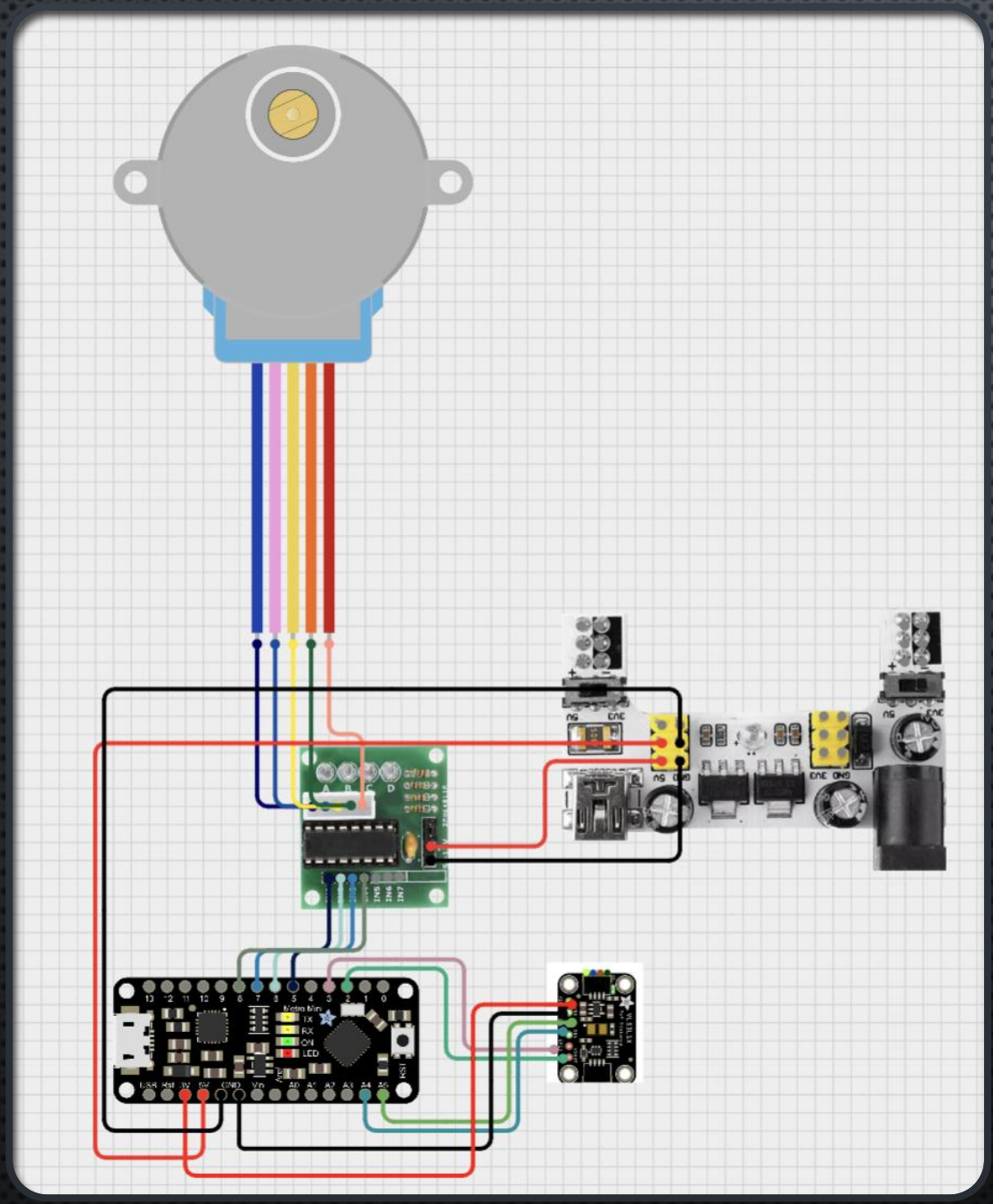
DESIGN OVERVIEW

System Diagram

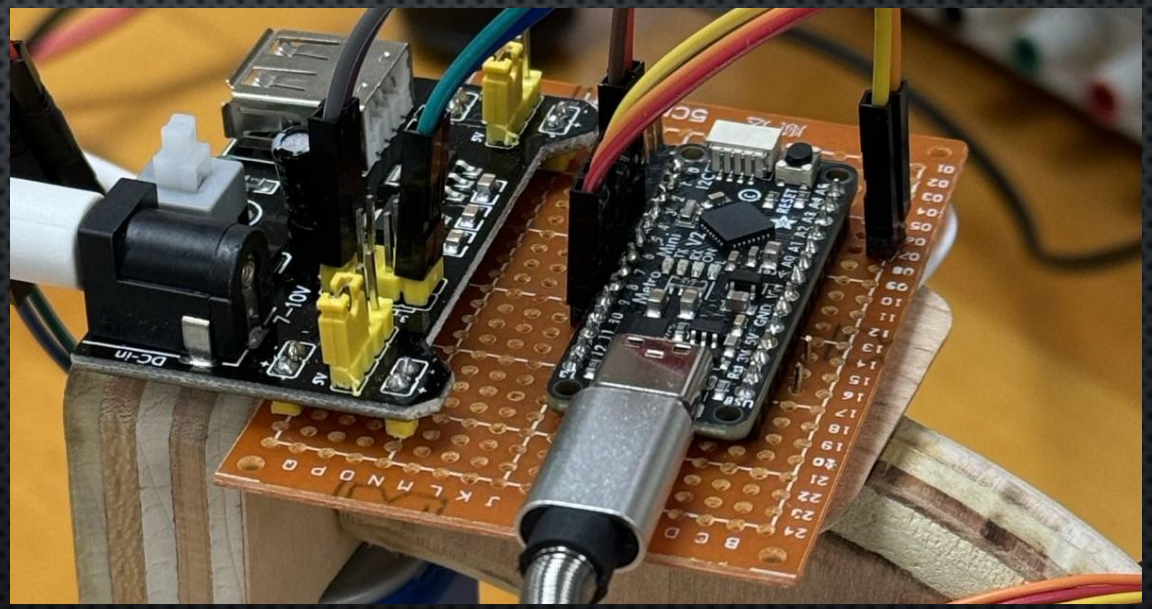


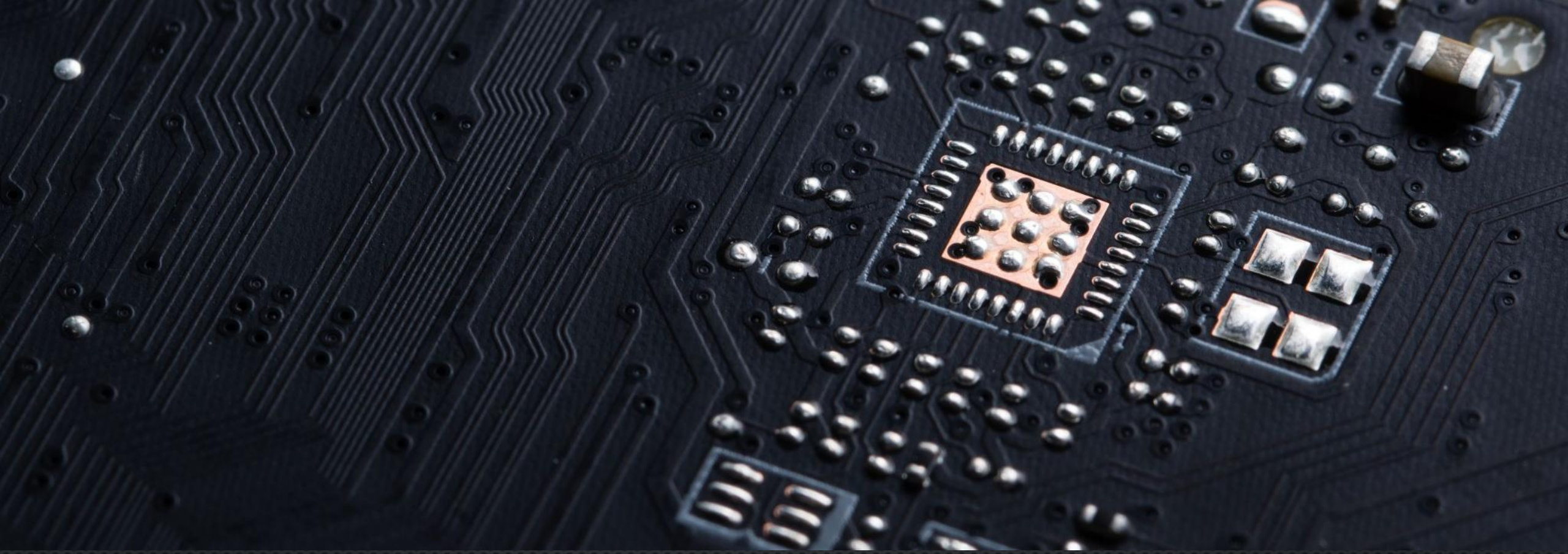
Hardware List

- Wooden base (CNC-fabricated)
- 3D-printed sensor arm
- ULN2003 motor controller
- 28BYJ-48 stepper motor
- Arduino Metro Mini (available from 332 lab)
- VL53L1X Time-of-Flight sensor **(Only used purchased part: $30)**
- USB-C cable
- Protoboard (soldered assembly)
- Assorted wiring
- Power module
- Zip ties (for cable management)



CIRCUIT DIAGRAM & PROTOBOARD





VL53L1X SENSOR

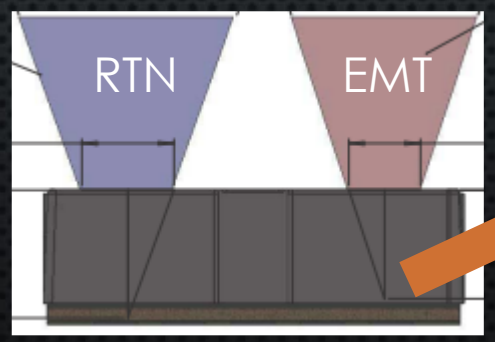
OVERVIEW

Table 5. **Maximum distance vs. distance mode under ambient light**

Distance mode	Max. distance in the dark (cm)	Max. distance under strong ambient light (cm)
Short	136	135
Medium	290	76
Long	360	73

- 940 nm Class 1 Laser Emitter
- Array of Single Photon Avalanche Diodes
- records time between emission and detection
- Measurement Range: 5-4000 mm
- Accuracy of 1mm and 50 Hz ranging frequency
- Typical FOV of 27 degrees

$$d = \frac{ct}{2}$$



DESIGN ITERATIONS

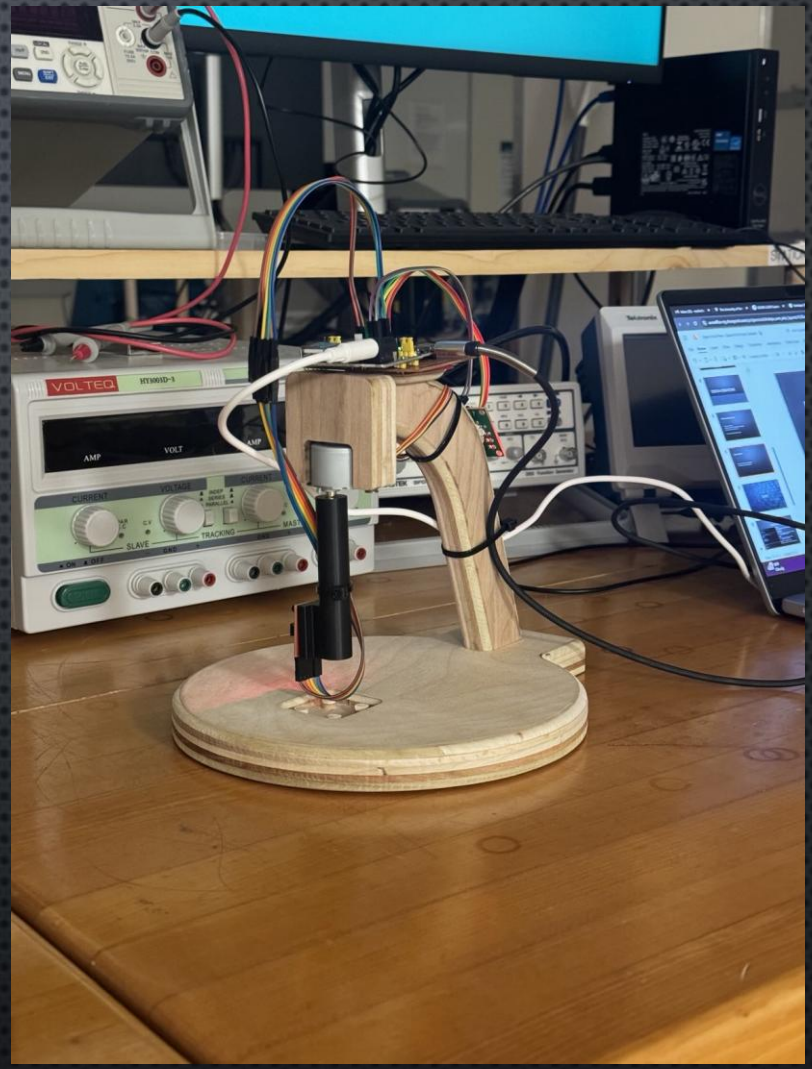
Initial Mirror Design

- Stationary ToF Sensor
- Mirror Suspended from Stepper Motor at 45 degrees
- Allows for Continuous Rotation

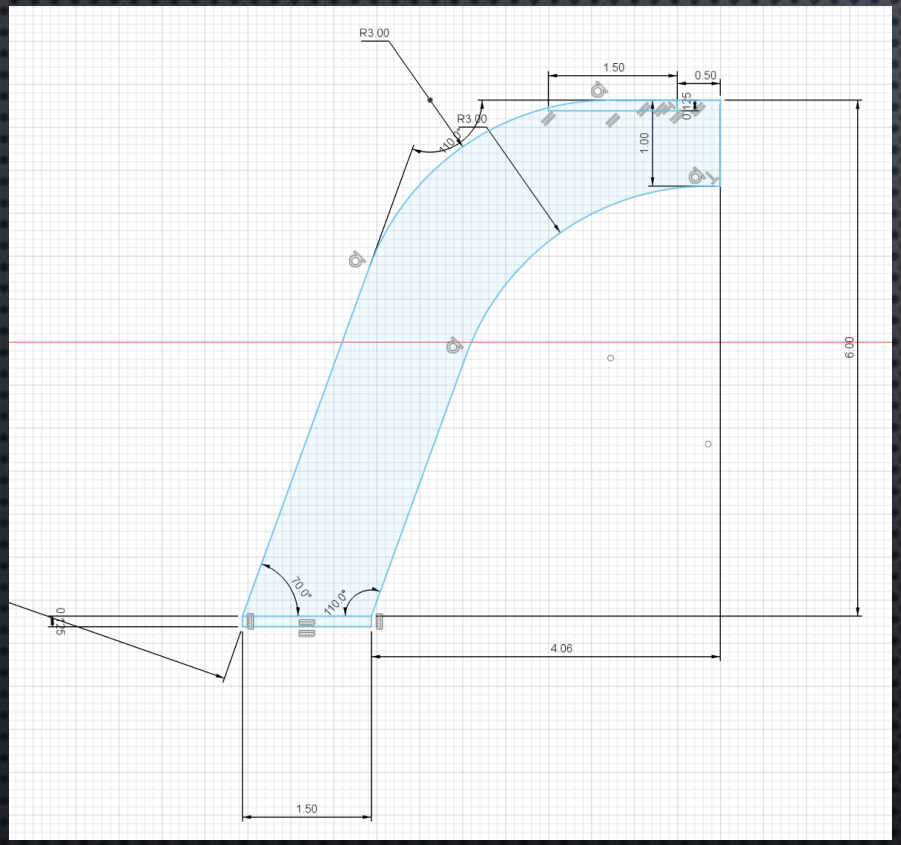
- ToF Sensor Measured Mirror Distance

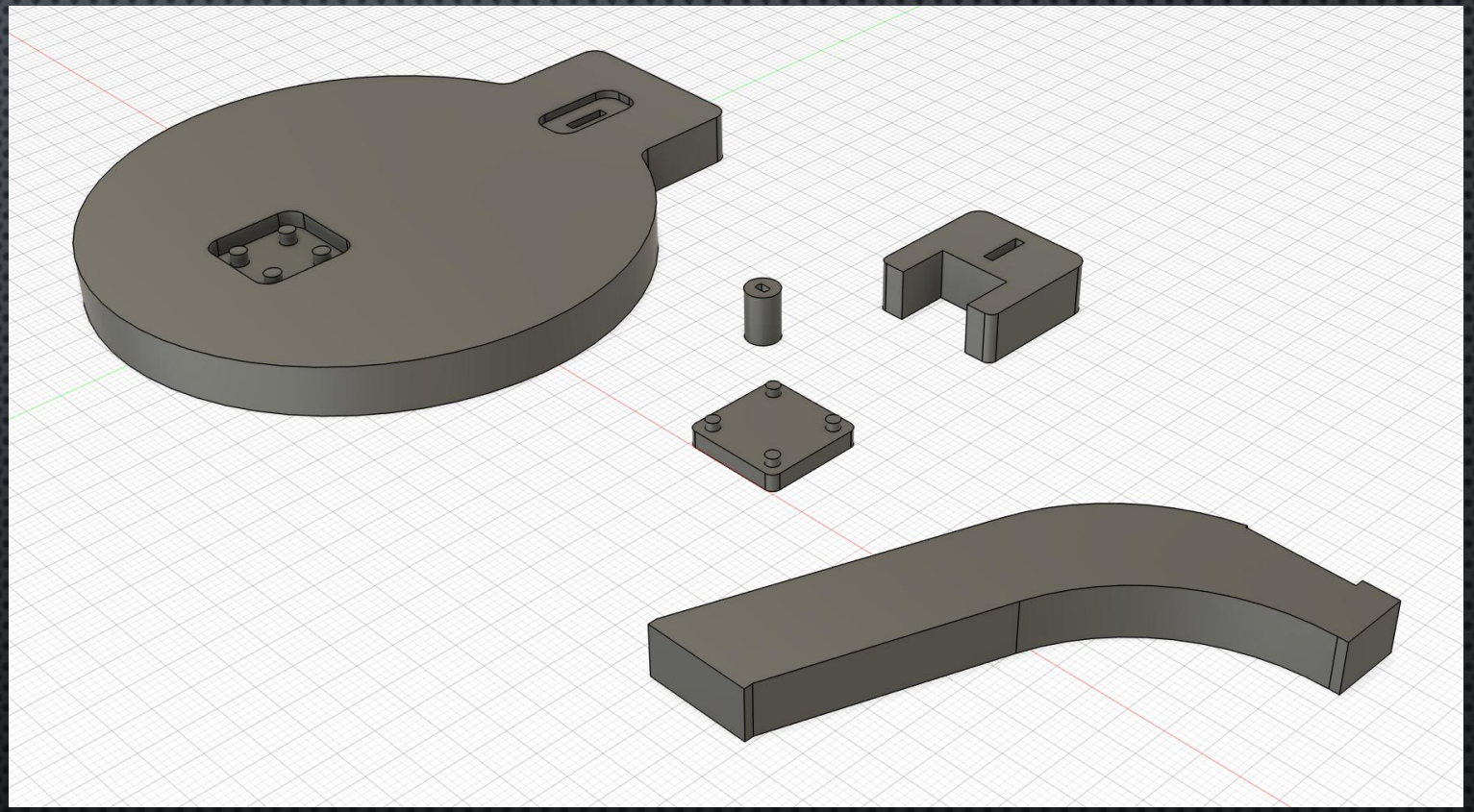
Final Design

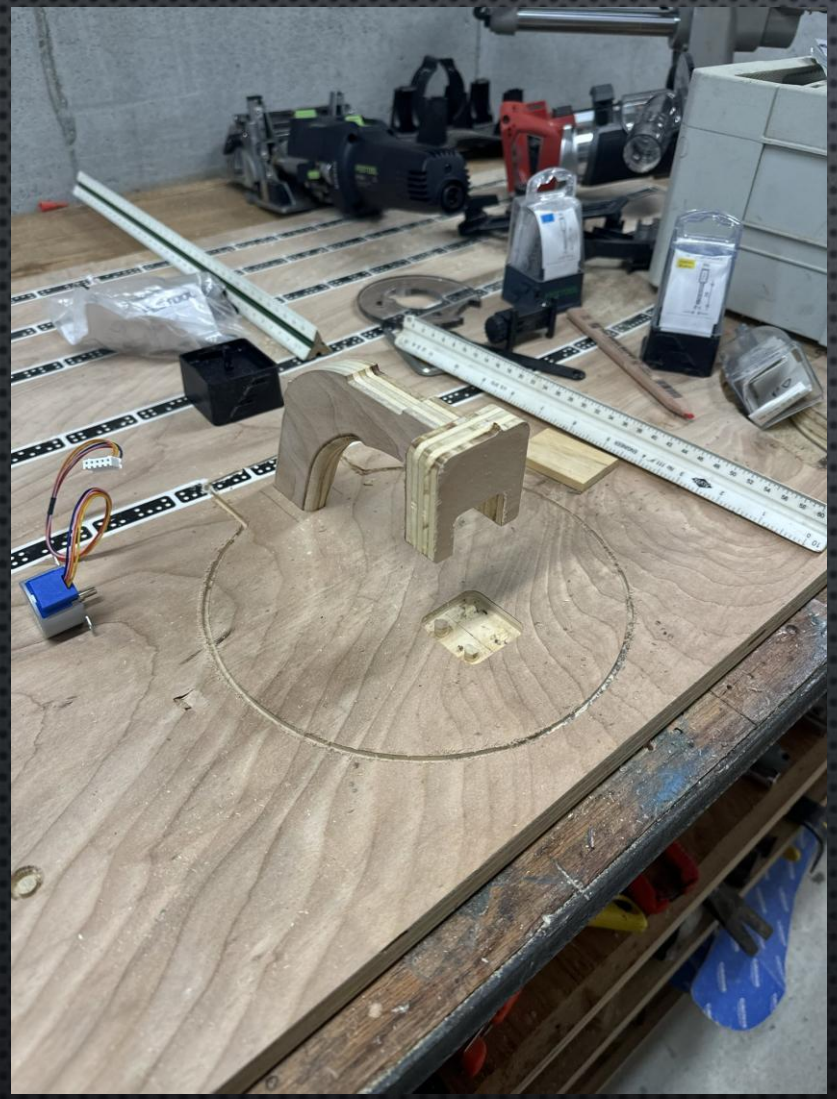
- Rotating ToF Sensor on Stepper Motor
- No Slip Ring: Limited to 360 degrees
- Addition of External Power Supply



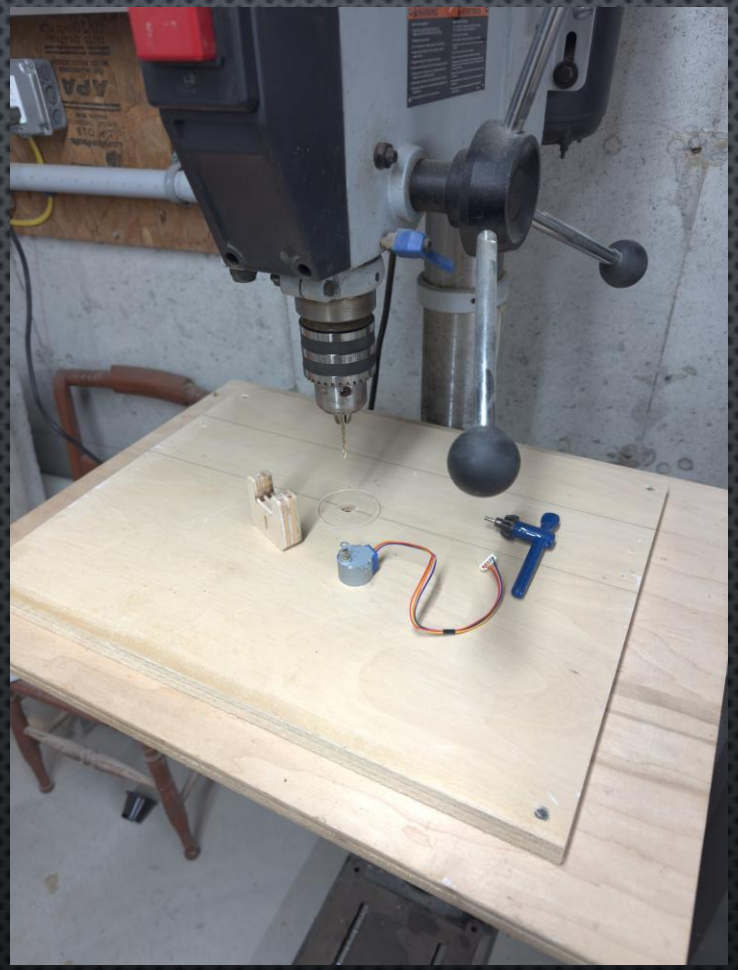
WOODEN BASE DESIGN

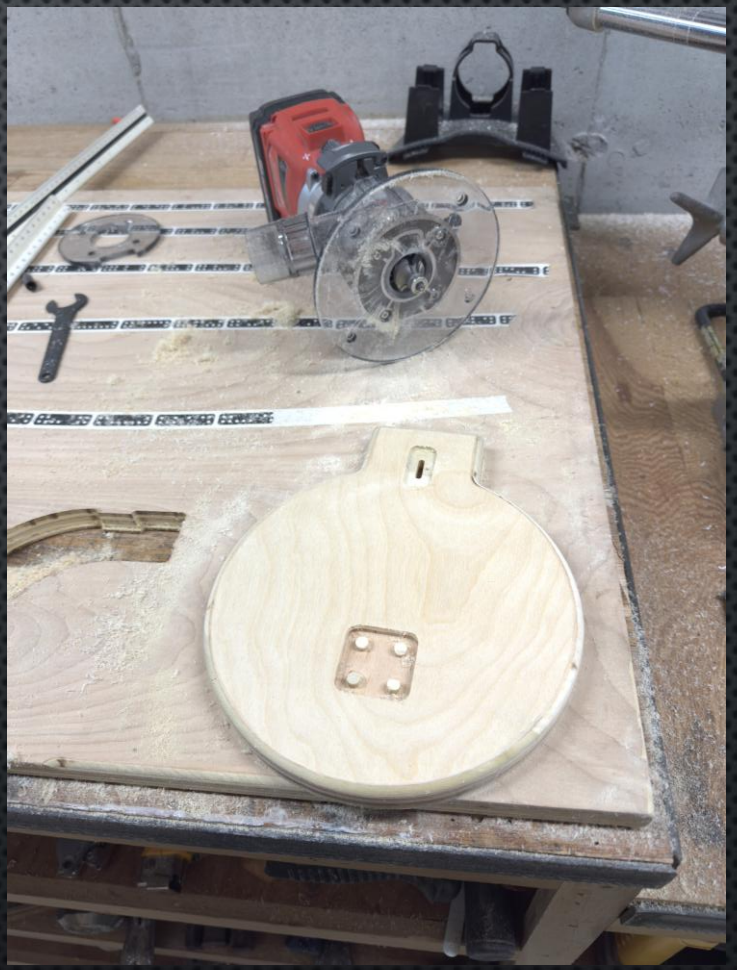

R3.00
1.50
0.50
R3.00
0.125
1.00
6.00
70.0°
110.0°
4.06
1.50



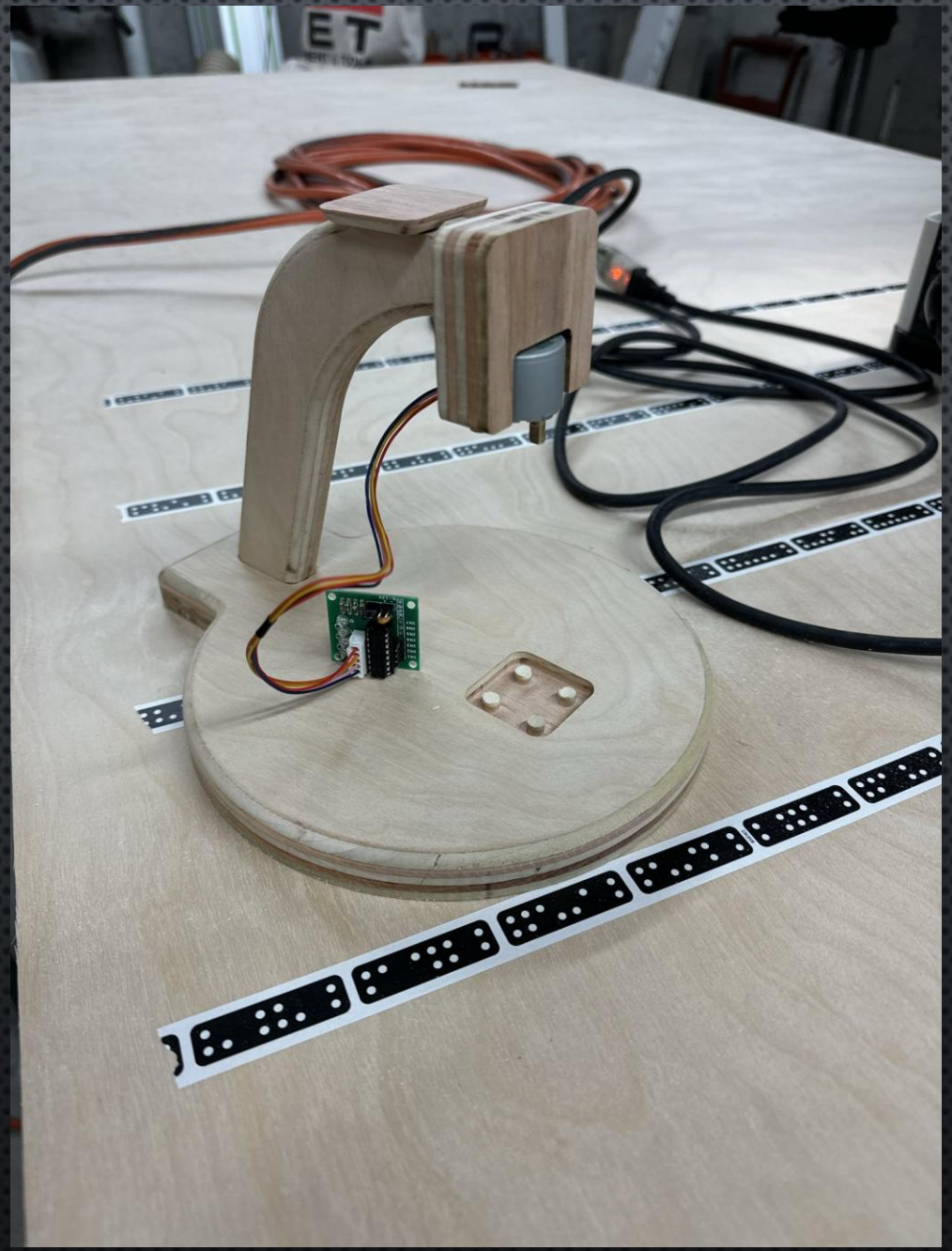



FESTOOL

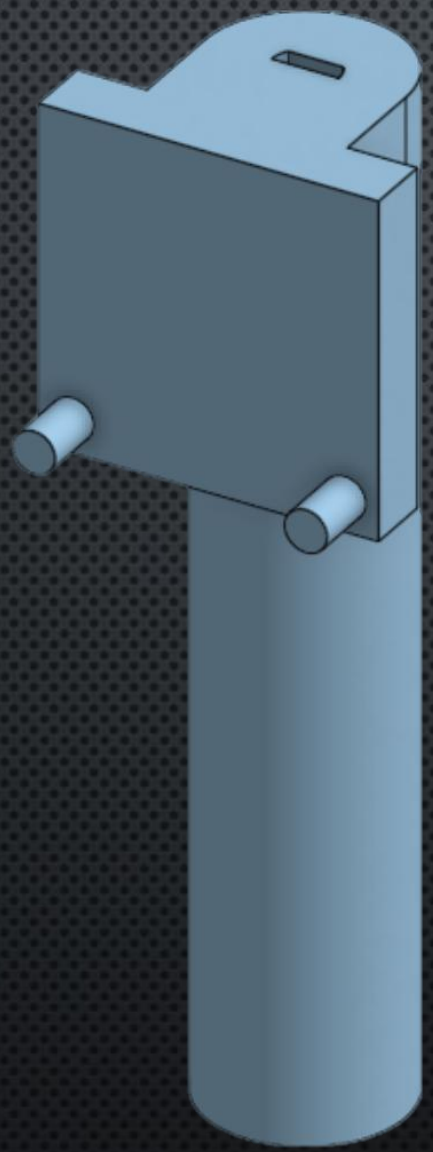


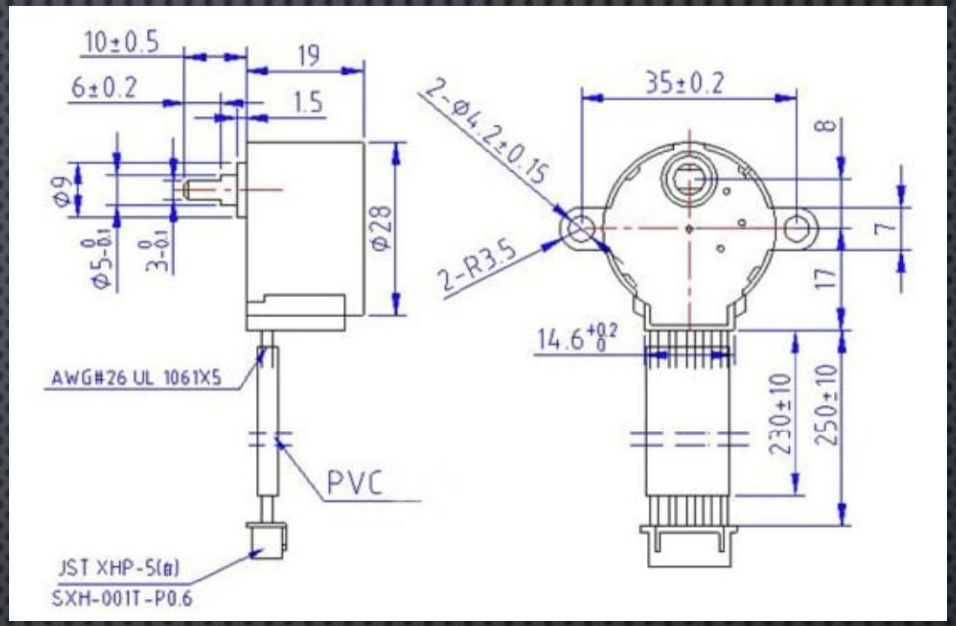


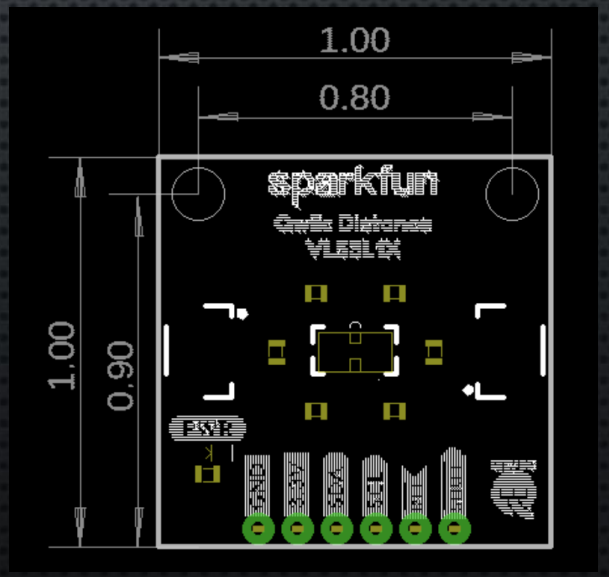

DOMINO



STEPPER ARM DESIGN




10±0.5
19
6±0.2
1.5
Ø9
Ø5-0.1
3-0.1
Ø28
AWG#26 UL 1061X5
PVC
JST XHP-S(a)
SXH-001T-P0.6
2-Ø4.2±0.15
2-R3.5
35±0.2
8
7
17
14.6+0.2
230±10
250±10


1.00
0.80
1.00
0.90
sparkfun


CODE

ARDUINO CODE

```
// define calibration function
// sensor needs to know 180 degrees when starting
void calibration() {
  // read initial distance
  distance = vl53.distance();

  // arm is known to be in this range, keep rotating until this distance is found
  while (distance <= 55 || distance >= 80) {
    // perform small steps, we don't know where the sensor is
    myStepper.step(1);
    delay(5);

    // remeasure distance
    if (vl53.dataReady()) {
      distance = vl53.distance();
      vl53.clearInterrupt();
    }
  }

  // the sensor will stop on the outer edge of the arm, add an extra 1.5 degrees to account
  myStepper.step(10);

  // update flag so loop can run
  calibrated = true;
```

Setup

1. Initializes sensor and calls calibration function

Calibration

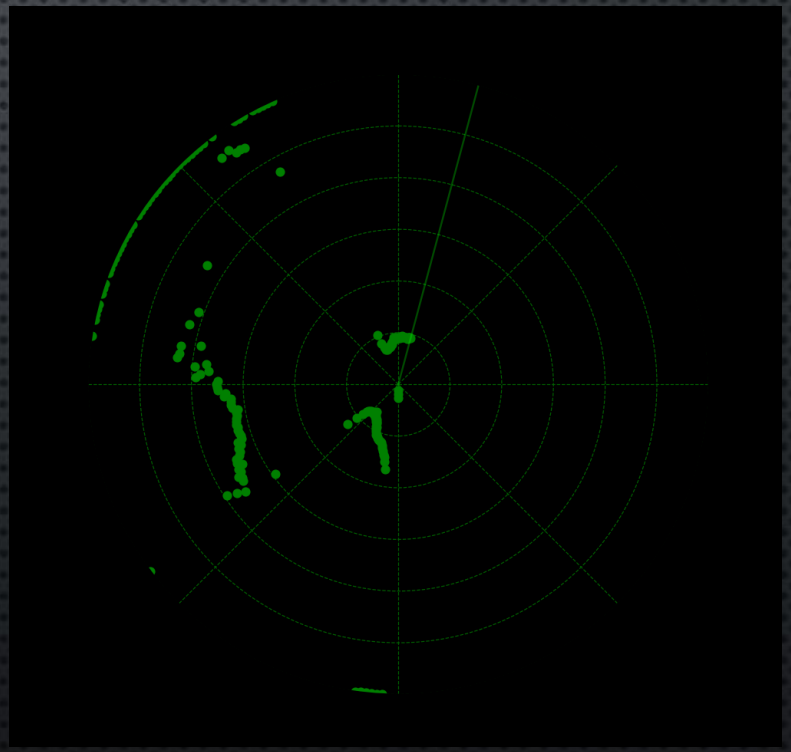
1. Rotates CCW until "wooden stem" of base is found (reference point)

Loop

1. Full CW & CCW rotations after calibration is complete
2. Steps ~1 degree each move
3. Prints angle and distance in CSV format

Python Code and Plotting

- Uses Matplotlib and Pyserial
- Establish Serial Connection
- Format Plot to Mimic Radar
- Parse CSV Data to Separate Distance/Angle
- Add Data to Array
- Animate Polar Plot
  - Refresh Plot Frequnetly While Checking Serial



DEMONSTRATION