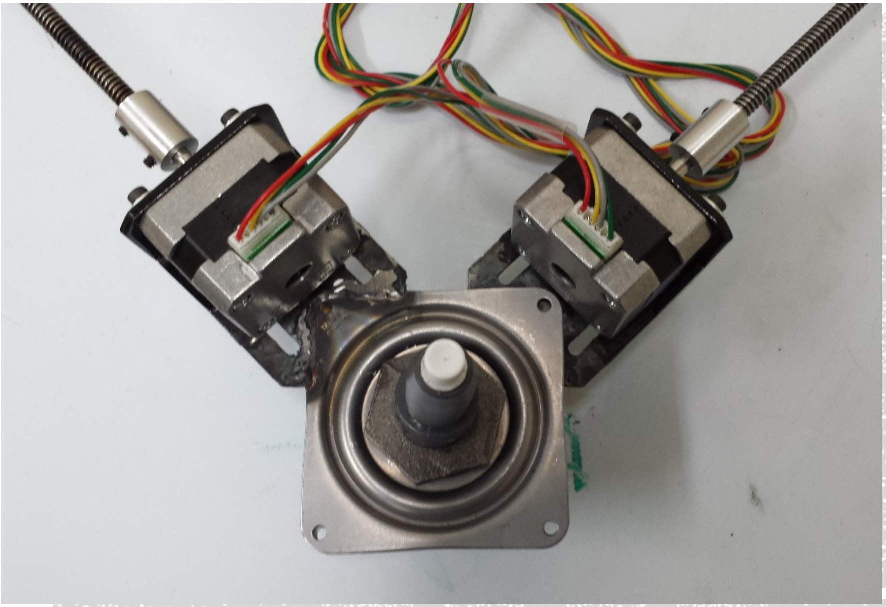
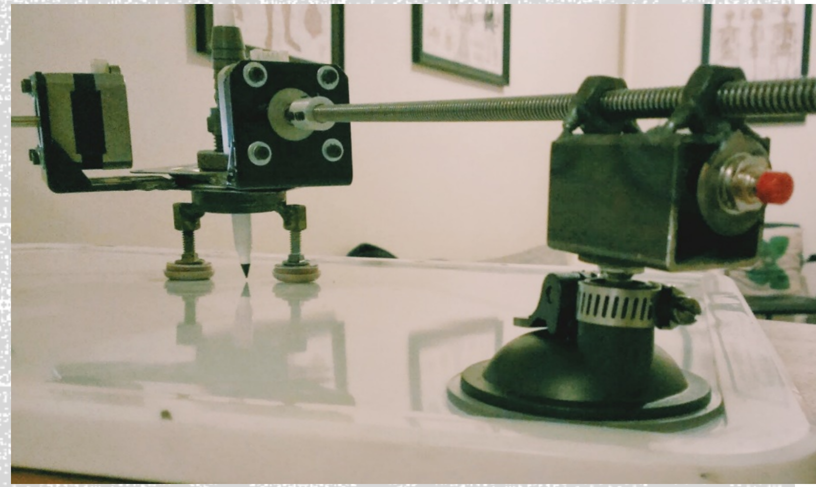
ME 5643 | MECHATRONICS PROJECT PROPOSAL: CNC PLASMA CUTTER PROOF OF CONCEPT





Andy Lynch, Rajiv Panday, Stephen Carter

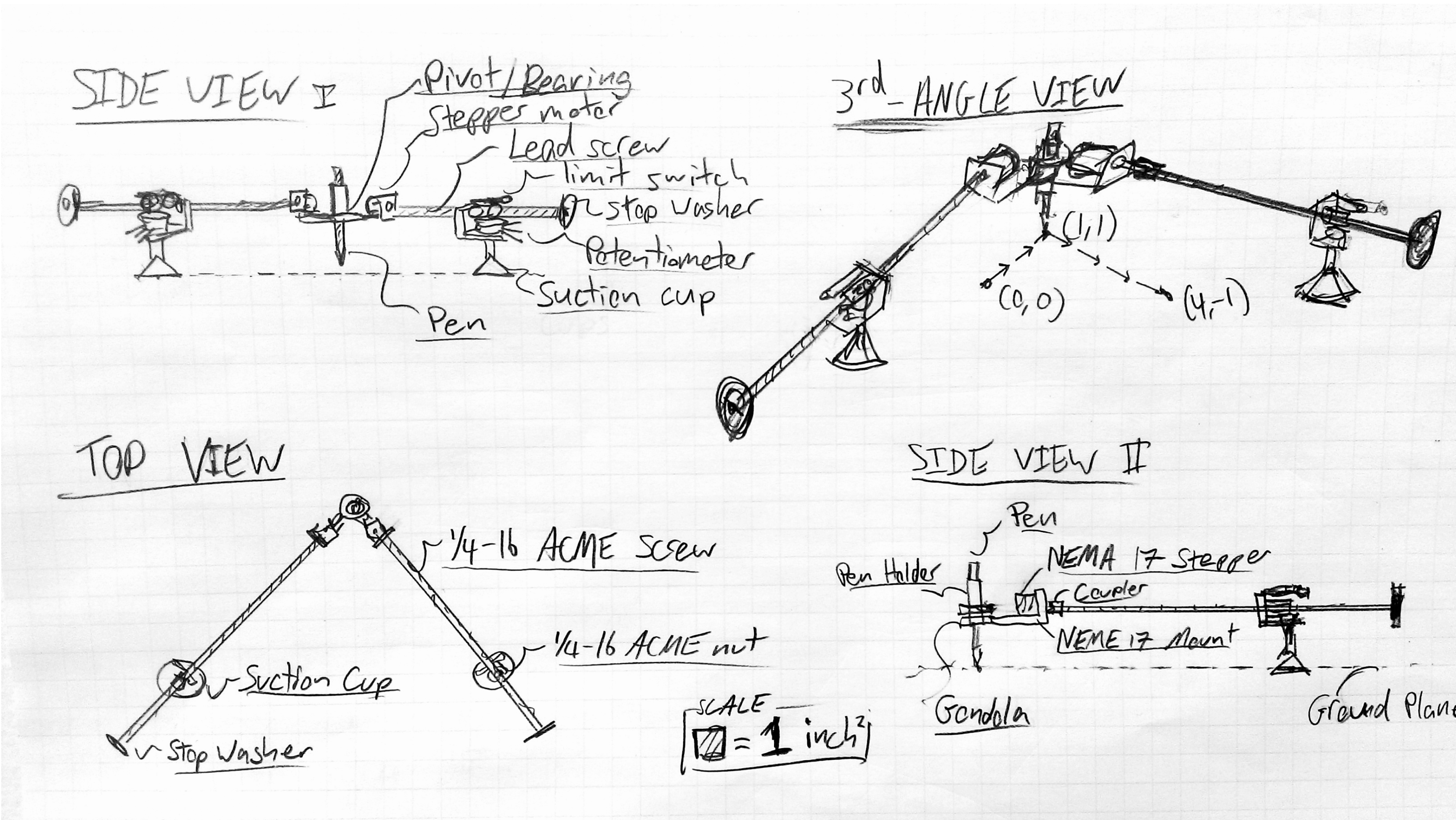
NYU Polytechnic School of Engineering

December 01, 2014

Professor SangHoon Lee

BAILEIGH PLASMA TABLE PT-22 – $4,495.00

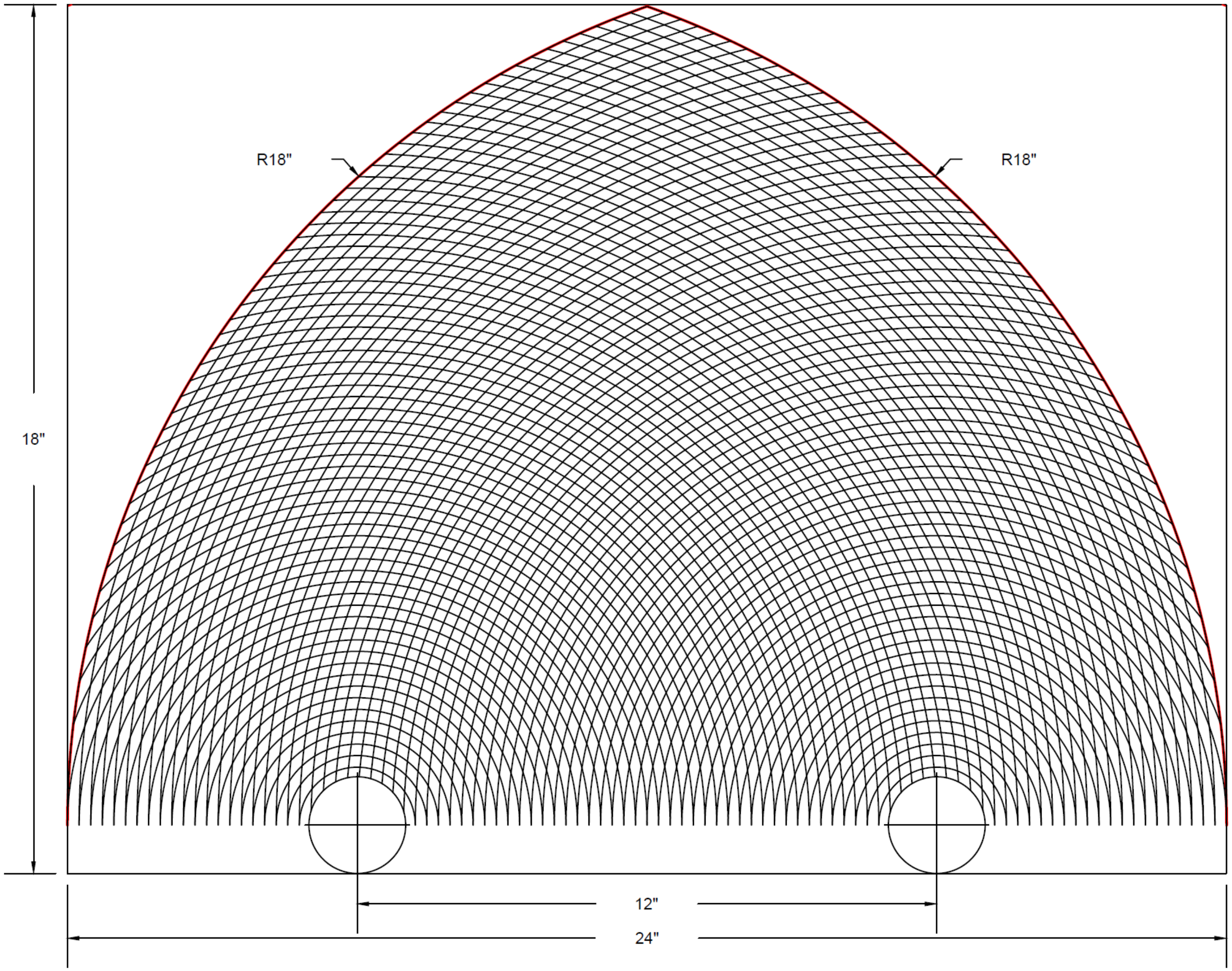
SKETCH OF PRODUCT



PRODUCT

- Computer Numerical Control Plasma Cutter
  - Rather than using an actual plasma cutter a pen/marker will be used as its simulate.
  - The design will be less expensive than a typical CNC plasma cutter.
  - The design for the plasma cutter's mechanism will be triangular rather than rectangular allowing it to be less rigid and therefore portable.

CONE OF RESOLUTION - 1



Radii

- Each Lead screw can sweep up to a radius of 1.5 ft.

Resolution

- Step size = 2% of the distance between the suction cups

LINE ALGORITHM

- Calculates new coordinates for the four optional directions.
  - Options: Motor A in, Motor A out, Motor B in, Motor B out.
- Check each option to determine which brings you closer towards the target coordinate.
- Of those options, select the option that stays closest to the line between the initial and target coordinates.

LINE ALGORITHM CODE

```
 * line algorithm to move both motors to a new position
 **/
void line(float newx,float newy) {
  float cx = 0; //current X
  float cy = 0;
  float dx=newx-px;
  float dy=newy-py;
  do{
    float cosA = (pow(A,2) - pow(B,2) - pow(C,2)) / (-2*B*C);
    float cosB = (pow(B,2) - pow(A,2) - pow(C,2)) / (-2*A*C);

    // List of x,y coordinates for 4 options for movement
    float xAin = - STEPSIZE * cosB + cx;
    float yAin = - STEPSIZE * sin(acos(cosB)) + cy;
    float xAout = STEPSIZE * cosB + cx;
    float yAout = STEPSIZE * sin(acos(cosB)) + cy;
    float xBin = STEPSIZE * cosA + cx;
    float yBin = - STEPSIZE * sin(acos(cosA)) + cy;
    float xBout = - STEPSIZE * cosA + cx;
    float yBout = STEPSIZE * sin(acos(cosA)) + cy;

    // Algorithm for choosing which of 4 option is best
    float cDist = sqrt(pow(dx-cx,2)+pow(dy-cy,2));
    float aInDist = sqrt(pow(dx-xAin,2)+pow(dy-yAin,2));
    float aOutDist = sqrt(pow(dx-xAout,2)+pow(dy-yAout,2));
    float bInDist = sqrt(pow(dx-xBin,2)+pow(dy-yBin,2));
    float bOutDist = sqrt(pow(dx-xBout,2)+pow(dy-yBout,2));

    float minDist = 100;
    int best = 0;
    if (cDist > aInDist){
      minDist = abs(dy*xAin-dx*yAin)/sqrt(pow(dy,2)+pow(dx,2));
      best = 1;
    }
    if (cDist > aOutDist){
      float test = abs(dy*xAout-dx*yAout)/sqrt(pow(dy,2)+pow(dx,2));
      if (test < minDist){
        minDist = test;
        best = 2;
      }
    }
    if (cDist > bInDist){
      float test = abs(dy*xBin-dx*yBin)/sqrt(pow(dy,2)+pow(dx,2));
      if (test < minDist){
        minDist = test;
        best = 3;
      }
    }
    if (cDist > bOutDist){
      float test = abs(dy*xBout-dx*yBout)/sqrt(pow(dy,2)+pow(dx,2));
      if (test < minDist){
        minDist = test;
        best = 4;
      }
    }

    // implements best option
    switch (best) {
    case 1: // Ain
      oneStep(0,0);
      cx = xAin;
      cy = yAin;
      A -= STEPSIZE;
      break;
    case 2:  // Aout
      oneStep(0,1);
      cx = xAout;
      cy = yAout;
      A += STEPSIZE;
      break;
    case 3:  // Bin
      oneStep(1,0);
      cx = xBin;
      cy = yBin;
      B -= STEPSIZE;
      break;
    case 4: // Bout
      oneStep(1,1);
      cx = xBout;
      cy = yBout;
      B += STEPSIZE;
      break;
    default:
      Serial.println("Hrrrrrmmmmmm!?");
    }
  }
  while (abs(dx - cx) > STEPSIZE*STEPMULT || abs(dy - cy) > STEPSIZE*STEPMULT);
  px+=cx;
  py+=cy;
}
```

CALIBRATION ALGORIGTHM

- Extend both lead screws to their maximum length.
- Continually retracts both lead screws until they form a 45-45-90 triangle.
- At this position, the coordinates are initialized to (0,0)

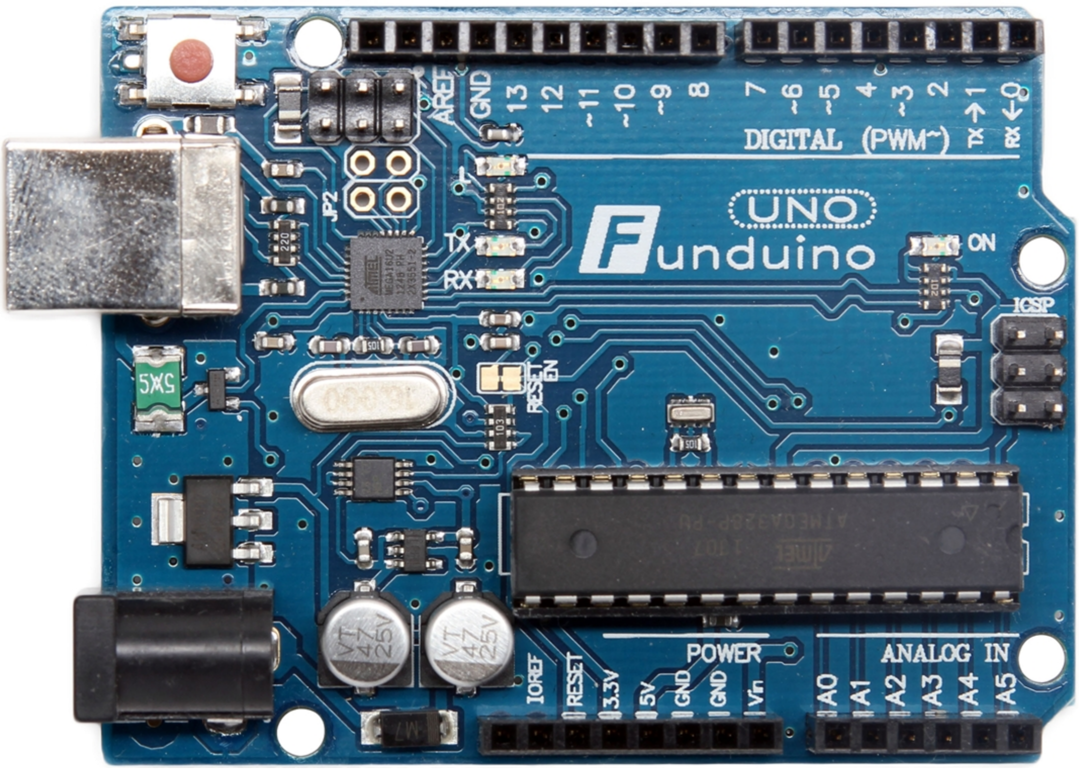
```
 * Self Calibration Sequence
 **/
void calibrate(){
  fullextend = 1;
  while( digitalRead(limitA) == 0 || digitalRead(limitB) == 0 ){
    if (digitalRead(limitA)== 0){
      oneStep(0,1);
    }
    if (digitalRead(limitB)== 0){
      oneStep(1,1);
    }
  }
  long countA=0;
  long countB=0;
  volatile float a = anglea();
  volatile float b = angleb();
  while( (a-45)>0 || (b-45)>0 ){
    a = anglea();
    b = angleb();

    if (a - 45 < 0){
      oneStep(0,1);
      countA++;
    }
    if (b - 45 < 0){
      oneStep(1,1);
      countB++;
    }
    if (a - 45 > 0){
      oneStep(0,0);
      countA--;
    }
    if (b - 45 > 0){
      oneStep(1,0);
      countB--;
    }
    A = MAXLENGTH + countA * STEPSIZE;
    B = MAXLENGTH + countB * STEPSIZE;
  }
  C = ((A + B)/2)*sqrt(2);
  output("A",A);
  output("B",B);
  output("C",C);
  fullextend = 0;
}
```

PARTS & COST

Hardware Parts and Cost List			
Quantity	Part	Unit Cost	Total Cost
1	1/4"-16 ACME Threaded Rods - 3ft	8.43	8.43
2	1/4"-16 ACME Hex Nut	2.08	4.16
2	1K Linear Taper Rotary Potentiometer	0.99	1.98
2	Suction Cup Mount	6.84	13.68
1	Funduino Uno R3	8.96	8.96
1	Proto Shield Prototype Kit Shield	5.68	5.68
1	JoyStick Breakout Module	2.69	2.69
1	AC Adapter Power Supply	12.85	12.85
1	Ballpoint Pen	0.99	0.99
1	10 Piece Hose Clamp Set 4 Sizes	3.50	3.50
2	Shaft Couplers	5.00	10.00
2	NEMA 17 Stepper Motor	8.99	17.98
2	NEMA 17 Easy Driver Shield	6.99	13.98
1	#8 Nylon Washer - 5 Pack	0.58	0.58
1	Stock Steel Rod & Angle Iron	4.00	4.00
1	Momentary Push Button Switch - 3 Pack	2.80	2.80
1	Linear Servo	5.99	5.99
1	Door Hinge	5.99	5.99
1	MultiColor Wire	9.99	9.99
	Total		**134.23**

MICROCONTROLLER



We are going to use the Funduino Uno R3, which is functionally identical to the Arduino Uno

MECHANISMS TO PREVENT DAMAGE

- Two limit switches are used to tell the program to stop feeding out the lead screw to ensure that it does not fall out of the mounting nut.
- Emergency Switch to disable motors

SERIAL INTERFACING

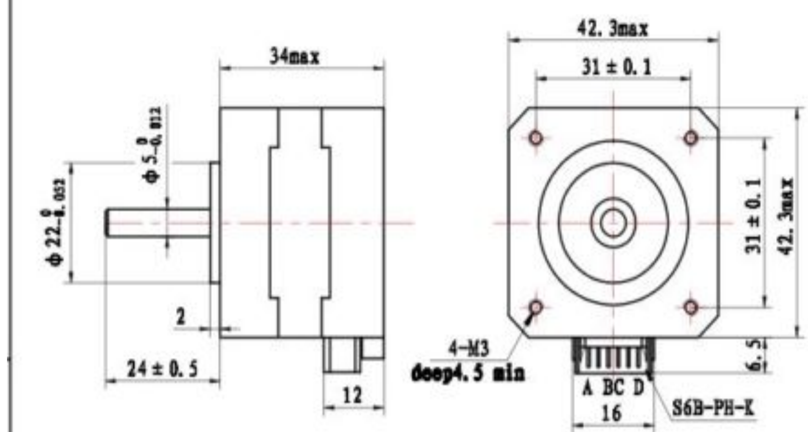
- Arduino communicates via Serial Interface.
- Uses a parsing algorithm to take in lines of G-code and tells the G-code sender to wait until current action is completed.
- In turn, G-code sender waits for “handshake” from the Arduino to enter next line of code.

```
 * display helpful information
 */
void help() {
  Serial.print(F("PolarGraphGcodeCNC"));
  Serial.println(VERSION);
  Serial.println(F("Commands:"));
  Serial.println(F("G00 [X(steps)] [Y(steps)] [F(feedrate)]; - linear move"));
  Serial.println(F("G01 [X(steps)] [Y(steps)] [F(feedrate)]; - linear move"));
  Serial.println(F("G04 P[milliseconds]; - delay"));
  Serial.println(F("G90; - absolute mode"));
  Serial.println(F("G91; - relative mode"));
  Serial.println(F("G92 [X(steps)] [Y(steps)]; - change logical position"));
  Serial.println(F("M100; - this help message"));
  Serial.println(F("M101; - calibrate polargraph"));
  Serial.println(F("M114; - report position and feedrate"));
}
```

ACTUATOR

NEMA17 17 Stepper Motor

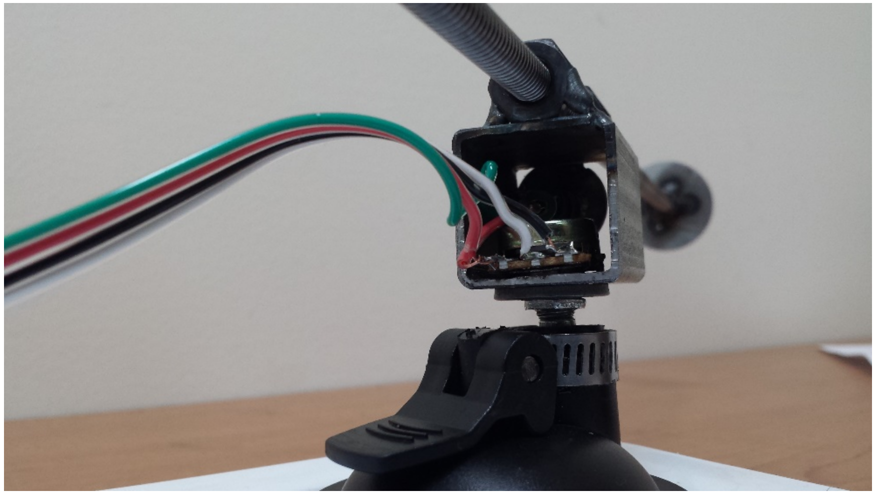
General specification		Electrical specification	
Step angle	1.8°	Rated voltage	12V
Number of phase	2	Rated current	0.4A
Insulation resistane	100MΩmin. (500V DC)	Resistance per phase	30Ω ± 10%
Insulation class	Class B	Inductance per phase	37mH ± 20%
Rotor inertia	38g.cm²	Holding torque	260mN.m
Mass	0.2kg	Detent torque	12mN.m



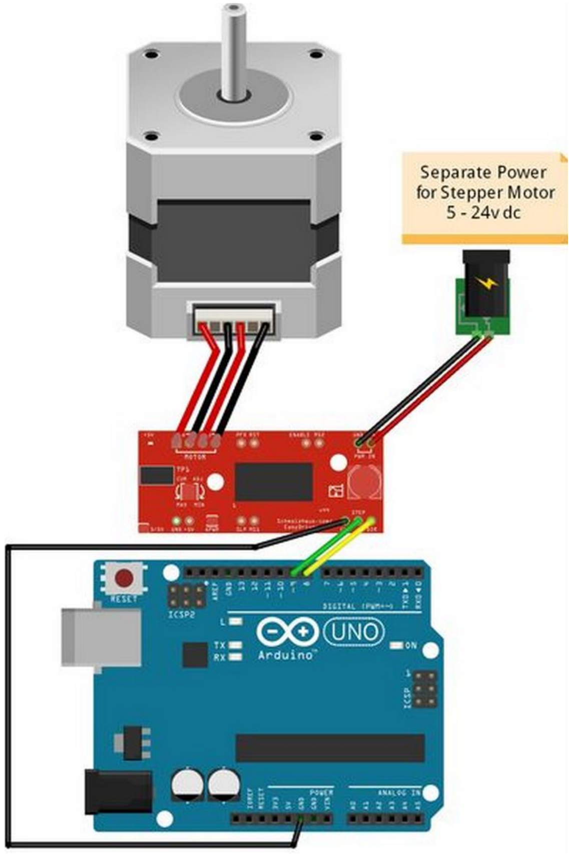
ANALOG/DIGITAL SENSORS

1k Linear Taper Rotary Potentiometer

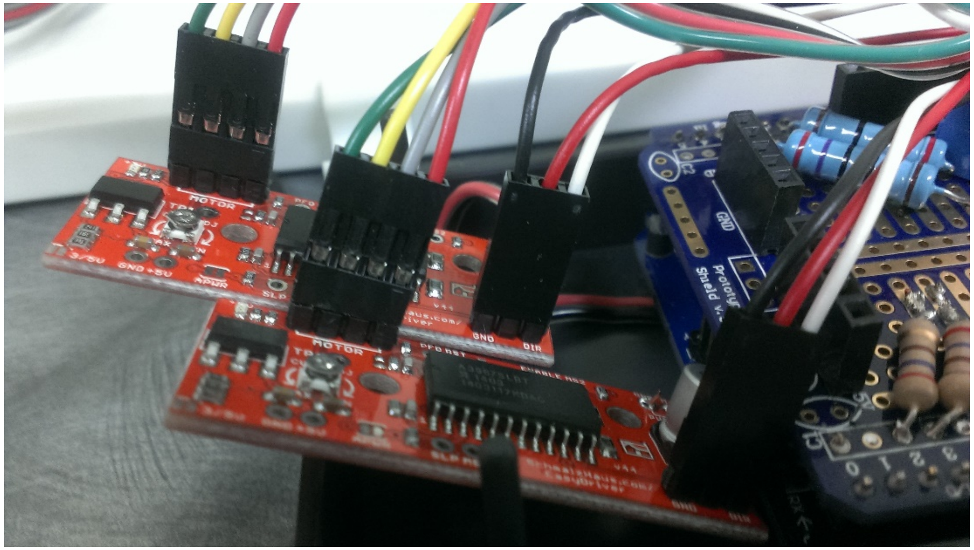
Limit Switch

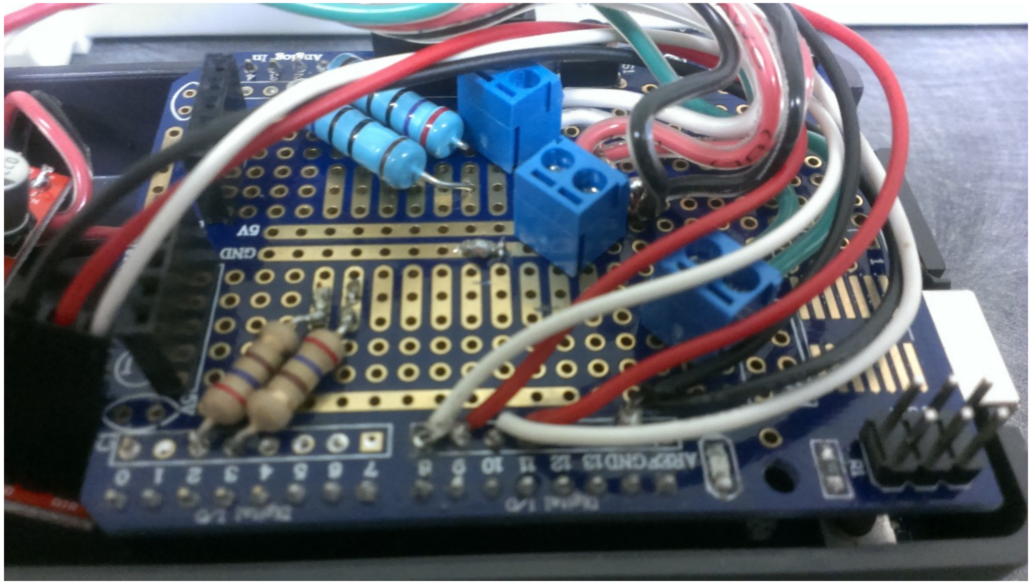


EASY DRIVER – NEMA 17 STEPPER MOTOR

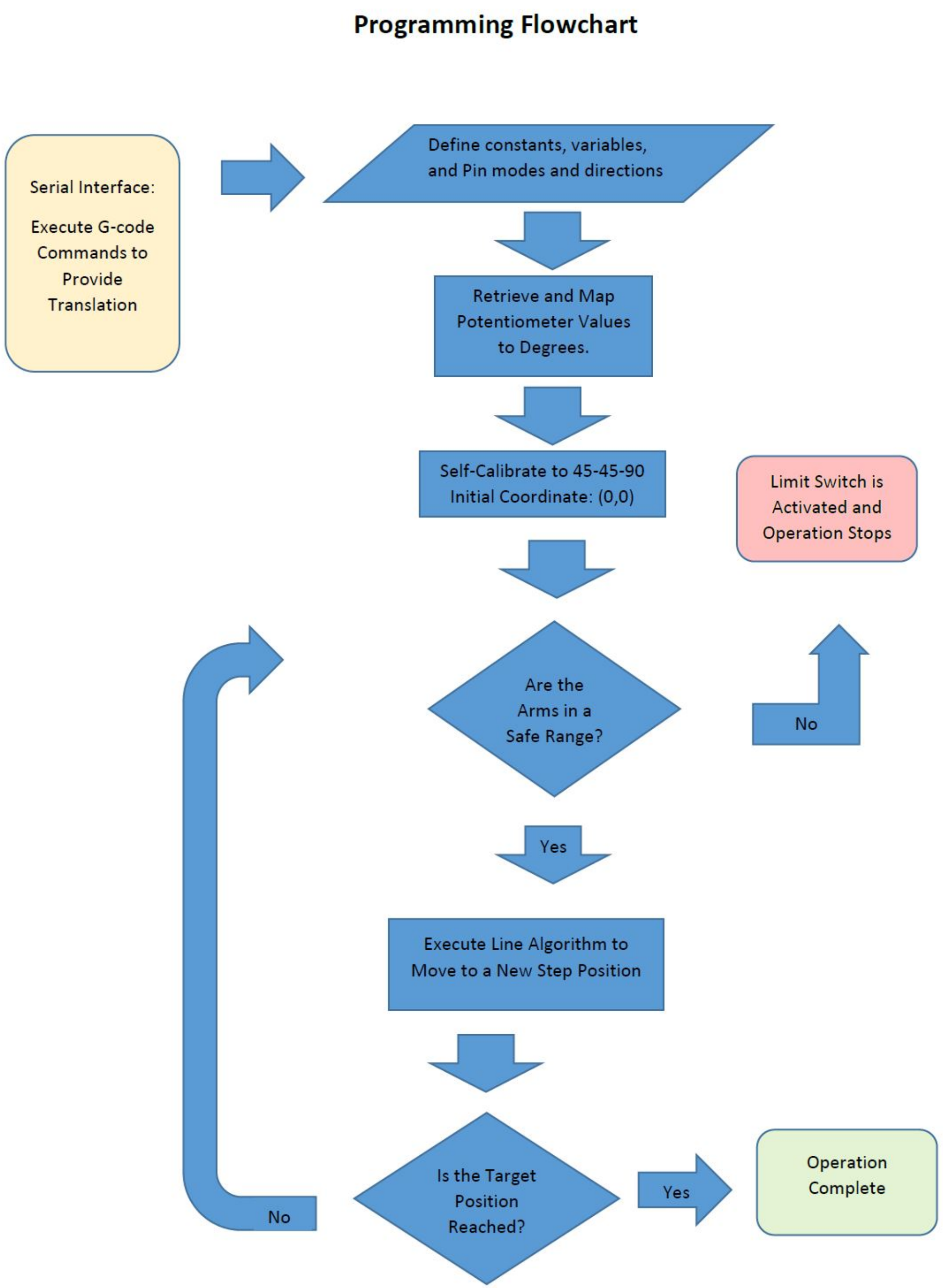


CIRCUITRY

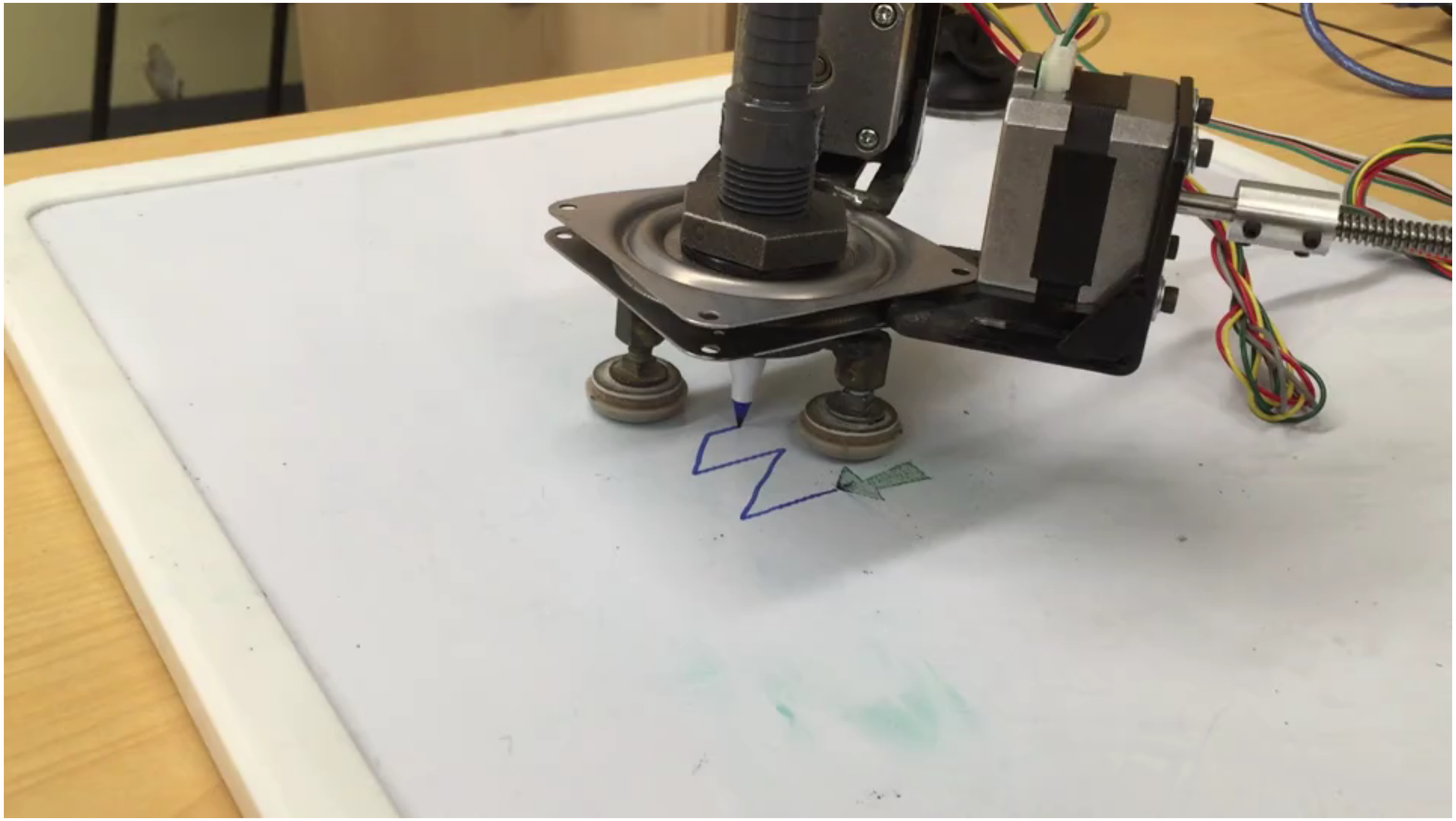




FLOWCHART


Programming Flowchart
Serial Interface:
Execute G-code Commands to Provide Translation
Define constants, variables, and Pin modes and directions
Retrieve and Map Potentiometer Values to Degrees.
Self-Calibrate to 45-45-90 Initial Coordinate: (0,0)
Limit Switch is Activated and Operation Stops
Are the Arms in a Safe Range?
No
Yes
Execute Line Algorithm to Move to a New Step Position
Is the Target Position Reached?
No
Yes
Operation Complete

DEMONSTRATION



PROS VS. CONS

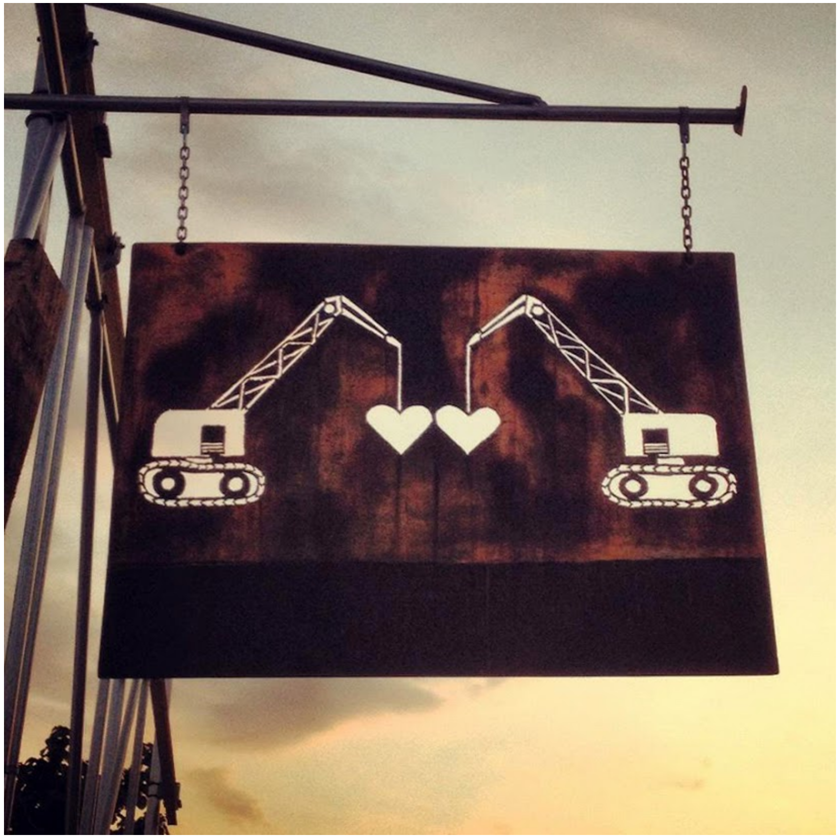
Pros

- Lower Cost than traditional CNC Plasma Cutter
- Triangulation provides the rigidity that would otherwise have to be provided by a robust frame
- By using suction cups to mount it, the device becomes portable and multi-purpose

Cons

- Resolution falls off outside of ideal work area
- Cannot constrain a device which provides force orthogonal to the plane it works on.
- Relies on gravity to hold it down

OTHER APPLICATIONS



QUESTIONS?