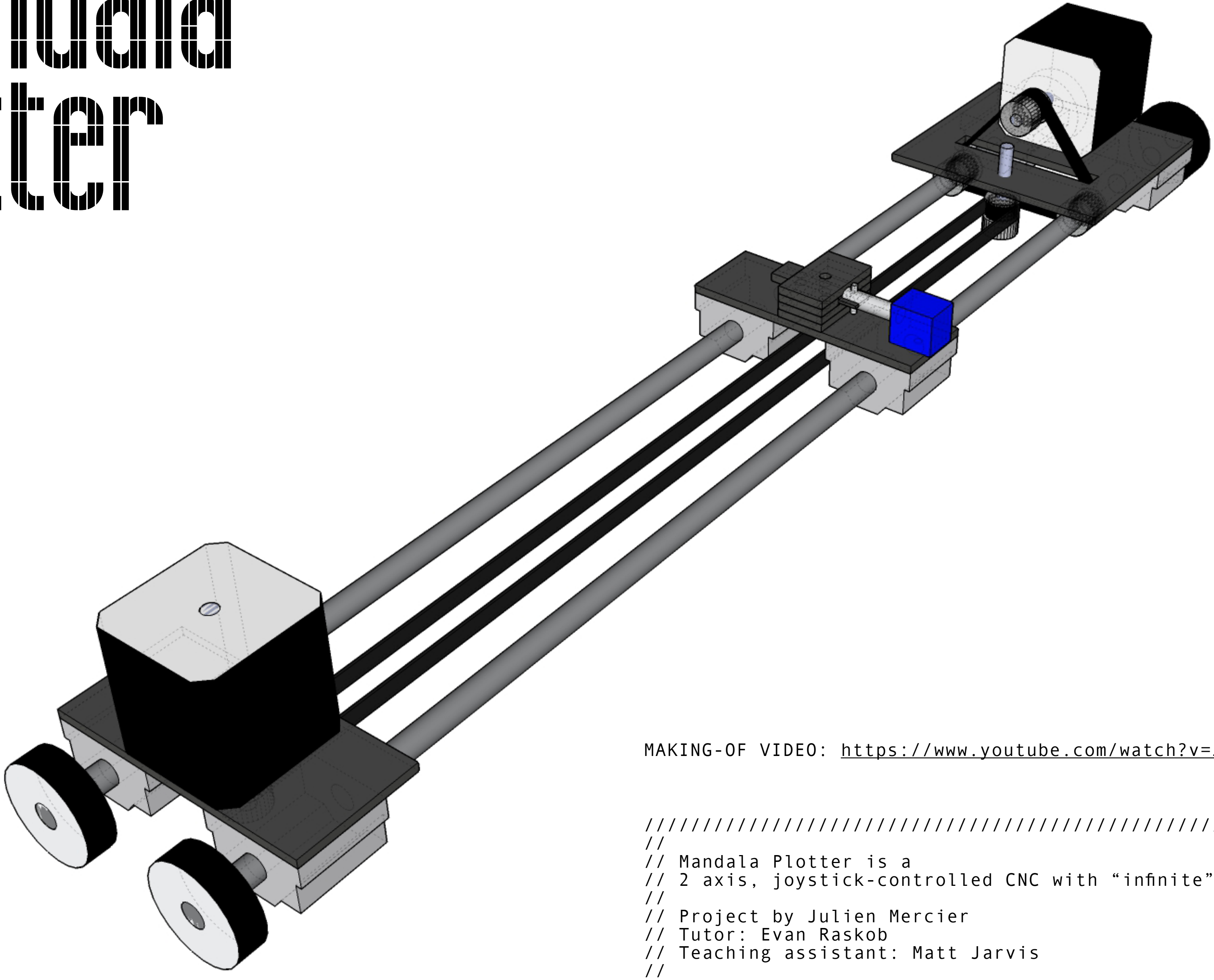
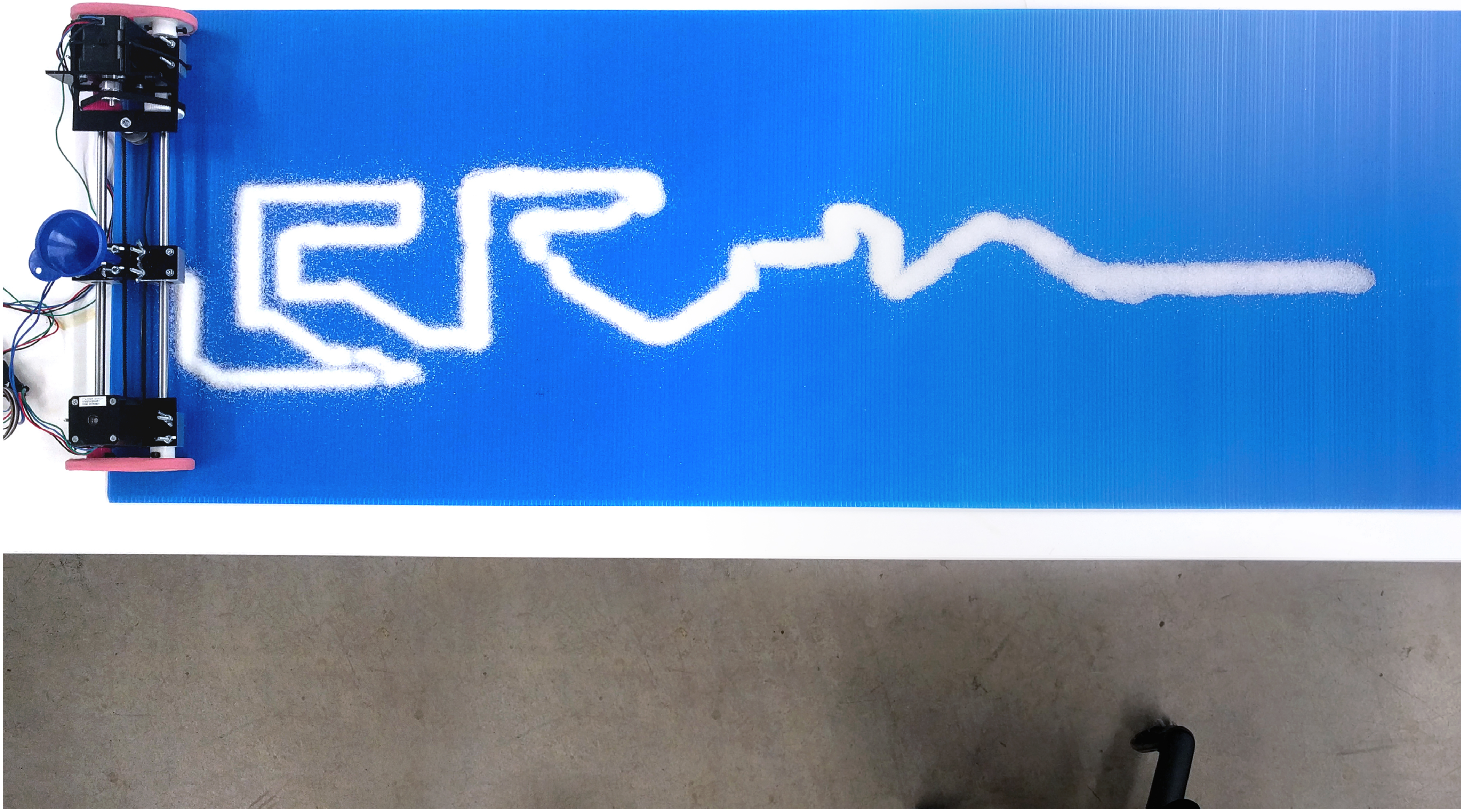
# Mandala Plotter



MAKING-OF VIDEO: https://www.youtube.com/watch?v=JPHRC_p_l5g

```
//////////////////////////////////////////////////////////////////////
//                                                                  //
// Mandala Plotter is a                                             //
// 2 axis, joystick-controlled CNC with “infinite” Y-axis           //
//                                                                  //
// Project by Julien Mercier                                        //
// Tutor: Evan Raskob                                               //
// Teaching assistant: Matt Jarvis                                  //
//                                                                  //
// Physical Computing/Computational Arts/Goldsmiths/2018-19         //
//                                                                  //
//////////////////////////////////////////////////////////////////////
```



# Foreword

This project is an attempt at building an ‘infinite Y-axis’, modular plotter that works with granular/powdery material, such as sand. It advantageously uses wheels instead of a second set of rails, allowing it to run until it hits the next wall.

Strategy, Habitat and Further development
While this current prototype is limited in size and reach, the aim is to develop it further into a fast-printing device for the making of large, ‘impermanent’ artworks on the floor, inspired by the spiritual and ritual art of mandala making.

My underlying strategy draws in the artistic tradition of “creating tools”, and as such is to be seen as a means, as opposed to an end in itself. My intention is to shorten the digital making toolchain between the artist and his outputs. My interest in CNC systems goes back many years, and their increasingly widespread use at all levels of artistic practice is by far not yet exhausted, from my own point of view.

My habitat signifies my current limits. Expanding my perspectives compares to expanding my habitat. The more room I can create, the more potential the outputs will have. I’ve long wanted to start building a modular system from scratch, that I could keep developing as my needs evolve. While this previously seemed like a bottomless chasm, I was able to make a substantial leap over this term, thanks to the sum of in-class tutoring and an important online community of makers, who share their experience and knowledge. The next technical steps will be to tame the GRBL library (possibly Marvin too), calibrate the steppers (steps, limits, keep track of position...), and use all the functions of the shield (limits, E-stop, cooler...) in order to develop into a ‘grown-up’, fully functioning CNC system.

The future artistic endeavours I’d like to pursue will discuss the utopian promises of new technologies as well as the way they induce new rituals that might substitute various spiritual traditions. Indeed, mandalas’ sacred nature lies in the time they take to realize, in their impermanence, and as such express the transient nature of life. In this context, trying to replace the human hand with a machine can be seen as sacrilege, but may also provide an opportunity for introspection. What place do we really want to give to technology? What are the limits we are not ready to exceed, for the time being? Techno-global capitalism does not spontaneously make room for such questions, and they all too often answers themselves at the users’ own expense.

## Intro

I have had a long-term interest for CNC systems. In general, I am a firm believer in the notion of creating my own tools as an artist, and get as much control as possible over the entire chain of events. I have always wanted to build my own that I could modify, adapt, and make modular. For this first attempt at making one, I set my objectives on a realistic scale. I simply wanted to get a good understanding of the mechanical, electronic and code structure behind such systems.

## Development

The early sketches I had were ones for a traditional structure, with X and Y axis along metallic bar and an overall frame to encase them. I built a quick prototype fig. 1 to evaluate the following steps such as positions of the steppers fig. 2, time pulleys and belt etc. This approach seemed good enough for my current expectation. But rather quickly, I started wanting to add a little of my own touch, to make it a little original. Observing this prototype, I figure the distance limitations on both axis was some kind of a problem and if there were any ways around it?

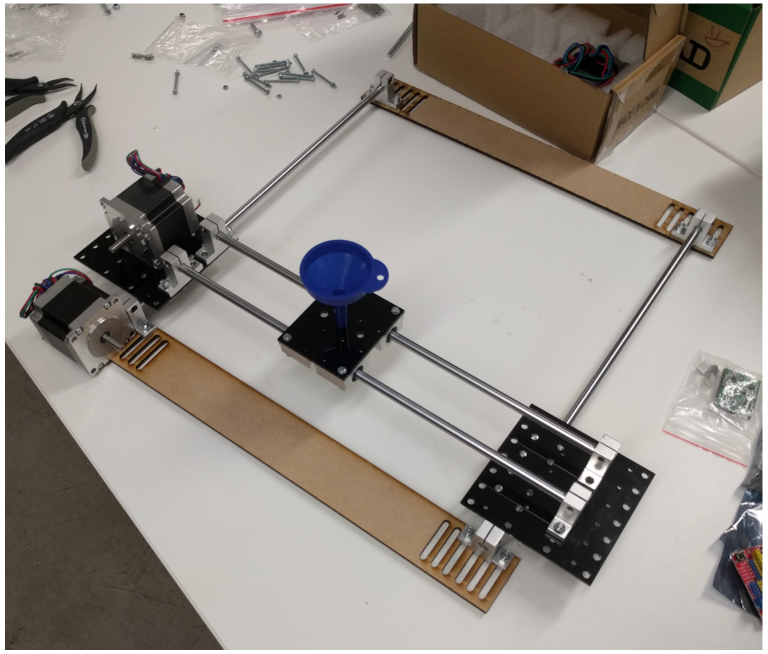
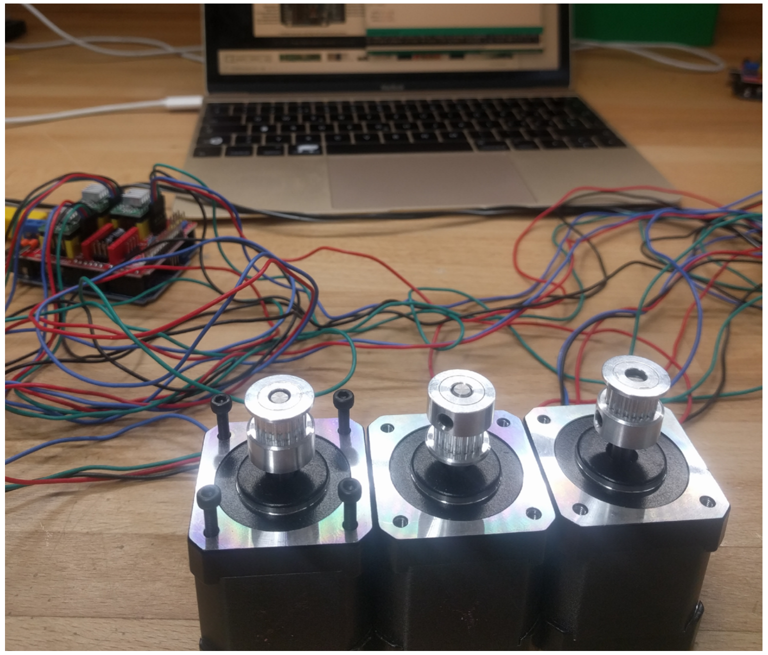
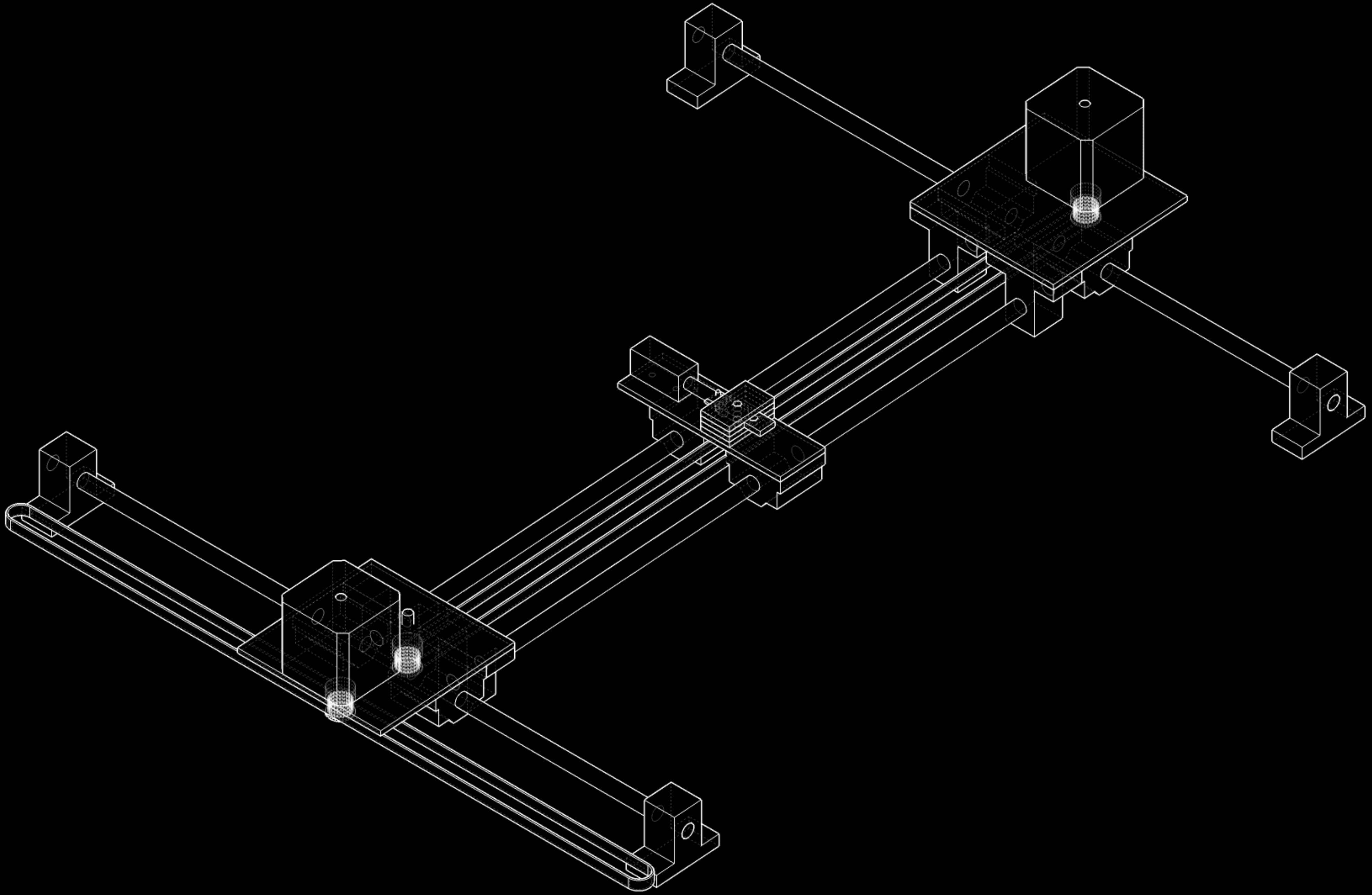


fig. 1 First structure prototype with static X & Y axis on rails



Initial design for a traditional CNC structure



I came up with the idea of replacing the Y axis rails by little wheels, like a remote controlled car. I could not replace both axis, because the wheels would then roll on the matter I would be depositting later. But replacing one axis only allowed to maintain the printing head above the surface to be "printed".

I was (still am) very unsure that this approach will give good printed results at the end of the road, but there were still enough good reason to try and make a traiditional cnc project a little more exciting. I could tell, and I would soon face it, would be adherence.

I sketched up fig. 3 the structure for this idea and started building the parts, starting with the Y axis module (on the left). The main idea being that I could use the same two metal bars for the X and Y axis, thus making the overall structure simplier and lighter.

From the 3D sketch, I retrieved the plates dimensions fig. 4 and added the necessary holes for assembling.

For the electronics (see further), I followed Evan and Matt advices and got an Arduino CNC shield from Protoneer (https://shop.protoneer.co.nz/). It comes with Pololu A4988 drivers that I needed to drive the Nema steppers I wanted to use. The wiring is well-explained on Protoneer's website and the forum is here to help with unresolved questions.

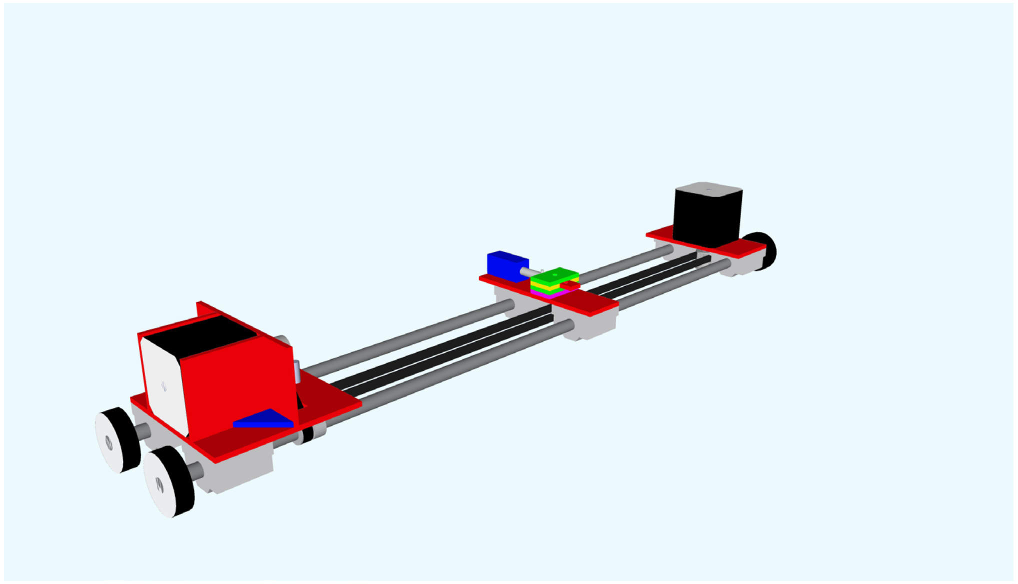


fig. 3 Design proposal on Google's SketchUp, with X and Y axis using the same bars.

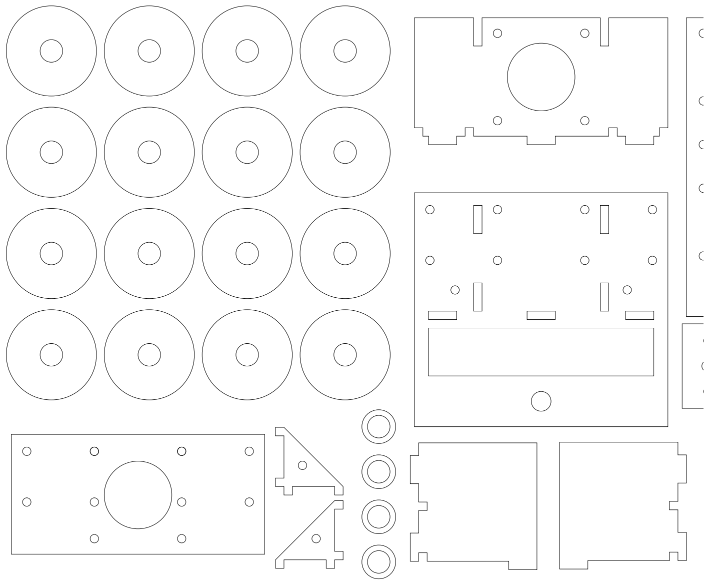
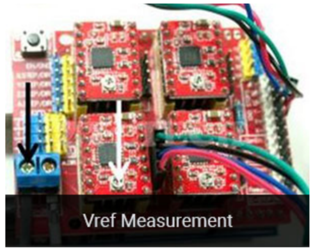
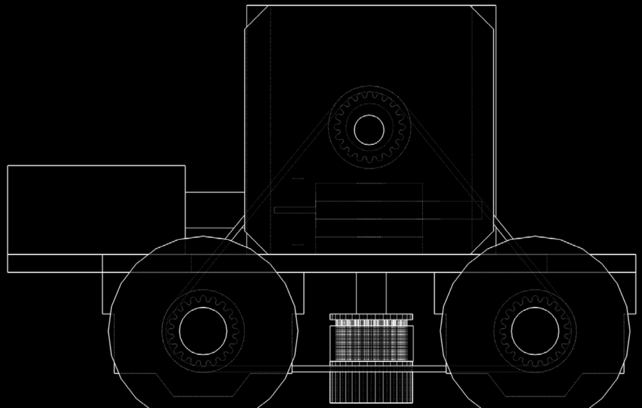
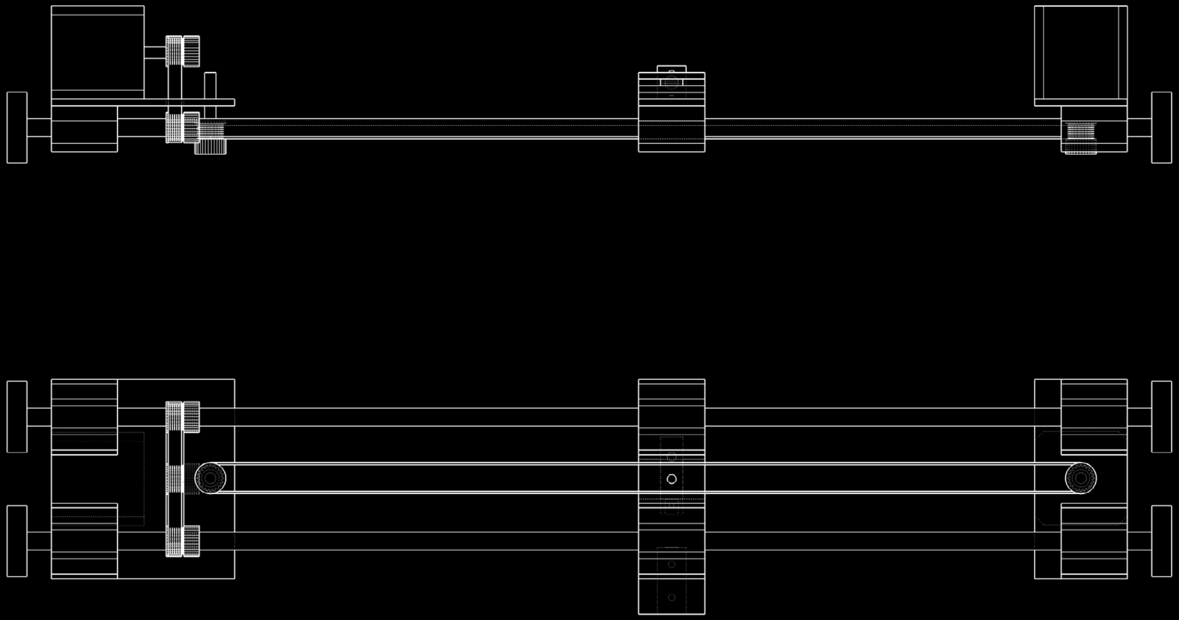


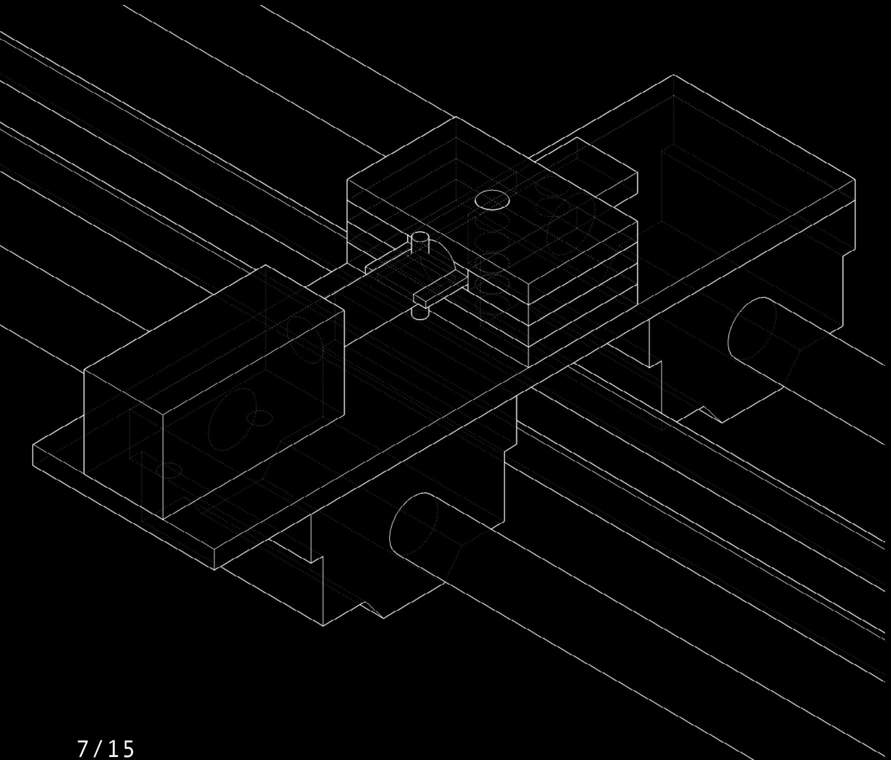
fig. 6 Y axis module parts for laser cutter production

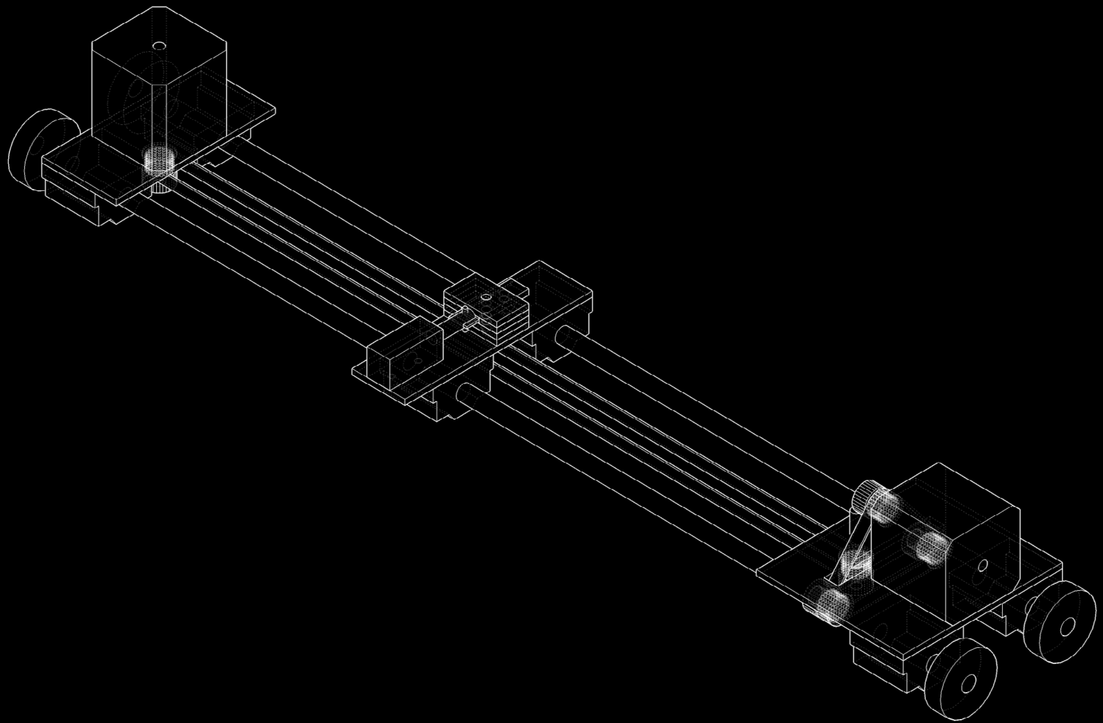


Design for a “remote-controlled” CNC printing device









After lasercutting the parts and tweaking back-and-forth, I also 3D printed time pulleys fig. 5 that I needed to connect. Some maths were needed to find a way to calculate the time pulleys' dimension, since I could not find any file available for my very specific dimensions.

fig. 5 3D printing time pulleys

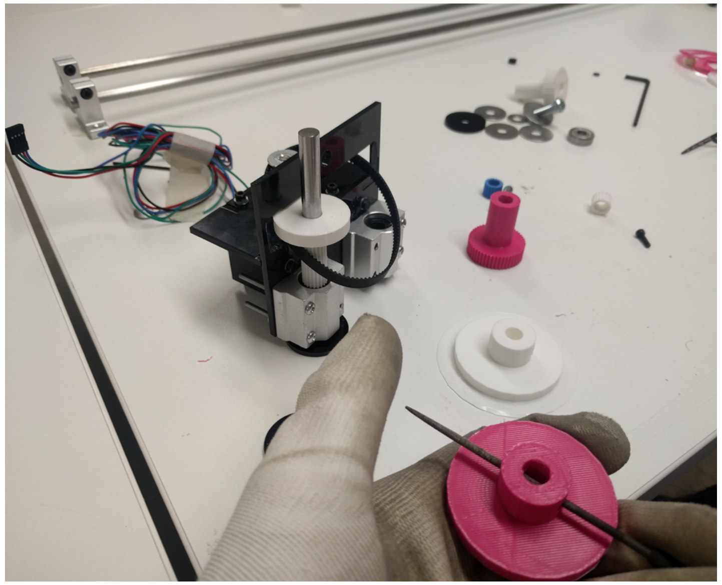
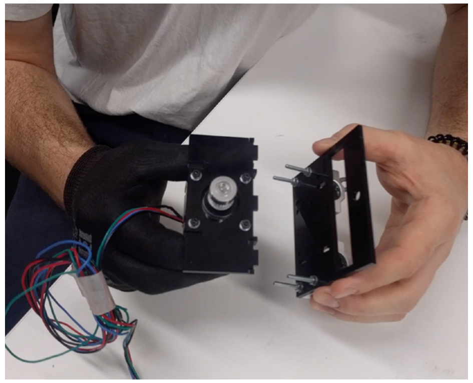
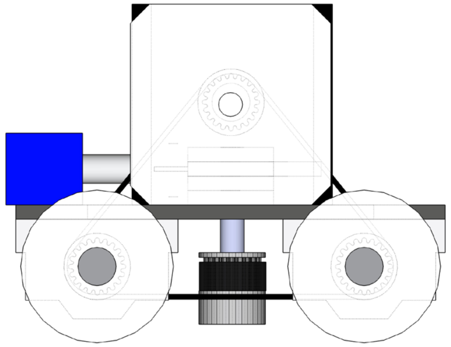


fig. 5 3D printed parts adjustement... and assembling process for Y axis module

The module was ready fig. 6 and I added the stepper. (I found smaller, Nema 17 stepper that would fit the current prototype better than the previous Nema 23.)





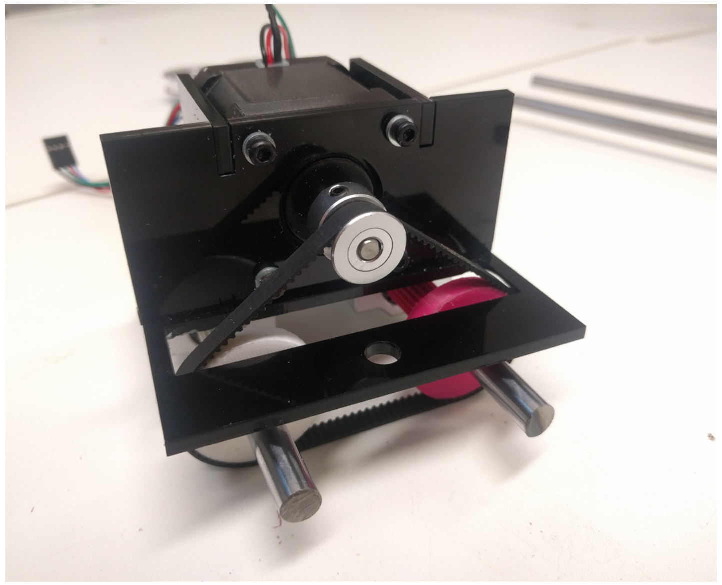


fig. 6 Y axis module with Nema 17 stepper

In comparison, the X module was a piece of cake and only needed to fit a Nema stepper pointing downwards for the upcoming belt that would drive the trolley. I decided to place a stepper on each side, in order to make it look more symmetrical and to distribute the weights more equally. However, this has an important drawback in having to wire both those sides. In a next version of this machine, I will try and place both the steppers on the same side, to limit the wire restraints.

For the trolley, I built on previous tests fig.7 that I made in class with solenoids. I decided that this “1st” printing head would deposit powdery material (sugar, flour, coffee, paster... whatnot), that would work with a simple funnel. I adapted the design to mount it on my trolley fig.8.

For the electronics fig.9, I followed instructions I found on [https-//www.bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/.jpg](https-//www.bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/.jpg)

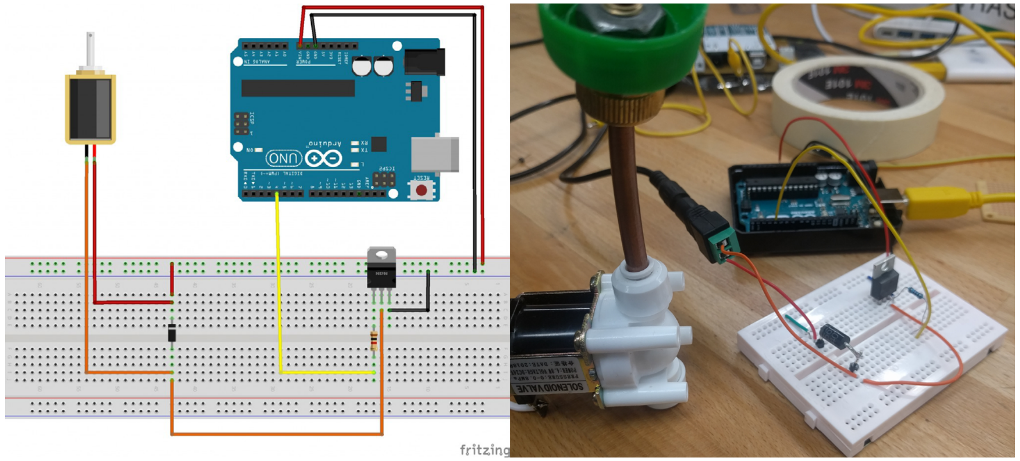


fig.9 Schematics of solenoid on Arduino, with external power (12V in this case).

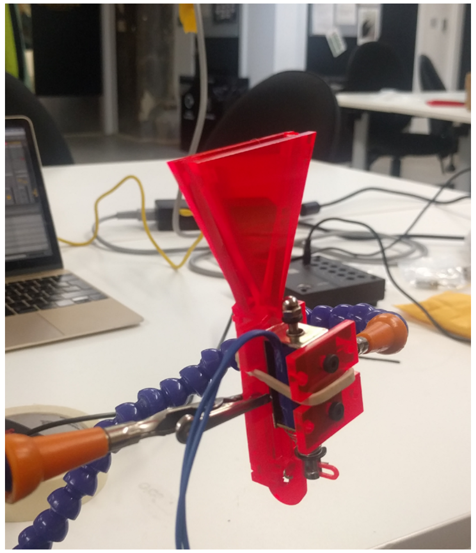


fig.7 Earlier prototyping with solenoids based “drop” modules

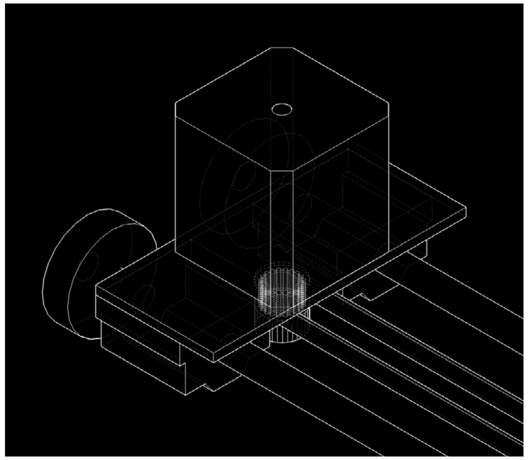


fig.6 bis. X axis module design

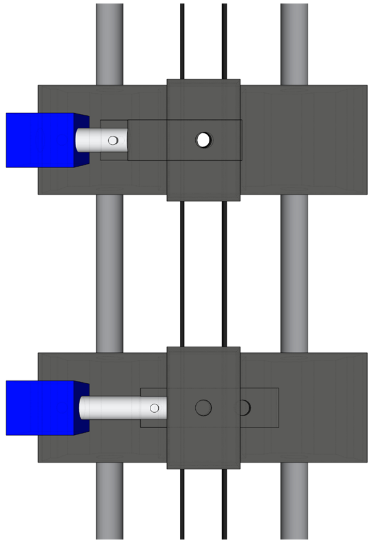


fig.8 Design for a “powder-dropper” module trolley

The finished module fig.10 worked surprisingly well, though the resolution at which it deposits materials (see tests, fig.11) needs further development. But for the time being, it did what I needed it to.

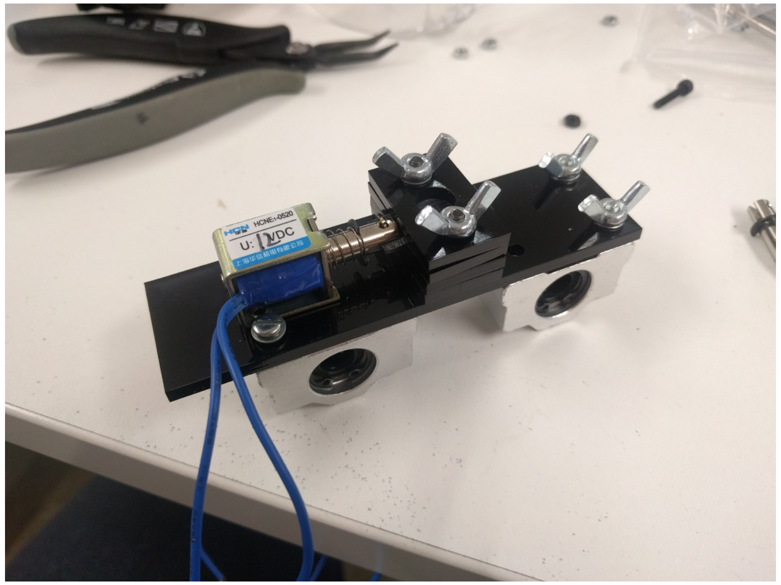
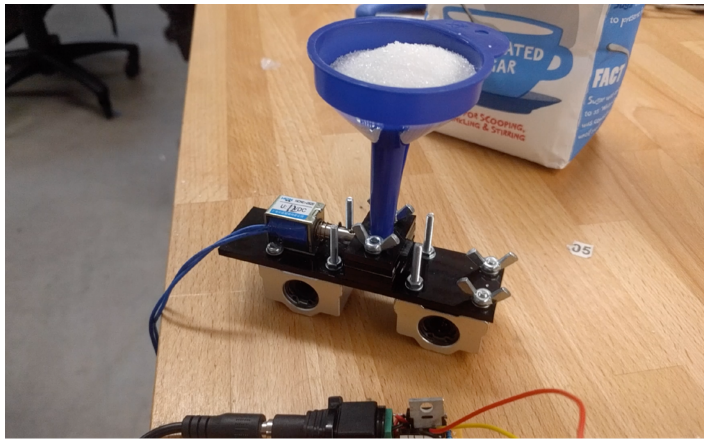
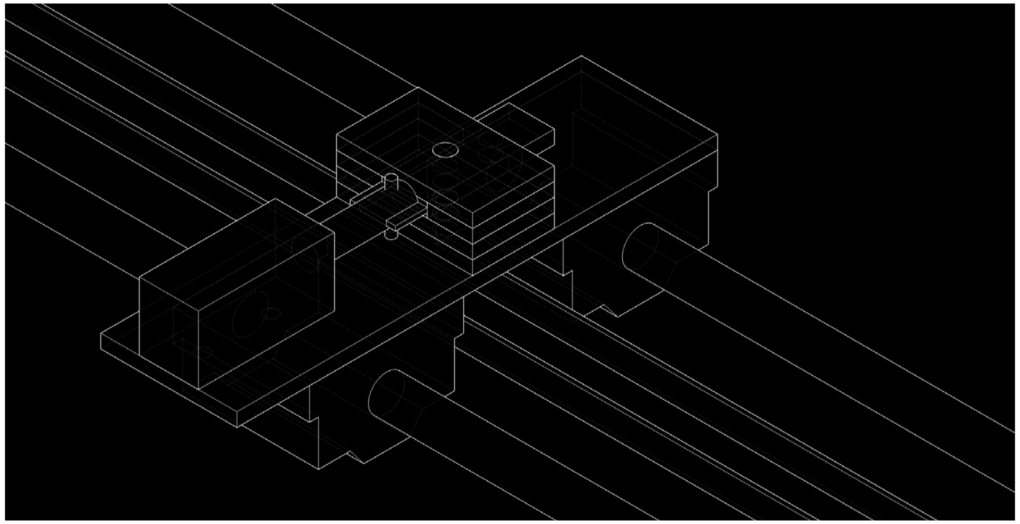


fig.10 Final "printing head" module...


... with funnel and sugar



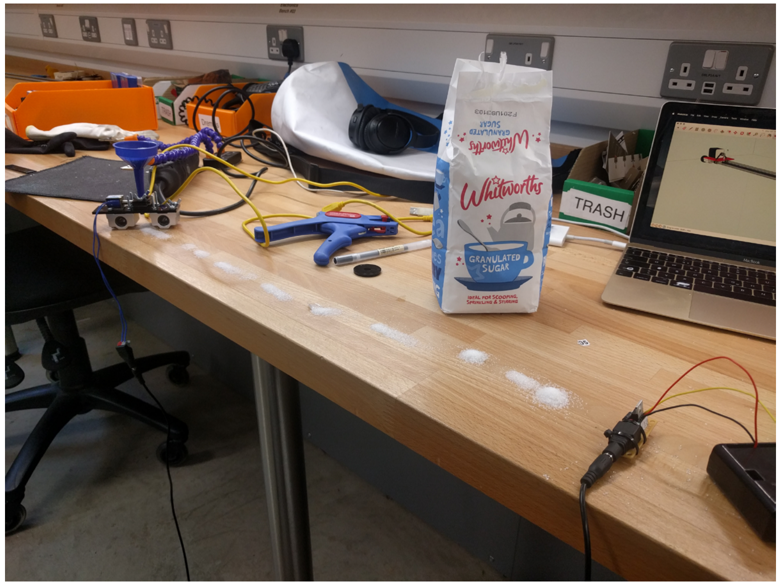
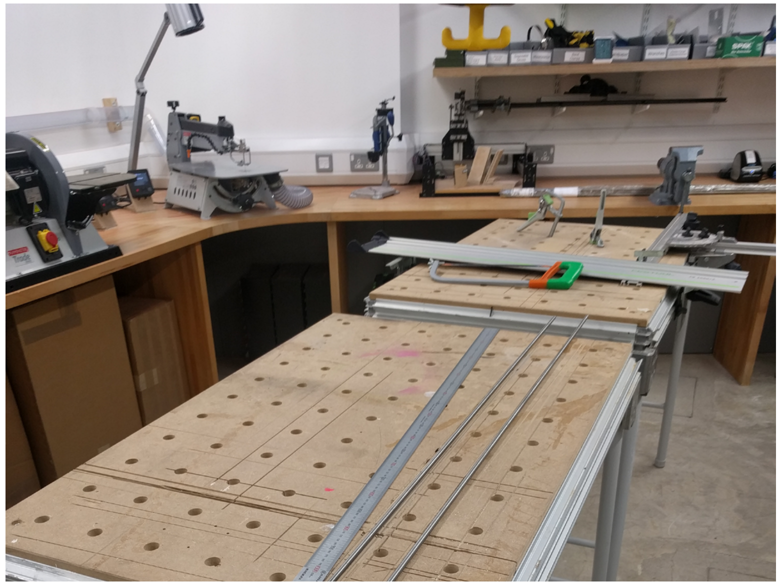


fig.11 "Printing tests with sugar"

Because the plan was always to make it a modular machine, I started playing with the idea of making a larger version of it right away, using longer axis [fig.12] in order to print on a wider area.

It somehow worked alright, after compromising in length. The initial 3m long bars were to flexible, but after cutting them in half, there resistance was about right. For the sake of simplicity, and to make the prototype fit the table and easier to handle, I decided to stick with my 40cm long axis for now.


[fig.12] Tests with longer axis (compromising down from 300 to 150 cm)

[fig.13] Assembling tests with long axis

## Programming

Now fully assembled and reasonably good-looking fig.14, I could focus on control and software for a bit. From my discussion with Evan (Raskob) and Matt (Jarvis), it made more sense—in the context of the class—to write my own code and have direct control over the machine's instruments, instead of using the pre-written GRBL library. The goal is still to use this library at a latter point, and be able to print accurate vector design, but this was a better way to learn more about control and building such devices.

So I decided to use an analog joystick, that would allow the user to drive the machine up and down, left and right, and activate the solenoid/printing head on click. For this, I followed the instructions on:
https://maker.pro/arduino/tutorial/how-to-control-servo-motors-with-an-arduino-and-joystick
and adapated the code (that was made for servos), for my steppers. I tried and assembled all my electronic parts (the arduino with the CNC shield, the homemade driver for the solenoid, and the joystick) into something that could fit in the hand fig.15.

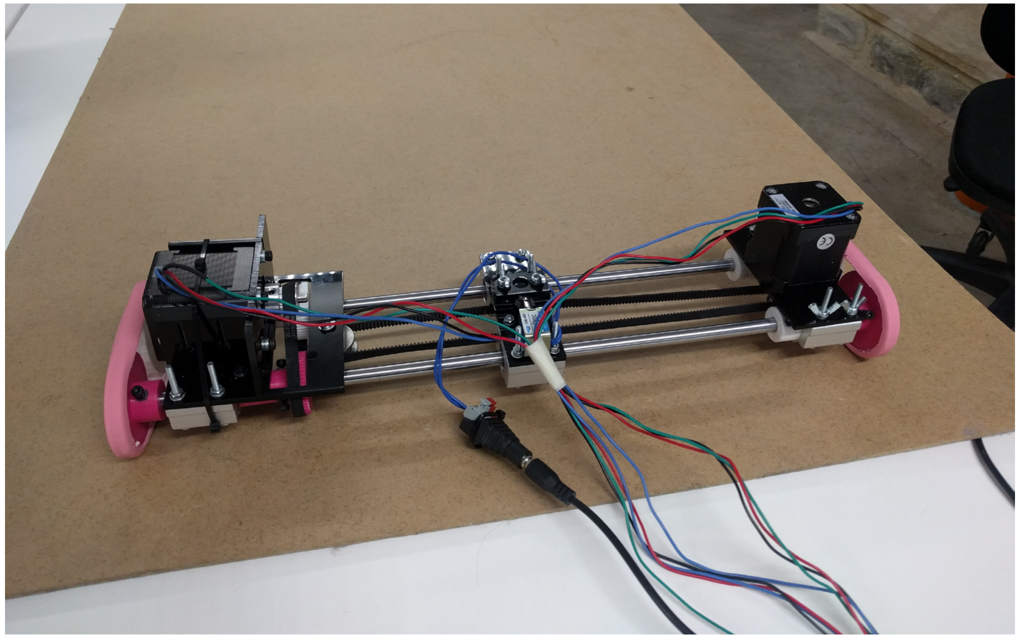


fig.14 Reasonably good-looking assembled prototype

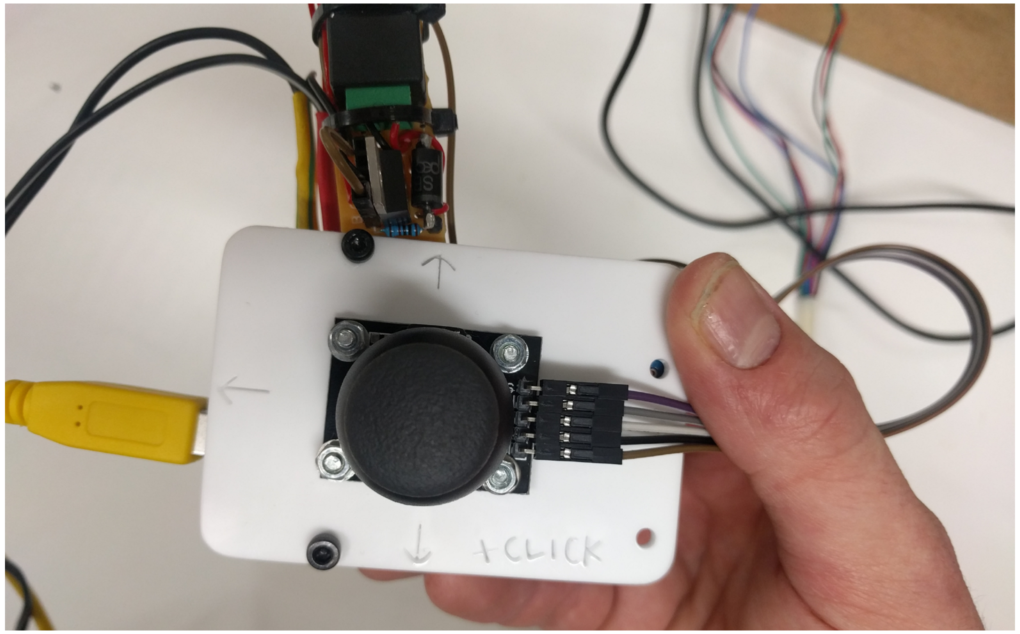


fig.15 The joystick, assembled with all electronic parts

# Code

Here's the final Arduino code I wrote and used. The tutorials I followed are linked on the next page.

```
//////////////////////////////////////////////////////////////////////
// Manual controlled CNC                                            //
// Physical Computing/Computational Arts/Goldsmiths/2018-19         //
// Project by Julien Mercier                                        //
// Tutor: Evan Raskob                                               //
//////////////////////////////////////////////////////////////////////

// include libraries to manage stepper acceleration.
#include <MultiStepper.h>
#include <AccelStepper.h>

// define joystick X & Y axis analog read
#define joyX A0
#define joyY A1

// define joystick switch input
const int SW_pin = 13;

// defin solenoid output
int solenoidPin = 12;

// steppers X & Y
#define STEPPER_ENABLE_PIN 8
#define MOTOR_X_STEP_PIN 2
#define MOTOR_X_DIR_PIN 5
#define MOTOR_Y_STEP_PIN 3
#define MOTOR_Y_DIR_PIN 6
AccelStepper motor_X(1, MOTOR_X_STEP_PIN, MOTOR_X_DIR_PIN);
AccelStepper motor_Y(1, MOTOR_Y_STEP_PIN, MOTOR_Y_DIR_PIN);

void setup() {
  //  Serial.begin(9600);
  // joystick analog read, X & Y
  pinMode(SW_pin, INPUT);
  digitalWrite(SW_pin, HIGH); // ??
  Serial.begin(115200);
  //Sets the solenoid's pin as an output
  pinMode(solenoidPin, OUTPUT);
  motor_X.setEnablePin(STEPPER_ENABLE_PIN);
  motor_X.setPinsInverted(false, false, true);
  motor_X.setAcceleration(200);
  motor_X.setMaxSpeed(200);
  motor_X.setSpeed(200);
  motor_X.enableOutputs();
  motor_Y.setEnablePin(STEPPER_ENABLE_PIN);
  motor_Y.setPinsInverted(false, false, true);
  motor_Y.setAcceleration(200);
  motor_Y.setMaxSpeed(200);
  motor_Y.setSpeed(200);
  motor_Y.enableOutputs();
}
void loop()
{
  int xValue = analogRead(joyX);
```

```
  int yValue = analogRead(joyY);
  //  print the values
  //  Serial.print("X=");
  //  Serial.print(xValue);
  //  Serial.print("\t");
  //  Serial.print("Y=");
  //  Serial.println(yValue);
  if (digitalRead(SW_pin) == LOW)digitalWrite(solenoidPin, HIGH);
  if (digitalRead(SW_pin) == HIGH)digitalWrite(solenoidPin, LOW);

  motor_X.run();
  motor_Y.run();
  if (xValue < 500)motor_X.move(-10);
  if (xValue > 550)motor_X.move(10);
  if (yValue < 500)motor_Y.move(-10);
  if (yValue > 550)motor_Y.move(10);


}
```

## References

– I've used the Arduino CNC shield, built and sold by Protoneer: https://blog.protoneer.co.nz/arduino-cnc-shield/

– I've used the "Accelstepper and Multistepper" library by Mike Mc Cauley, available here: http://www.airspayce.com/mikem/arduino/AccelStepper/ and here: https://github.com/vberkaltun/MultiStepper

– I've followed and adapted this tutorial hosted on brainy-bits.com in order to add analog joystick control: https://www.brainy-bits.com/arduino-joystick-tutorial/

– I found additional help and information on how to link the joystick read to the steppers (adapting servos to steppers) from this page hosted on maker.pro: https://maker.pro/arduino/tutorial/how-to-control-servo-motors-with-an-arduino-and-joystick

– I've followed this tutorial from bc-robotics and built the circuit, in order to control and power the solenoid I use in my printing shaft. I built and soldered the matching circuit: https://www.bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/

## Testing

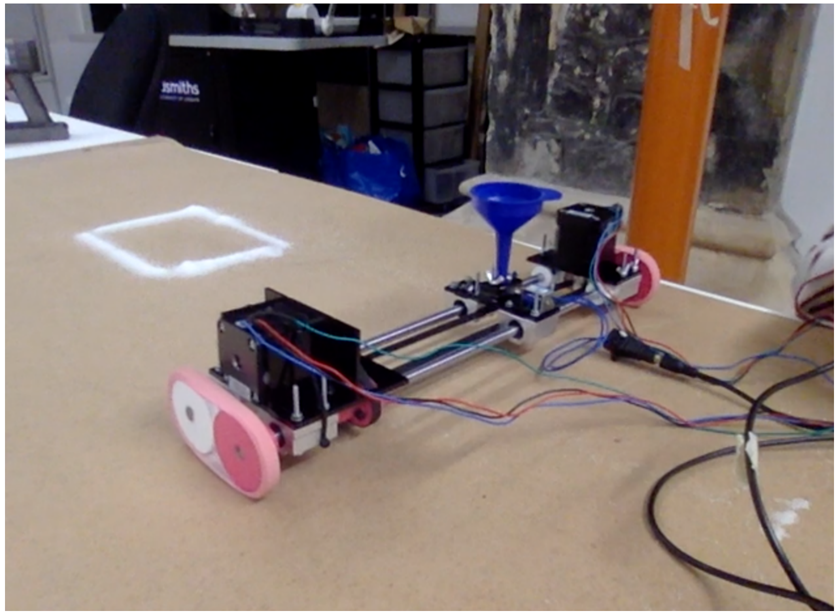


fig.16 Attempt at drawing a square

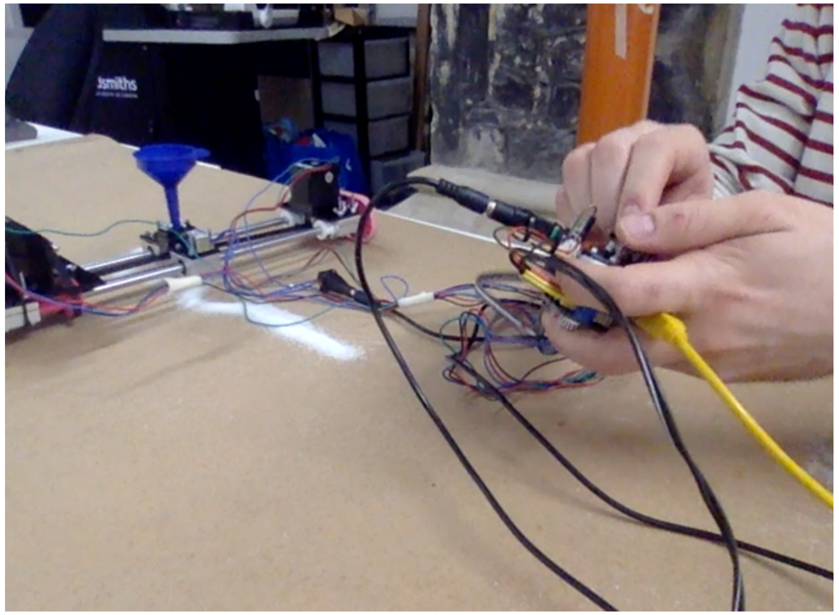


fig.16 Attempt at drawing a wavy line

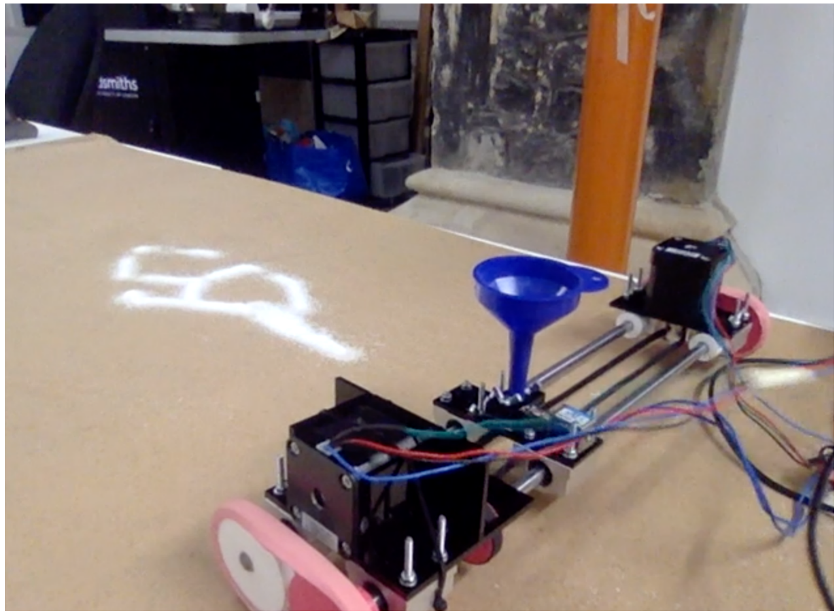


fig.16 Attempt at drawing ???