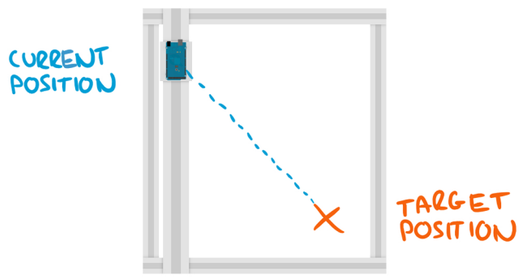
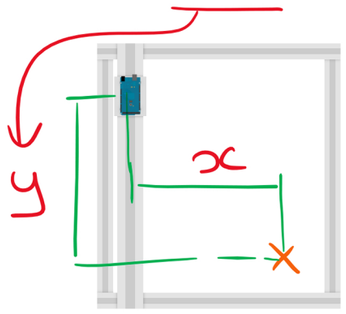
gantry movement

logic overview

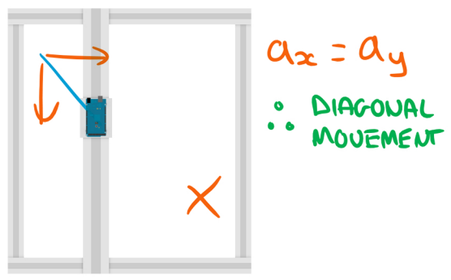
1. Input requested x, y position



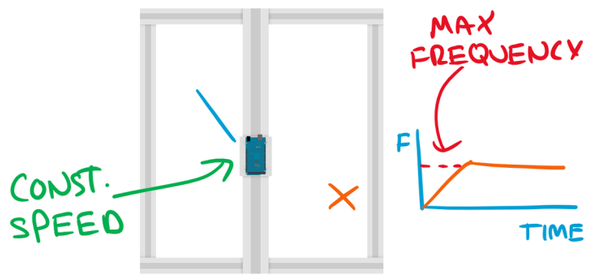
2. Determine shortest dimension



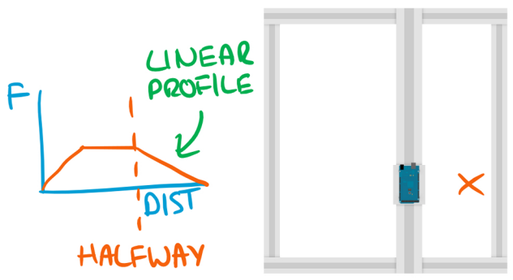
3. Accelerate both axes



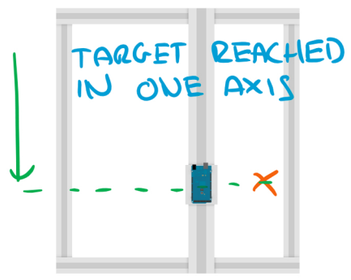
4. If max speed reached, stop ramping



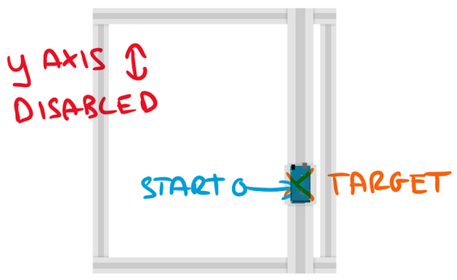
5. If halfway to target, proportionally ramp down



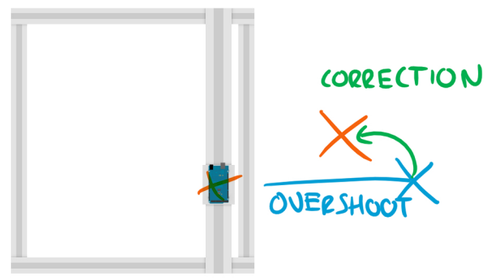
6. Stop once target is reached



7. Repeat for longer dimension



8. Make minor corrections if required

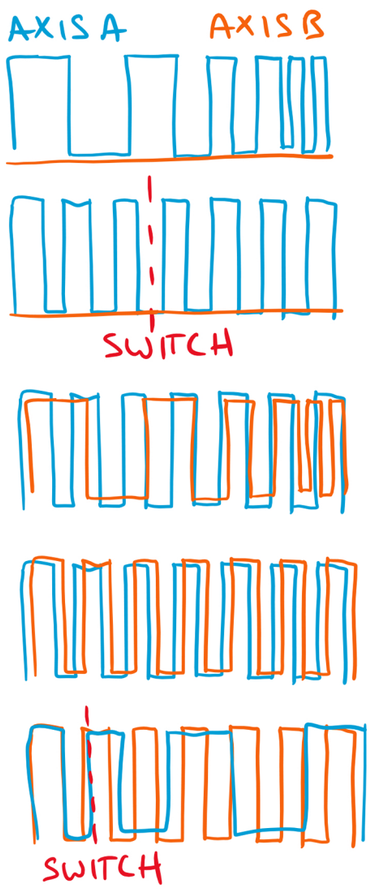


electronics

The gantry stepper motors require a pulse train to move. This is provided by the Arduino 'playTone' function.
This function can only be used on one pin at a time... but there are two axis, so two stepper motors.

why?

original idea



1. Stepper axis A ramps up using playTone pin.

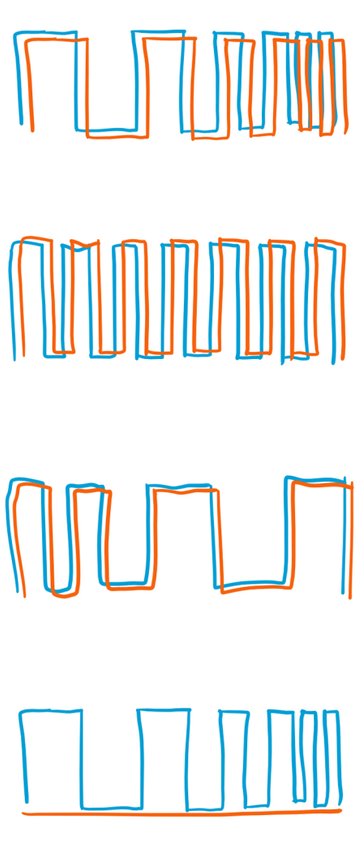
2. When axis A is at full speed, input to motor switches to a constant pulse generator; a 555 astable oscilator circuit.

3. This allows axis B to be ramped up with playTone pin.

4. Both axis run at full speed using the 555 oscilator, until they approach their target.

5. Control switches back to playTone pin when ramping down.

final idea



1. Both axes ramp up simultaneously using playTone pin.

2. Continue ramping until maximum speed reached, or over halfway to closest target.

3. Ramp down until shortest axis reaches it's target, then disable this axis only.

4. Repeat this movement for longer axis, until it reaches it's target.

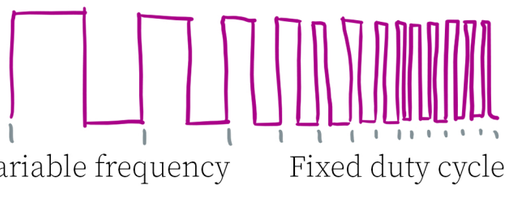
Pulse train waveform

Square wave function

PWM

Fixed frequency Variable duty cycle

playTone



Variable frequency Fixed duty cycle

Designed to play notes from buzzers!

positional feedback

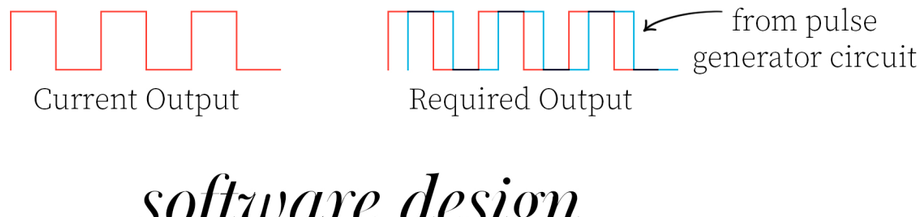
BACKGROUND
No. of pulses sent to stepper motors is proportional to the distance traveled.

PROBLEM A
No built in way to count sent pulses. Estimations can be inaccurate.

SOLUTION A
Use the Arduino to read the pulses, and keep track through software.

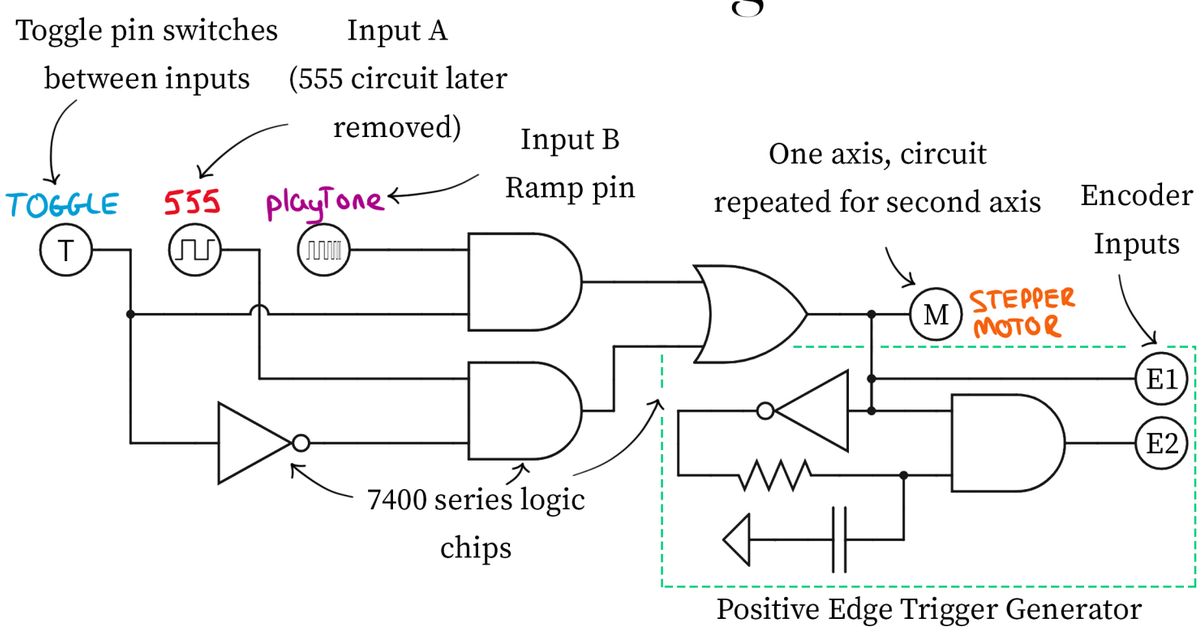
PROBLEM B
No built in way to count single steps, but there is a rotary encoder library...

SOLUTION B
Turn the single pulse train into something recognisable as a rotary encoder, and use the library.



Current Output Required Output

circuit design



software design

```
function = goToPos([x, y])

enable both axes

while less than halfway to closest target axis
    update current position variables
    output pulses to axes

    if frequency less than max frequency
        increase frequency

while more than halfway to closest target axis
    ...
    reduce frequency proportionally to distance left

%% Repeat for remaining distance on long axis only

if target surpassed
    perform minor corrections
```

ACCELERATION PHASE

DECELERATION PHASE

CORRECTION PHASE