XY Plotter Design Document

1. Abstract

The plotter designed for the Inter-Domain Task utilizes an Cartesian configuration, where each arm consists of rails supporting a rack system. The Y-axis arm moves the X-axis arm, enabling accurate bidirectional movement of the pen across the workspace. A paper clip holds the pen in place on a servo motor mounted on the X-axis arm, providing basic functionality for controlled movement. This straightforward design enables the plotter to perform the necessary operations effectively.

2. Game Plan

The plotter will be built in a Cartesian setup, using a rack and pinion system for both the X and Y axes. The Y-axis arm will support and move the X-axis arm, allowing controlled movement in both directions. A ball caster attached to the X-axis arm will provide stability as it moves along the Y-axis, reducing drag. A servo motor, mounted on the X-axis arm, will hold the pen in a paper clip and control its contact with the paper. Each motor will connect to a microcontroller through a motor driver, allowing us to control and calibrate both axes and the pen lift. Testing will include calibrating steps per unit for each axis and adjusting the pen height for consistent contact. After calibration, the plotter will be ready to receive design files and execute G-code for accurate drawing tasks.

3. Design and Mechanism

3.1 Mechanical

3.1.1 Components

Components	No. of items
Rack and Pinion	2
Drawer Rails	2
Aluminum Channel	2
Motor brackets	2
Pen Mechanism	1
Ball castor	1
Wooden Base plate	1

Table 3.1.1 Mechanical Components

Rack and Pinion: The Rack and Pinion are used to move the pen through the x and y axis, Each arm has a single Rack and Pinion system.

Drawer Rails: The drawer rails provide smooth, guided linear movement for the plotter's X and Y axes. The Rack is mounted on these rails to facilitate movement in the desired axis, with the help of the motor.

Aluminium Channel: This is the base of each arm on top of which the rails are mounted, They have been reinforced with wood to increase their structural integrity.

Motor Brackets: The motor brackets securely position the stepper motors for optimal alignment with the rack gears. This ensures smooth and precise movement of each arm along its axis while enhancing stability and minimizing vibrations that could affect performance.

Pen Mechanism: The pen mechanism consists of a U-shaped aluminum block that securely holds the pen and connects it to the servo motor. This setup allows the servo to raise and lower the pen smoothly, ensuring consistent contact with the paper.

Ball Castor: The ball caster provides foundational support for the X-axis by stabilizing it from beneath. It bears the combined weight of the X-axis arm, stepper motor, servo motor, and pen mechanism, ensuring level positioning.

Wooden Base Plate: The wooden base plate serves as the primary structural foundation of the plotter, providing a stable and level surface for mounting all components. It supports the entire assembly, including the X and Y axes, motors, and electronic components, minimizing vibrations.

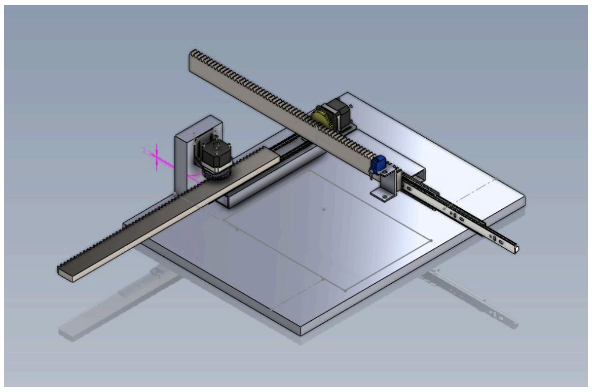


Fig 3.1.1. Isometric view of Plotter

3.1.2 Dimensions

Component	Dimensions
X Aluminium Channel	50x25mm
Y Aluminium Channel	50x25x260(L)mm
X Rack	410x15(L)mm
Y Rack	410x15(L)mm
Pinion	400x600(D)mm
Wooden Base Plate	435x550mm
Pen Holder	75x50x25mm
Motor Bracket	50x50x2m
BOT Weight	

Table 3.1.2. Mechanical Dimensions

3.2 Electronic

3.2 Electronics Components

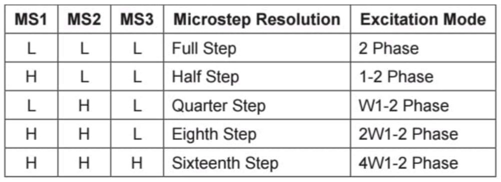
The electronics design for the plotter integrates a range of components configured to deliver synchronized control and high precision in plotting tasks. This section provides a detailed overview of the electronic architecture, specifying the function of each component and the technical rationale for their selection. Refer to table 3.1.1 for the components.

Components	No. of items
Arduino Nano	1
A4988 Motor Drivers	2
Nema 17 Stepper Motor	2
Sg90 Servo Motor	1
LM7805 Voltage Regulator	1
Capacitors 100µF	2
4 Way Dip Switch	2
Screw Terminal	1

Table 3.2.1. Electronic Components

Arduino Nano: The Arduino Nano, powered by an ATmega328P microcontroller, controls the X and Y axis movements and pen-lifting servo with precision, low power consumption, and a compact design, making it ideal for this plotter.

A4988 Motor Drivers: The A4988 is a microstepping driver for bipolar stepper motors, supporting full to 1/16-step modes with up to 35V and 2A output. It includes current limiting and thermal shutdown for motor and driver protection, ensuring reliability and longevity.



MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

Fig 3.2.1. Microstepping datasheet

Nema 17 Stepper Motor: The NEMA17 stepper motor is a compact and reliable motor with a 1.8° step angle, operating at 1.2 to 2.5 amps per phase and delivering 40–45 N·cm of holding torque. Its 42x42mm Size makes it ideal for precise movements in space-constrained applications like plotter.

SG90 Servo Motor: The SG90 servo motor is a lightweight, compact servo that operates at 4.8 to 6 volts and delivers up to 2.5 kg·cm of torque. Its 0 to 180-degree rotation range makes it ideal for precise pen-lift control in plotters.

LM7805 Voltage Regulator: The LM7805 is a reliable linear voltage regulator that outputs a steady 5V from input voltages of 7V to 35V, supplying up to 1A. Its thermal shutdown and current-limiting features ensure safe operation, protecting the circuit from overload.

Capacitors 100µF: Capacitors filter electrical noise that can impact motor and microcontroller performance. A 100µF capacitor smooths power supply fluctuations, ensuring cleaner voltage and improved system reliability.

4 Way Dip Switch: The 4-way DIP switch, utilized as a 3-way switch, connects to the stepper motor driver to control microstepping resolution. Toggling the switches allows for quick adjustments to the motor's step size, facilitating easy calibration of the plotter's precision without reprogramming.

Screw Terminal: The screw terminal provides a secure connection for powering components in the plotter. It allows for easy assembly and disassembly, ensuring reliable electrical connections for the motors and power supply.

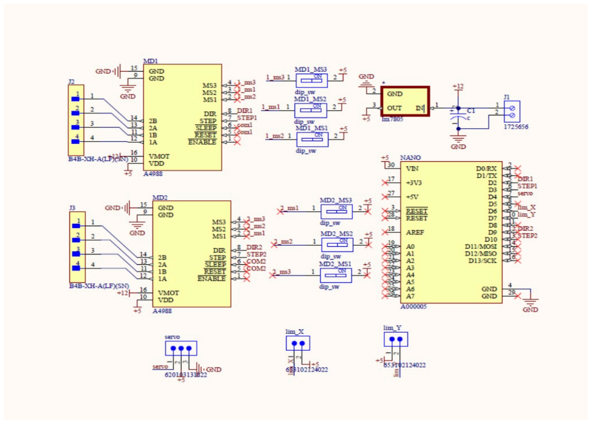


Fig 3.2.2. Electronic Schematics

3.1.2 Working

The plotter's power and control setup begins with the Linear Voltage Regulator (LVR) receiving 12V input, which it converts to a stable 5V output. This 5V line powers the Arduino Nano, limit switches, logic levels for motor drivers, and the SG90 servo motor. The 12V line directly powers the stepper motors. Two 100µF capacitors are added to stabilize the voltage and minimize noise in the circuit.

Each motor driver connects to a 4-way DIP switch (configured as a 3-way switch) to set their respective microstepping modes, allowing precise control of motor movement. The Arduino Nano's digital pins are allocated as follows: D3 connects to the step pin and D2 to the direction pin for one motor, while

D10 connects to the step pin and D9 to the direction pin for the other motor. The step pins control the stepper motor speed, while the direction pins determine rotation direction.

The commands are sent to the Arduino, which then interprets and controls the timing and extent of the stepper motor movements. Additionally, the Arduino determines when the servo motor engages the pen with the paper, coordinating accurate drawing operations for the plotter.

3.1.3 Calculations

To set the current limit for the stepper motor, we use the formula:

$$Current\ Limit = \frac{Vref}{8 \times Resistance}$$

For a target current limit of 1A, we substitute the values as follows:

$$1A = \frac{Vref}{8 \times 0.1}$$

Solving for Vref, we get:

$$Vref = 1 \times 0.8 = 0.8V$$

Thus, we set Vref to 0.8V. To ensure this setting, we will measure the potential difference across the potentiometer on the stepper motor driver and the ground pin, while the driver is powered with a 5V supply. Adjusting the potentiometer to achieve 0.8V across these points will set the current limit to 1A, protecting the motor and driver from overcurrent.

4. Coding

The software workflow for the plotter involves creating or editing an image and processing them to Arduino, which processes the commands that dictate the movements required for drawing. It translates these into specific control signals for the A4988 drivers, which in turn regulate the stepper motors' movements based on the programmed instructions. This section provides a detailed breakdown of each software component used and how the control functions are implemented.

4.1 Software and Integration

This Arduino code allows manual control of two stepper motors (X and Y axes) and a servo motor (for pen lifting) using the keyboard. The controls are mapped to the W, A, S, and D keys for movement.

4.2 Function Code Snippets

4.2.1 Header Files

These header files include the necessary libraries for controlling the stepper motors (`AccelStepper`) and the servo motor (`Servo`).

```
#include <AccelStepper.h>
#include <Servo.h>
```

4.2.2 Global Declaration

These `#define` statements assign pin numbers to the control signals for the X and Y stepper motors and the servo motor. This makes the code easier to read and modify.

```
#define X_STEP_PIN 3
#define X_DIR_PIN 2
#define Y_STEP_PIN 10
#define Y_DIR_PIN 9
#define SERVO_PIN 5
```

4.2.3 setup() function

The `setup()` function initializes serial communication and attaches the servo to its designated pin. The pen is set to the "up" position at startup.

```
void setup() {
    Serial.begin(115200);
    penServo.attach(SERVO_PIN);
  penUp();    stepperX.setMaxSpeed(500);
  stepperX.setAcceleration(500);
  stepperY.setMaxSpeed(1000);
  stepperY.setAcceleration(500);
  pinMode(X_LIMIT_PIN, INPUT_PULLUP);
pinMode(Y_LIMIT_PIN, INPUT_PULLUP);
  autoHome();
```

4.2.4 loop() and move function

The `loop()` function runs continuously. It checks if there is any data available on the serial port and reads it if present. Also, the below block translates the keyboard commands into movements. It moves the stepper motors by 25 steps in the specified direction based on the key pressed.

```
void loop() {
 if (Serial.available() > 0) {  // Check i
data is available from the serial port
       char command = Serial.read();
// Read the incoming character

// Map keyboard commands to movements
 if (command == 'w') {
       moveStepper(stepperY, 25);
// Move forward (Y axis)
 } else if (command == 'a') {
       moveStepper(stepperX, -25);
// Move left (X axis)
 } else if (command == 's') {
       moveStepper(stepperY, -25);
// Move backward (Y axis)
 } else if (command == 'd') {
       moveStepper(stepperX, 25);
// Move right (X axis)
 }}
```

4.2.6 `moveStepper()` function

The `moveStepper` function controls the movement of the stepper motors. It takes in a reference to the stepper motor and the number of steps to move, executing the movement and waiting for it to complete.

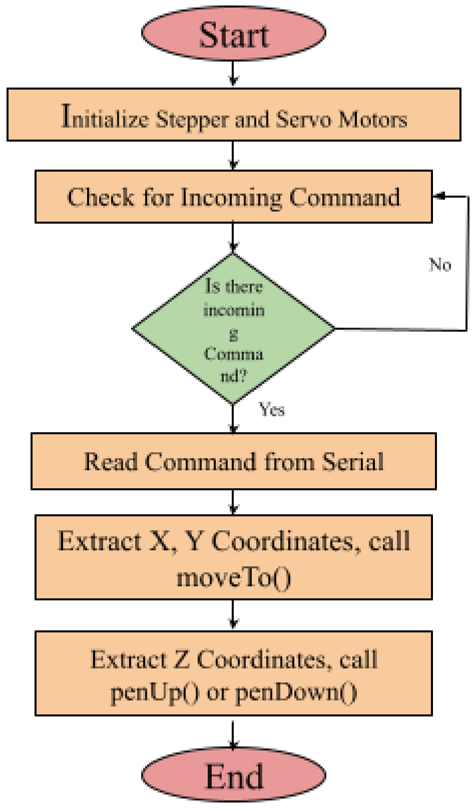
```
void moveStepper(AccelStepper
&stepper, int steps) {
 stepper.move(steps);  // Move the
stepper by the specified number of
steps
 while (stepper.distanceToGo() != 0) {
// Wait until the movement is complete
       stepper.run();  // Keep running
the stepper until it reaches the
target position
 }
}
```

4.2.6 `penUp()` and `penDown()` function

The **Pen Up and Pen Down functions** are are used to control the servo motor by using `penServo.write()` function, i.e. Lift the pen up to the pen up position which is 90 and lower the pen down to the pen down position which is 0 respectively.

```
void penUp() {
  penServo.write(penUpPosition);
}
void penDown() {
  penServo.write(penDownPosition);
}
```

4.3 Flow Chart


Start
Initialize Stepper and Servo Motors
Check for Incoming Command
Is there incomin g Comma nd?
No
Yes
Read Command from Serial
Extract X, Y Coordinates, call moveTo()
Extract Z Coordinates, call penUp() or penDown()
End

Algorithm of the Arduino Code

5. Conclusion

This documentation outlines the design, assembly, and calibration of a precision plotter using an Arduino Nano, A4988 stepper motor drivers, and supporting components like limit switches, capacitors, and a servo-driven pen mechanism. By following these guidelines, the plotter can be configured to execute G-code instructions accurately, achieving smooth movement across both X and Y axes. The integration of limit switches allows for consistent calibration and a reliable home position, while careful setting of the current calibration, this plotter is ready to perform precise and repeatable drawing tasks, offering a reliable solution for automated plotting needs.

6. References

https://www.friendlywire.com/projects/ne555-servo-safe/SG90-datasheet.pdf

https://pages.pbclinear.com/rs/909-BFY-775/images/Data-Sheet-Stepper-Motor-Support.pdf

https://www.pololu.com/file/0j450/a4988_dmos_microstepping_driver_with_translator.pdf

https://www.sparkfun.com/datasheets/Components/LM7805.pdf

https://howtomechatronics.com/projects/diy-pen-plotter-with-automatic-tool-changer-cnc-drawing-machine/

https://www.instructables.com/X-Y-Plotter/