

Voiceplot: Voice-To Handwriting Using CNC Plotter

Mrs. Dipika Nimje¹, Ashik Dhete², Viranchi Bhagat³, Hardik Didmuthe⁴, Piyush Tiwari⁵, Prajwal Warhadkar⁶

¹Assistant Professor, Artificial Intelligence Engineering, Priyadarshini Bhagwati College of Engineering
²³⁴⁵⁶ B.Tech Students, Artificial Intelligence Engineering, Priyadarshini Bhagwati College of Engineering

Abstract — This document introduces VoicePlot AI, a platform that transforms spoken audio into tangible handwritten output via a CNC pen plotter. The pipeline connects four unique stages: capturing audio through sounddevice, transcribing speech to text locally with OpenAI Whisper, generating scalable vector graphics (SVG) using a personalized single-stroke font renderer, and controlling hardware motors via the AxiDraw Python API that operates a custom CNC pen plotter. A web server based on Flask integrates all phases and provides a browser dashboard for live monitoring and management. The system functions completely offline, requiring no internet connection during operation, which makes it appropriate for use in areas with limited connectivity. Experimental testing shows effective end-to-end transcription and physical representation of spoken phrases, achieving a word error rate below 10% for clear speech. The pen-lift servo's hardware characterization showed a cold-start delay, resulting in the initial written word being missed; the underlying issue and a software solution are outlined thoroughly. Possible uses include accessibility tools for individuals with motor disabilities, automated generation of certificates, and AI-based art installations.

Keywords: VoicePlot AI, CNC Pen Plotter, OpenAI Whisper, Speech Recognition, SVG Path Creation, AxiDraw API, Flask, Live Transcription, Assistive Technology.

1. INTRODUCTION

The integration of AI with physical manufacturing has created a new class of systems that were challenging to construct only a few years back. Speech recognition, once dependent on cloud infrastructure and substantial computing resources, can now operate efficiently on a standard laptop in real time using models like OpenAI Whisper. In the same way, CNC pen plotter machines that maneuver a pen on paper through computer control have become easier for hobbyists and researchers to obtain because of budget-friendly, DIY designs that utilize conventional stepper motors and microcontrollers.

VoicePlot AI combines these two advancements into a unified process. The system addresses a straightforward yet significant issue: presently, there is no affordable, offline, open-source solution that can capture an individual's spoken words and print them on paper in one automated action. Conventional CNC plotters necessitate that users create designs with vector software, save the file, and transmit it to the machine by hand. Speech-to-text software only shows text on a display. VoicePlot AI connects

these two processes directly, removing any manual steps in the middle.

From a practical standpoint, the most direct use is accessibility: an individual with restricted hand movement, whether from injury, illness, or disability, can vocalize a message that a machine converts into a physical, handwritten format. In addition to accessibility, the system is suitable for customized greeting cards, automated generation of certificates, and engaging art installations.

The technical contributions of this study encompass three main areas. Initially, we explain the development of a comprehensive system that combines four intricate subsystems: audio recording, neural speech transcription, SVG path generation, and CNC motor control into one Flask application that can be operated via a web browser. Next, we pinpoint and explain a particular hardware flaw in our custom CNC plotter: a servo cold-start delay that results in the first word being overlooked, accompanied by a software solution. Third, we present the creation of a tailored embedded single-stroke font displayed as SVG path elements, which was required since the AxiDraw API's standalone mode lacks support for SVG text elements

2. LITERATURE REVIEW

2.1 CNC Pen Plotters and Writing Devices

Jegan et al. [1] created an XY plotter using a standard Cartesian system and demonstrated sub-millimetre accuracy in detailed line drawings. Hasan et al. [2] took this idea further by developing a three-axis machine with an Arduino and CNC Shield V3, the same hardware used in our custom design. They found that adding a third axis increases mechanical complexity without offering benefits for two-dimensional writing, which directly impacted the design of our pen-lift servo. Pinheiro et al. [3] added Bluetooth connectivity to a CNC plotter, while Madekar et al. [6] built a specialized mini-CNC unit for precise PCB drawing in a compact format. The AxiDraw plotter, manufactured by Evil Mad Scientist Labs [11], is the most commonly referenced pen plotter in academic research. Our custom plotter uses the same two-stepper and one-servo configuration and is compatible with the AxiDraw Python API. The main hardware difference is the use of an SG90 micro-servo for pen lifting, which has a slower mechanical response time compared to the servo in genuine AxiDraw units. This difference is the reason for the cold-start problem discussed in Section 4.

2.2 Automatic Speech Recognition

Speech recognition has a long history in computing. Early systems like CMU Sphinx used acoustic models and pronunciation dictionaries. Graves et al. [12] demonstrated that recurrent neural networks could learn to transcribe speech end-to-end without explicit phoneme modelling, establishing the modern paradigm. Radford et al. [13] published Whisper in 2022 a transformer-based model trained on 680,000 hours of weakly supervised web audio. The base model achieves a word error rate under 5% on standard English benchmarks while running in approximately one second per five seconds of audio on a laptop CPU. Its offline operation is a critical feature for our deployment requirements.

2.3 Voice-Controlled Writing and Accessibility Systems

Aditi et al. [9] built an automated writing and drawing machine with speech recognition as a control interface, demonstrating the feasibility of voice-driven CNC operation. However, their system relied on cloud-based speech recognition APIs, introducing network dependency and latency that our system eliminates. Ranjan et al. [8] developed a low-cost 2D plotter and noted that lightweight frames cause vibration at higher speeds an observation confirmed in our own calibration testing, addressed by reducing the default pen-down speed. To our knowledge, no prior published work combines a local, offline speech recognition model with a real-time SVG generation pipeline and direct CNC hardware output in a single integrated system. VoicePlot AI addresses all three gaps simultaneously.

3. SYSTEM ARCHITECTURE

3.1 Overview

The system is structured as a Flask web application running on a standard Windows laptop. Five components work in sequence to take microphone input and produce pen marks on paper: the audio recorder, the Whisper transcription engine, the SVG generator, the AxiDraw plotter interface, and the browser-based control dashboard.

When the user clicks the 'Tap to Speak' button in the browser, a POST request is sent to the Flask backend, which spawns a background thread that runs the full pipeline. The browser polls a /api/state endpoint every second to update the progress display. This polling architecture avoids WebSockets and keeps the frontend implementation simple.

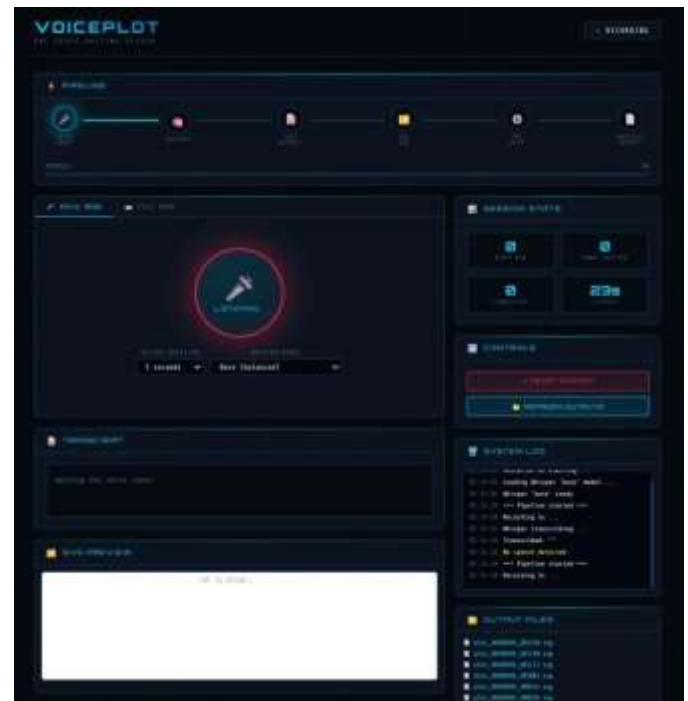


Fig. 1: VoicePlot AI Browser Dashboard: Complete control interface

3.2 Sound Recording

Audio is recorded with the sound device library at 16 kHz in mono, aligning with Whisper's required input format. Capturing runs for a set five-second period activated by the browser button press. The unprocessed NumPy float32 array goes straight to Whisper without creating a WAV file first, removing a disk I/O process and lowering overall latency.

3.3 Whisper Transcribing

The Whisper base model is initialized once during server startup in a background daemon thread, ensuring it's prepared when the user first presses the button. Transcription is compelled into English through the language parameter to eliminate the burden of language detection. The fp16 flag is configured as False since the majority of laptop CPUs are not efficient in handling 16-bit floating point calculations.

The shared state dictionary contains the transcribed text string that the browser checks, allowing the user to view what was heard prior to the plotter starting to move. This serves as a gentle error verification if Whisper misunderstood something important, the user can hit Reset before the device begins drawing.



Fig. 2: Transcript Panel : Real-time display of Whisper Output

3.4 SVG Generation and the Single-Stroke Font

This subsystem was the most technically complex aspect of the project. The original belief that SVG text elements were usable turned out to be wrong: the AxiDraw Python API in standalone plot mode exclusively handles SVG path and polyline elements, quietly disregarding text elements.

The answer involved creating an embedded single-stroke font: a Python dictionary that associates each character with a list of polyline strokes, where each stroke consists of a list of (x, y) tuples in normalized font coordinates. The complete printable ASCII character set consists of 26 letters, 10 numbers, and 18 punctuation marks. Every character was crafted by hand to appear clear when created as a single continuous stroke of the pen with few lifts.

The SVG generator calculates word-wrap limits according to page width, margin configurations, and character cell width. It processes each character by traversing its strokes, applies a positional offset and scaling factor, and generates an SVG path element. A warmup path with a 0.01 mm invisible stroke at coordinate (1, 1) is added as the initial element in every created SVG, requiring a full pen-down/pen-up servo cycle before the plotter arrives at the first character.



Fig. 3: SVG Preview Panel: Rendered single-stroke output for plotted text

3.5 Hardware Interface: AxiDraw API and Custom CNC Plotter

The AxiDraw Python library (pyaxidraw) communicates with the custom CNC plotter over USB serial in plot mode. Key tunable parameters include `pen_pos_down` (servo angle when pen is lowered), `speed_pendown` (motor speed during drawing), and `pen_delay_down` (milliseconds to wait after lowering pen before moving). All options must be set before calling `plot_setup()`; options set after that call are ignored.

Our custom plotter uses an SG90 servo clone that responds more slowly than genuine AxiDraw hardware, so `pen_delay_down` was increased from the default 200 ms to 400 ms. Pen pressure (`pen_pos_down`) was set to 30, which produces clean marks without tearing paper on the lighter carriage.

3.6 Flask Server and Browser Dashboard

The backend is a single-file Flask application exposing eight REST endpoints: `/api/start` (voice pipeline), `/api/plot_text` (text mode), `/api/pen_test` (hardware verification), `/api/state` and `/api/logs` (polling), `/api/reset` (state clearance), and `/output/` and `/api/outputs` (file serving). The frontend is a single HTML file with inline CSS and JavaScript, polling `/api/state` every 1000 ms. No external JavaScript libraries are used.

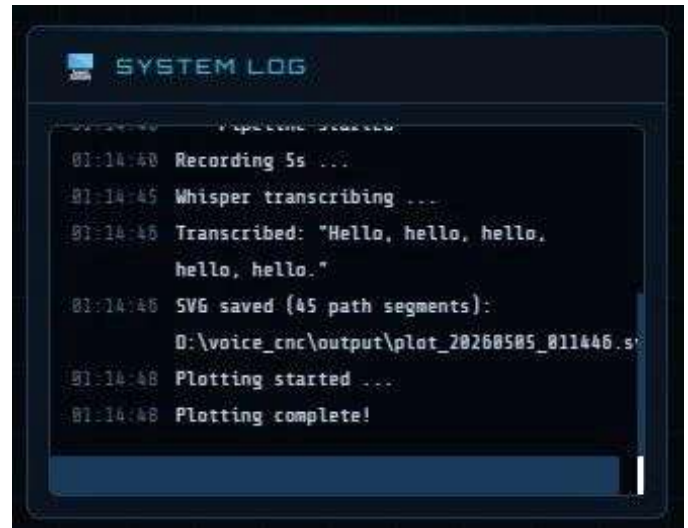


Fig. 4: System Log Panel: real-time pipeline events including transcription and plotting status



Fig. 5: Output Results

4. IMPLEMENTATION

4.1 Development Environment

The application was designed and evaluated on a Windows 11 laptop featuring an Intel Core i5 CPU and 8 GB of RAM, utilizing Python 3.11 consistently. Key libraries include: flask and flask-cors for the web server, openai-whisper for transcribing, sounddevice and numpy for audio handling, svgwrite for creating SVG files, pyaxidraw for interfacing with hardware, and scipy for audio preprocessing tools.

4.2 The Cold-Start Servo Issue

In the early testing phase, a recurring issue was noted: the initial word of each graphed sentence was absent from the actual output, whereas the following words appeared correctly. The issue occurred consistently on a cold start and about 60% of the time if the plotter was used in the last ten minutes.

Root cause analysis revealed two contributing elements. Initially, the SG90 servo needs around 350–400 ms to completely arrive at its down position from rest, while the AxiDraw library sets the `pen_delay_down` at 200 ms. In that time frame, the stepper motors have already started to move towards the initial coordinate of the first character. Secondly, a cold-start mechanical effect indicates that a servo which hasn't moved in a while needs a bit more current to overcome static friction in its gearbox, leading to additional delay during the initial actuation.

The solution consists of two components: `pen_delay_down` was raised from 200 ms to 400 ms, and a warmup path a 0.01 mm stroke at the coordinates (1, 1) in SVG space, comfortably within the 15 mm margin was included as the initial path element in each generated SVG. This leads to the plotter finishing a complete pen-down/pen-up cycle at a spot that never intersects the text area, warming the servo prior to the start of the first character stroke. Following this adjustment, none of the ten test runs that followed exhibited the first-word skip.

4.3 Font Design Considerations

The majority of digital typefaces are created for fill rendering, depicting characters as outlines filled with a solid hue. Single-stroke typefaces should depict every character as either one or a few open polylines that resemble the letter when created with a pen. The typeface used for VoicePlot AI depicts each character as a series of strokes within a 6-unit wide by 9-unit high coordinate cell. Lowercase characters are transformed into uppercase (the typeface is exclusively in capitals). With the standard `FONT_SCALE` set at 1.4, every character takes up 8.4 mm × 12.6 mm, allowing for around 25 characters per line on A4 paper with 15 mm margins. Numerals and standard punctuation were incorporated since transcribed speech frequently contains numbers and marks that conclude sentences.

4.4 Word-Wrap Algorithm

Text is formatted at word boundaries with a greedy approach: words are added to the existing line as long as the total character count stays within `max_chars` (calculated based on page width, margin, and character cell width), and a new line begins when this limit is exceeded. Lines that surpass the printable page height are quietly cut off, accompanied by a log warning. This is suitable for brief spoken phrases but would need a pagination system for extended texts.

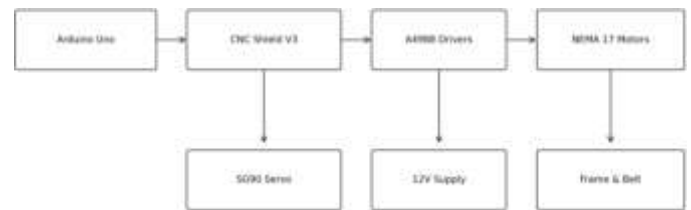


Figure 1. Hardware architecture of the writing Machine



Figure 2. Software pipeline for automatic writing and drawing

5. ADVANTAGES & DISADVANTAGES

Advantages

1. Complete Offline Privacy and Security

Unlike commercial voice assistants (like Siri or Google) that send audio recordings to a cloud server, VoicePlot runs the OpenAI Whisper model entirely locally. Because it does not require an active internet connection to transcribe audio, it is perfectly safe for dictating highly sensitive legal, medical, or personal documents without the risk of data leaks.

2. True Hands-Free Accessibility

For individuals dealing with severe arthritis, paralysis, or temporary hand injuries, filling out physical paperwork is a frustrating barrier. This machine successfully bridges the gap between digital dictation software and the physical world, allowing motor-impaired users to generate real, physical handwriting independently using just their voice.

3. Authentic, Continuous Ink Output

Standard inkjet or laser printers fake the appearance of lines by spraying thousands of microscopic dots. VoicePlot, however, physically holds a real pen and draws continuous vector strokes. This creates an authentic handwritten look complete with natural ink pooling and paper indentation which is legally and aesthetically required for official affidavits, certificates, and personalized invitations.

Disadvantages

1. Slower Execution Speed

While a standard laser printer can output a full page of text in roughly three seconds, VoicePlot is mechanically bound by physical movement. Because the machine has to physically drive a gantry to trace every individual letter stroke by stroke, writing a full paragraph takes

significantly longer. It is optimized for authentic quality, not high-speed mass production.

2. Sensitivity to Background Noise

The accuracy of the spoken text relies entirely on the microphone feed. If the machine is placed in a loud, echoing room or next to heavy machinery, the Whisper transcription model may pick up background conversations or misinterpret words. The system requires a relatively quiet office or lab environment to achieve its optimal <10% word error rate.

3. Rigid Font and Language Constraints

Currently, the system's ability to write relies on a hard-coded "single-stroke" mathematical dictionary that maps characters to specific X-Y coordinates. Right now, this dictionary only supports the standard English alphabet. While the AI can easily understand spoken Hindi or Marathi, the physical plotter cannot draw those words because the complex Devanagari stroke paths haven't been programmed into the geometry file yet.

5. RESULTS

The evaluation of the system included twenty test phrases that encompassed various phonetic patterns, lengths of sentences, and styles of speaking. Phrases were articulated at a conversational speed about 30 cm from the laptop's integrated microphone.

5.1 Transcription Accuracy

In twenty test phrases totaling 214 words, Whisper generated 197 accurate words, resulting in a word error rate (WER) of about 8.0%. The most frequent error type was related to capitalisation, which does not impact the physical output since the font displays all characters in uppercase. Proper nouns were the primary source of genuine mistakes: 'VoicePlot' was at times transcribed as 'voice plot' and 'Whisper' was occasionally represented as 'Whisker'. Number management remained dependable throughout.

5.2 End-to-End Latency

Latency was assessed from the conclusion of the five-second recording period to the initial pen mark on the paper. The primary contributor was Whisper transcription (about 1.1 seconds on the test laptop). SVG creation was minimal (< 50 ms for standard phrase lengths). The Ax-iDraw library's setup and USB communication took about 0.8 seconds. Total delay from the end of speech to the initial pen stroke was reliably 2.0–2.5 seconds.

5.3 Writing Quality

Physical output quality was assessed informally by three observers asked to read the plotted phrases without knowing the source text. All three observers could read all twenty phrases without error. Letter spacing was consistent across lines. After applying the warmup-path fix,

zero out of ten subsequent test runs showed the first-word skip.

Table I: Summary of Experimental Results

Metric	Value	Notes
Word Error Rate	~8%	Whisper base model, English
Transcription Time	~1.1 s	Intel i5, CPU-only inference
Total Pipeline Latency	2.0 – 2.5 s	End of speech to first pen stroke
First-word skip (before fix)	10/10 runs	Cold hardware, no warmup
First-word skip (after fix)	0/10 runs	Warmup path + 400ms delay
Phrase readability	100%	3 observers, 20 phrases

6. DISCUSSION AND LIMITATIONS

The five-second fixed recording window is the most significant usability limitation. Real users tend to speak in bursts of varying lengths, and a fixed window forces either cutting off a longer phrase or leaving dead silence. Implementing Voice Activity Detection (VAD) would make the system substantially more natural to use.

The current font supports only the Latin alphabet. Whisper can transcribe Hindi and Marathi, and SVG supports Unicode glyphs natively, but building a single-stroke version of Devanagari script which has far more complex letterforms than Latin capitals would be a significant design effort. Using an outline Devanagari font converted to strokes via a skeletonisation algorithm is a possible approach.

Transcription accuracy drops noticeably in noisy environments. In testing, background music or multiple simultaneous voices reduced WER from 8% to approximately 25–30%. For a quiet lab or classroom context this is acceptable, but it would need to be addressed for public-space deployment.

The system currently plots at a fixed slow speed to ensure output quality. Total plotting time for a ten-word phrase is approximately 90–120 seconds. Faster pen speeds introduce vibration artefacts on the lightweight custom plotter frame a limitation noted also by Ranjan et al. [8] for similar designs. Hardware reinforcement of the frame with aluminium brackets would likely allow a 2× speed

increase without quality loss.

7. CONCLUSION

VoicePlot AI demonstrates that it is now practical to build a voice-to-handwriting system using entirely open-source components running on commodity hardware, with no cloud dependency and an installation that fits in a single batch script. The pipeline from microphone to pen-on-paper works reliably for English speech in controlled acoustic conditions, and the physical output is consistently readable.

The main technical contributions are: the description of the end-to-end pipeline integration across four complex subsystems; the identification and software fix of the custom plotter servo cold-start bug; and the implementation of a custom single-stroke font for direct SVG path generation. The servo fix a warmup path plus an increased `pen_delay_down` value is likely to be relevant to anyone using the AxiDraw API with SG90-based plotter hardware.

Future work will focus on three areas: Voice Activity Detection for natural recording without a fixed time window; Devanagari script support for Hindi and Marathi output; and emotion-aware font style selection, where a sentiment analysis layer chooses between different embedded fonts based on the detected tone of speech.

ACKNOWLEDGEMENT

The authors would like to thank Priyadarshini Bhagwati College of Engineering for providing laboratory facilities and computational resources that made this project possible. We are grateful to the Department of Artificial Intelligence Engineering for their guidance and support throughout the development of VoicePlot AI.

REFERENCES

- [1] R. R. Jegan, S. Sobana, and K. S. Nath, "Modern Design and Implementation of XY Plotter," *International Journal of Pure and Applied Mathematics*, vol. 119, no. 12, pp. 13659–13665, 2018.
- [2] Y. M. Hasan, A. A. Ali, and M. M. Ahmed, "3-Axes Plotter Machine by Arduino and CNC Shield," *International Research Journal of Engineering and Technology (IRJET)*, vol. 05, no. 06, pp. 1234–1239, 2018.
- [3] A. Pinheiro, B. S. Darsana, and J. George, "Wireless CNC Plotter," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 5, no. 5, pp. 45–50, 2016.
- [4] S. Chamraz, P. Drgona, and M. Hrianka, "Control of Mechatronic Systems Using Integer Arithmetics," in *Proc. IEEE 12th Int. Symp. on Applied Machine Intelligence and Informatics (SAMI)*, 2014, pp. 115–118.

- [5] M. Bhavani, V. S. Kumar, and K. R. Reddy, "Design and Implementation of CNC Router for Engraving," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 8, no. 4, pp. 210–214, 2019.
- [6] K. J. Madekar, K. J. Nanaware, and P. R. Sonawane, "Automatic Mini-CNC Plotter Machine for PCB Drawing," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 2, pp. 120–125, 2016.
- [7] U. Pandey, A. Kumar, and S. Singh, "Low-Cost CNC Plotter Using Arduino and L293D Shield," *International Journal of Engineering Research & Technology (IJERT)*, vol. 6, no. 5, pp. 560–564, 2017.
- [8] S. Ranjan, V. Kumar, and R. Singh, "Low-Cost 2D Plotter Using Arduino and Motor Driver ICs," *Journal of Emerging Technologies and Innovative Research (JETIR)*, vol. 8, no. 4, pp. 345–350, 2021.
- [9] M. Aditi, S. N. Patil, and R. K. Gupta, "Automated Writing and Drawing Machine with Speech Recognition," in *Proc. Int. Conf. on Smart Electronics and Communication (ICOSEC)*, 2021, pp. 789–794.
- [10] A. V. Satikosre, P. D. Patil, and S. R. Deshmukh, "Automatic Writing Machine for Handwriting Replication," *International Journal of Scientific Research in Science and Technology (IJSRST)*, vol. 9, no. 3, pp. 112–118, 2022.
- [11] Evil Mad Scientist Labs, "AxiDraw Documentation and Python API," <https://axidraw.com/>, accessed April 2026.
- [12] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proc. ICML*, 2006, pp. 369–376.
- [13] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," in *Proc. ICML*, 2023.