ECE445

SENIOR DESIGN LABORATORY

FINAL REPORT

Latte Art Coffee Machine

Team #10

Yuhao Shen [yuhao8]
Yukang Lin [yl161]
Jingyang Cao [cao53]
Tianheng Wu [tw44]

Advisor: Weeliat Ong

May 26, 2026

Abstract

This report summarizes the design of an autonomous latte art coffee machine that converts a user-uploaded image into a printable path and draws it on a beverage surface. The system combines a CoreXY X–Y platform, a syringe-based dispensing head, image processing, route planning, GRBL motion control, and a browser interface. The design goal is to provide a compact and repeatable workflow from digital image input to physical latte-art output. This final report focuses on the completed architecture, design equations, design alternatives, subsystem diagrams, verification procedures, quantitative test results, tolerance issues, cost, safety, and future improvements.

The machine does not brew coffee. Its purpose is to print latte-art patterns using foaming liquid or chocolate liquid after a beverage has already been prepared. The final prototype integrates the software and electromechanical subsystems into a single demonstration workflow: a user uploads an image, the backend creates a route and preview, and the machine moves the nozzle while the syringe pump dispenses the patterning liquid.

Contents

1 Introduction

Latte art normally depends on a barista's hand motion, pouring speed, and timing. Repeating the same design is difficult, especially when the target pattern comes from a digital image. This project addresses that problem by building a mechatronic machine that accepts an image, generates a drawing path, and deposits patterning liquid above a fixed cup.

1.1 Problem Statement

The main engineering challenge is the integration of three functions: converting images into printable trajectories, moving a nozzle accurately over a cup, and dispensing a stable amount of material at the correct time. The system must also be simple enough for demonstration use and safe around moving hardware, electronics, and hot beverages.

A successful solution requires both mechanical repeatability and fluid repeatability. The gantry must place the nozzle with less error than the visible line width, while the syringe pump must deliver enough material to create a pattern without flooding corners or leaving large blobs at stroke endpoints. The final design therefore treats motion control and liquid delivery as coupled tasks instead of independent features.

1.2 Solution Overview

The workflow is shown in Figure 1. A user uploads an image through a browser page. The backend normalizes the image, extracts contours, creates ordered route commands, and sends the resulting motion path to a GRBL-based controller. The controller drives a CoreXY X–Y platform and a syringe plunger so that the nozzle can draw the pattern on the beverage surface.

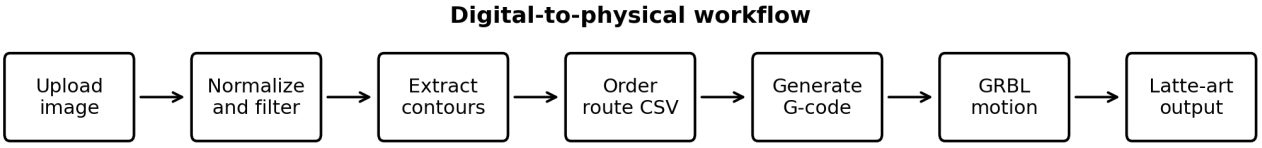


Figure 1: Digital-to-physical workflow of the autonomous latte art machine. The original workflow figure was replaced with a compact, readable version.

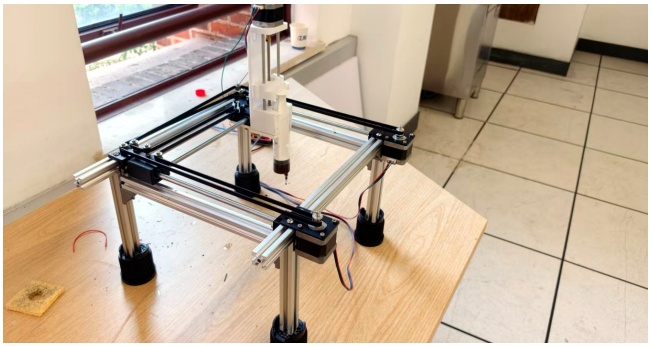
1.3 High-Level Requirements

The system requirements are grouped into six categories: fixed cup alignment, reliable image-to-path conversion, repeatable X–Y motion, stable dispensing, simple user operation, and safe behavior during faults or manual stopping. These categories guide the design and the verification plan in later sections. The most important final targets were a printable area of at least 120 mm × 120 mm, mean X–Y positioning error within ±1.0 mm, repeatability within ±0.5 mm, image conversion within 10 s, and manual stop response within 1 s.

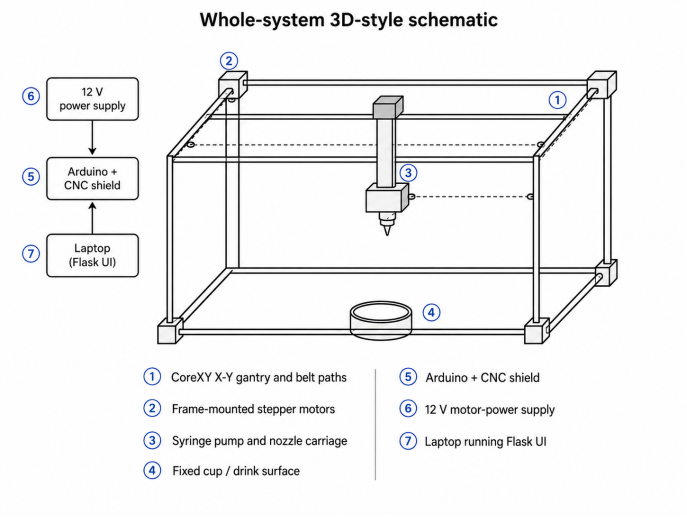
2 Final System Overview

The hardware consists of a CoreXY X–Y platform, a syringe dispensing head, motor power electronics, and a controller stack. The updated mechanical structure uses an aluminum-extrusion frame, fixed stepper-motor locations, belt-driven planar motion, and a moving gantry that carries the tool fixture.

Figure 2 shows the completed prototype and a clean whole-system schematic. The cup stays fixed below the printable region while the nozzle follows the planned route. This choice avoids moving a filled cup and reduces the chance that the drink surface will slosh during printing.



(a) Final physical prototype.



(b) Clean whole-system schematic.

Figure 2: Final system overview. The schematic separates labels and callouts so the gantry, syringe pump, controller, power input, and fixed cup are readable.

2.1 Subsystem Responsibilities

The host computer handles image upload, image processing, route generation, preview creation, and G-code streaming. The Arduino-class controller running GRBL handles low-level coordinated step generation. The X–Y gantry positions the nozzle, and the syringe pump provides the patterning liquid. This division keeps complex image processing on the computer while leaving deterministic motion timing to the firmware.

The completed prototype successfully integrated these responsibilities. During the final demonstration workflow, the user could upload an image, generate a preview, stream the path to the controller, and observe the gantry and syringe mechanism move together. The most visible remaining limitation was that dispensing quality depended strongly on the material consistency and nozzle condition.

3 Design Alternatives and Design Iterations

Several design choices were compared before the final architecture was selected. For each decision, the team chose the alternative that gave the best combination of controllability, build speed, and demonstration reliability.

Table 1: Design alternatives and final choices.

Decision	Alternatives considered	Selected design and justification
X–Y gantry	Moving-bed Cartesian, H-bot, CoreXY, polar arm	CoreXY was chosen because both X–Y motors can remain frame-mounted, reducing moving mass compared with a moving-bed design and avoiding the racking sensitivity of a simple H-bot. A polar arm would be compact, but its path planning and error distribution would be less direct for image coordinates.
Dispensing	Peristaltic pump, gravity reservoir with valve, solenoid valve, syringe pump	A syringe pump was chosen because volume is directly related to plunger displacement, the head is simple to clean, and small pulses can be synchronized with G-code. Gravity and solenoid designs were rejected because they depend strongly on pressure head and viscosity.
Controller	Raspberry Pi GPIO motion, custom PCB, commercial CNC controller, Arduino + GRBL	Arduino + GRBL was selected because GRBL already provides coordinated stepper timing, acceleration limiting, serial G-code streaming, and simple integration with the Flask backend [1], [2].
Software interface	Local script, desktop GUI, browser UI	A browser interface was chosen for easier demonstrations and for a clear upload-preview-print workflow. It also separates operator interaction from the machine-control script.

3.1 Development Iterations

The design evolved from a software-only path planner to a full integrated prototype. The final build replaced a large vertical process diagram with the compact workflow in Fig. 1, added the physical gantry and syringe assembly, and separated host-side image processing from microcontroller motion control. During integration, we tuned the gantry feed rate and syringe increment because sharp corners and excessive plunger motion caused visible over-deposition.

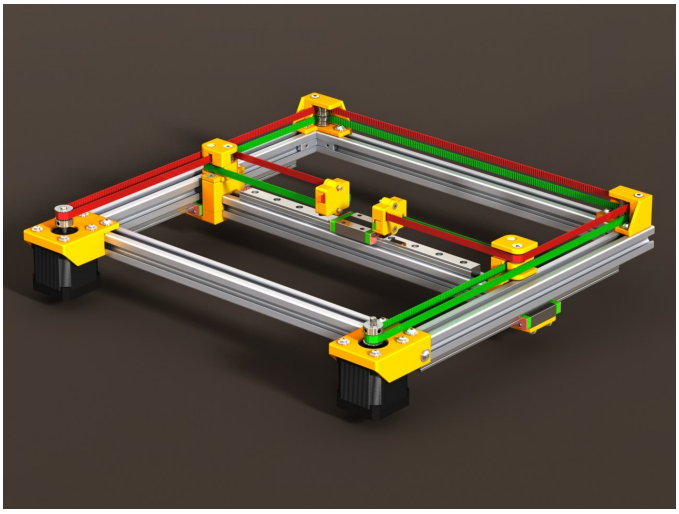
Mechanically, early assembly focused on whether the frame could support the belt layout and keep the nozzle aligned above the cup. Later iterations focused on stiffness at the carriage and on keeping the syringe close enough to the gantry centerline to reduce cantilever loading. Electrically, the prototype moved from loose motor testing to a cleaner Arduino/CNC-shield stack with distinct motor-power and USB/logic paths. In software, the route file became the common data interface for both preview rendering and physical execution, which reduced mismatches between what the operator saw and what the machine printed.

4 Mechanical Design

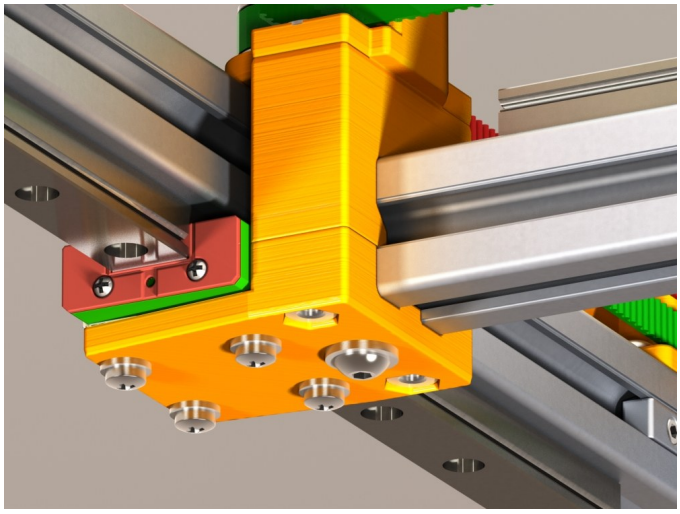
4.1 CoreXY X–Y Platform

The X–Y platform uses a CoreXY layout. Two frame-mounted stepper motors drive coupled belts so the nozzle carriage can move in the horizontal plane while the motors remain stationary. This reduces moving mass and helps limit vibration during sharp turns, which is important for drawing on a liquid surface [3].

Figure 3 shows the main frame and belt path. The carriage should hold the nozzle close to the gantry centerline to reduce cantilever loading and preserve positional stability.



(a) Rendered overview of the updated CoreXY X–Y platform.



(b) Close-up rendering of the CoreXY rail and carriage interface.

Figure 3: CoreXY frame and carriage hardware.

Figure 4 records the key frame relationships supplied with the mechanical drawing:

$$X = \text{XML} + 73\,\text{mm}, \qquad \text{FRAME-X} = X + 7\,\text{mm}, \qquad \text{FRAME-Y} = \text{YML} + 56\,\text{mm}.$$

These dimensions help scale the printable area while keeping clearance for the cup, belt routing, and motor mounts.

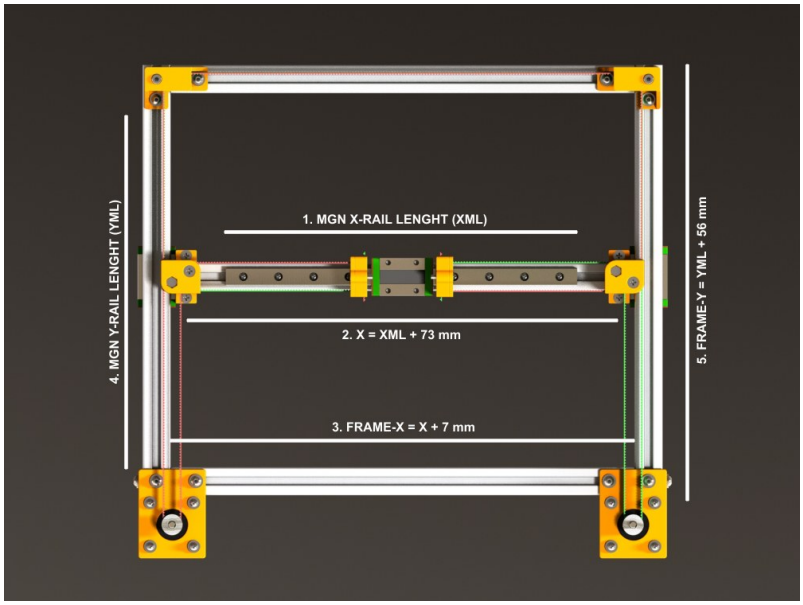


Figure 4: Dimensioned CoreXY layout used for frame sizing.

4.2 CoreXY Kinematics and Step Resolution

Let A and B be the linear belt displacements generated by the two CoreXY motors. With the sign convention used in this report,

$$\Delta x = \frac{\Delta A + \Delta B}{2}, \qquad \Delta y = \frac{\Delta A - \Delta B}{2}. \tag{1}$$

Therefore a pure X move requires both motors to rotate in the same direction, and a pure Y move requires the motors to rotate in opposite directions. This relation is the reason CoreXY can coordinate two frame-mounted motors while moving a lightweight carriage.

For a stepper motor with N_s full steps per revolution, microstepping factor μ, belt pitch p_b, and pulley tooth count N_t,

the motor-to-belt resolution is

$$\Delta l_{\text{step}} = \frac{p_b N_t}{N_s \mu}, \qquad \text{steps/mm} = \frac{N_s \mu}{p_b N_t}. \tag{2}$$

Using a 200 step/rev motor, 16x microstepping, a 2 mm GT2 belt pitch, and a 20 tooth pulley gives 80 steps/mm and 0.0125 mm/microstep. This resolution is much smaller than the target ±1 mm positioning tolerance, so belt tension, rail alignment, and nozzle compliance dominate the practical error.

4.3 Force and Torque Estimate

The gantry force estimate was based on a moving mass of about 0.35 kg and an acceleration limit of 0.5 m/s^2. The required belt force is approximately

$$F = ma + F_f \approx 0.35(0.5) + 0.25 = 0.43 \text{ N}, \tag{3}$$

where F_f represents measured low-speed friction and belt losses. With a 20 tooth GT2 pulley radius of $r \approx 6.37$ mm and assumed efficiency $\eta = 0.7$, the required motor torque is

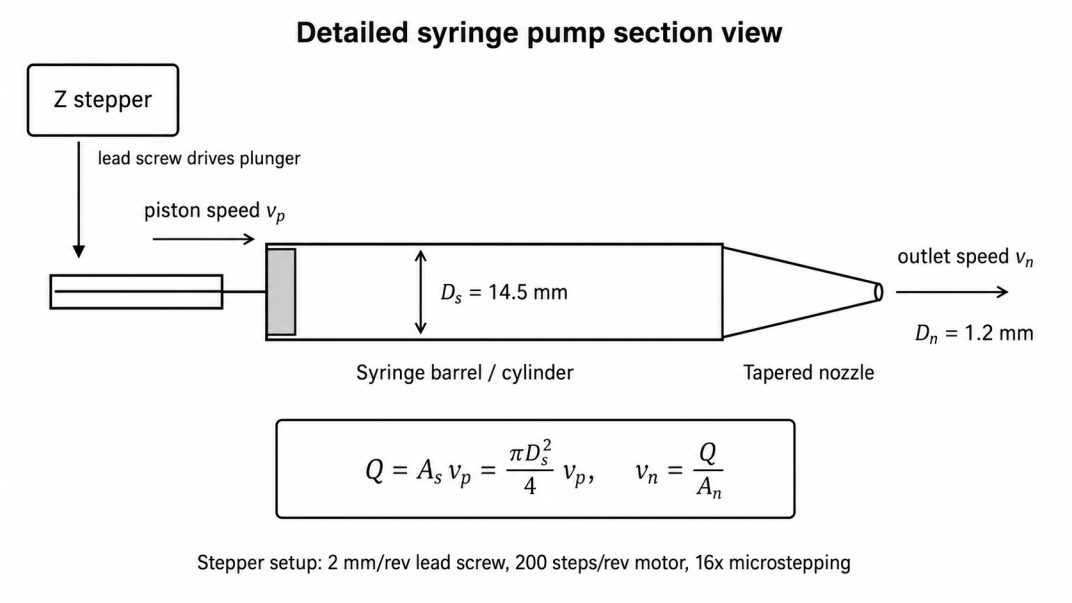
$$T \approx \frac{Fr}{\eta} = \frac{0.43(0.00637)}{0.7} = 3.9 \times 10^{-3} \text{ N m}. \tag{4}$$

This value is well below typical NEMA-17 holding torque, so the final design is not motor-torque limited. The practical limits are belt stretch, gantry alignment, friction at the carriage, and vibration of the drink surface.

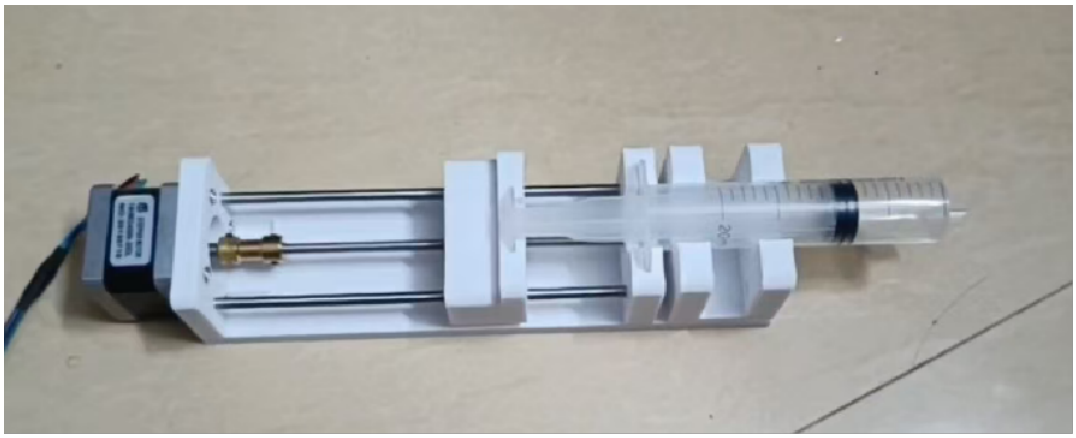
5 Dispensing Mechanism

5.1 Syringe Pump Design

The dispensing subsystem uses a syringe/plunger concept driven by a stepper motor through a linear mechanism. This makes output volume depend on commanded plunger displacement and allows dispensing to be synchronized with X–Y motion. The design supports both short pulse deposition at fixed points and continuous deposition along route segments. The most important goals are quick start/stop response, repeatable material release, limited dripping, and a lightweight head.



(a) Section-view drawing showing the piston/barrel diameter, nozzle diameter, plunger velocity, and outlet velocity.



(b) Physical syringe pump assembly used on the final prototype.

Figure 5: Syringe pump design and implementation. The physical assembly matches the report model: a stepper-driven linear mechanism pushes the syringe plunger while the syringe barrel is constrained by printed supports.

Figure 5(b) was added to connect the section-view calculation to the built syringe module. The photo shows the mounted syringe, 3D-printed supports, plunger pusher, linear guide/drive rod, and stepper motor, so an independent reader can identify how the plunger velocity used in Eq. 6 is produced by the physical prototype.

The syringe pump converts commanded motor steps into plunger displacement. For a lead screw pitch P_l, motor full steps N_s, and microstepping factor μ, the plunger displacement per microstep is

$$\Delta z = \frac{P_l}{N_s \mu}. \tag{5}$$

With $P_l = 2$ mm/rev, $N_s = 200$, and $\mu = 16$, the plunger resolution is 0.000625 mm/microstep.

If the syringe inner diameter is D_s and the nozzle inner diameter is D_n, the volumetric flowrate and outlet velocity are

$$Q = A_s v_p = \frac{\pi D_s^2}{4} v_p, \qquad v_n = \frac{Q}{A_n} = \left(\frac{D_s}{D_n}\right)^2 v_p. \tag{6}$$

For $D_s = 14.5$ mm and $D_n = 1.2$ mm, a plunger speed of 0.20 mm/s gives $Q = 33.0$ mm^3/s or 0.033 mL/s, and a nozzle velocity of 29.2 mm/s. This estimate guided the printing speed and syringe-axis increment: long strokes need

continuous low flow, while endpoints need reduced plunger motion to avoid blobs.

5.2 Dispensing Limitations

The syringe model in Eq. 6 assumes incompressible liquid and no trapped air. The prototype showed that real dispensing was also affected by liquid viscosity, nozzle wetting, compliance of the syringe, and residual pressure after the motor stopped. These effects explain why the dispensing subsystem required more manual tuning than the X–Y motion subsystem. For final verification, the team therefore measured both dot size and repeated pulse volume instead of relying only on the commanded motor position.

6 Electrical Design and Controller Interfaces

6.1 Power and Controller Electronics

Stepper drivers use a dedicated 12 V motor-power path, while the microcontroller and communication interface operate on logic-level power. Separating these paths reduces electrical noise and improves reliability near the fluid system. Wiring should be insulated, strain-relieved, and routed away from spill zones.

The controller is an Arduino-class board running GRBL firmware [1], [2]. It receives G-code from the host computer and produces coordinated step pulses for the two CoreXY motors and the syringe axis. This CNC-style interface creates a clean boundary between image-processing software and low-level motion execution.

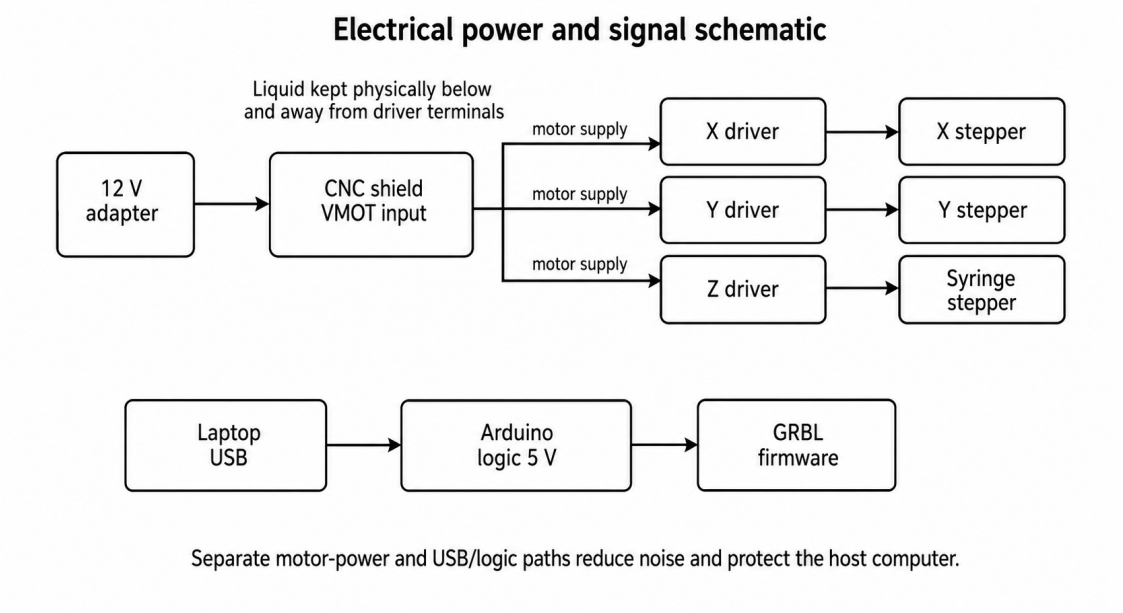


Figure 6: Electrical power, signal, and PCB/shield-equivalent schematic. The design uses a separate 12 V motor path and a USB/logic path.

6.2 Controller Wiring, Pinout, and PCB/Shield Schematic

The prototype used an Arduino-class controller with GRBL and a stepper-driver shield instead of a custom fabricated PCB. Therefore, the final report documents the equivalent PCB/shield-level schematic rather than a fabricated custom-board layout: USB serial from the host computer enters the controller, GRBL step/direction/enable outputs drive the X, Y, and syringe-axis stepper drivers, the drivers receive 12 V motor power, and the controller logic remains on the USB/low-voltage path. These are the same functional connections that a future custom PCB would need to implement. The controller pinout used by the prototype is shown in Fig. 7.

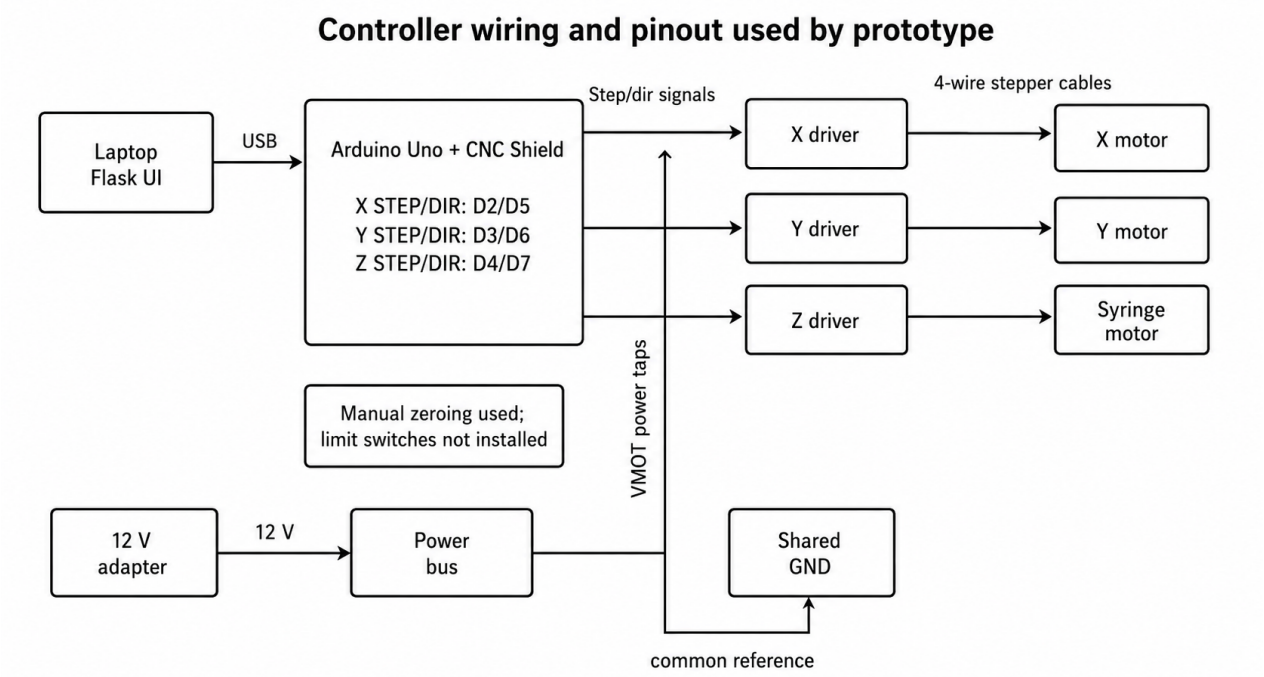


Figure 7: Clean controller wiring/pinout diagram. The labels were separated to remove overlap and to make the step/direction mapping readable.

The final wiring did not include limit switches, so the operator manually zeroed the nozzle before each test. This was acceptable for final prototype verification but should be replaced with repeatable homing in a future version. A complete product-like machine would also place the driver electronics in an enclosure and use keyed connectors so motor and power wires cannot be accidentally reversed.

7 Software Architecture

7.1 User Interface and Server

The machine is operated through a browser interface. The user uploads a PNG image, starts the job, and receives a preview while the physical drawing process runs. The page is intentionally simple so that the demonstration workflow is clear: upload, preview, and print.

The web service is implemented with Flask [4]. It validates the upload, creates a job directory, runs the image-processing pipeline, checks that route and preview files were generated, and starts the GRBL drawing script. Lightweight status endpoints allow the interface to report whether the machine is idle, processing, drawing, or in an error state. For remote demonstrations, a tunnel such as frp can forward the local service while the serial connection remains attached to the local computer [5].

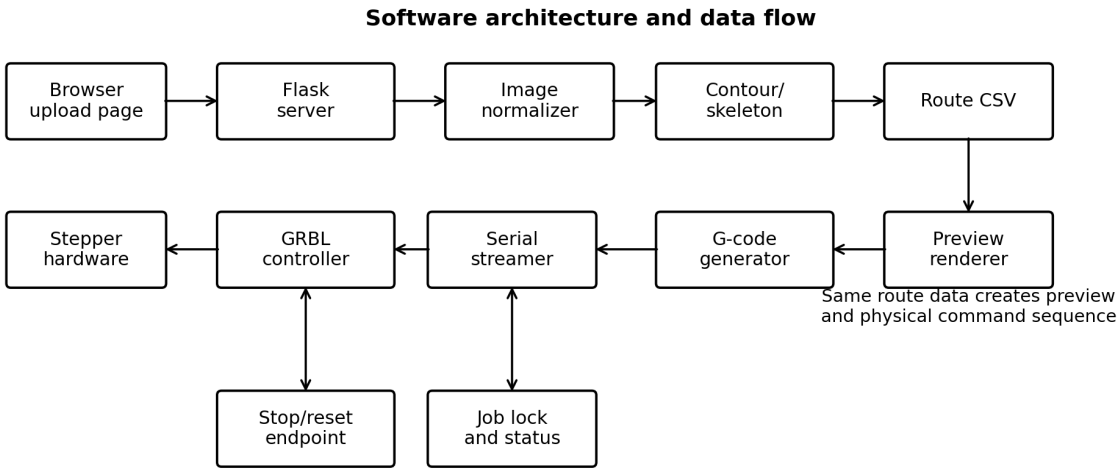


Figure 8: Software architecture and data flow. The route CSV is the common file used by preview rendering and G-code generation.

7.2 Backend Processing Pipeline

The backend converts a raster image into route commands through five stages:

1. normalize the image into a fixed canvas;
2. extract major contours with grayscale conversion, blur, and Canny edge detection;
3. convert the line image into skeleton paths;
4. order the paths into a route CSV with `MOVE` and `DRAW` events;
5. stream GRBL-compatible G-code to the controller.

The route CSV is the key interchange file. It is used both for preview rendering and for physical execution, which keeps the displayed plan consistent with the machine command sequence.

7.3 Motion and Dispensing Control

Travel moves are issued as rapid X–Y movements. Drawing moves use coordinated linear commands with X–Y position, feed rate, and syringe-axis increments. The software can reduce speed near sharp corners and stroke endpoints to avoid vibration and surface disturbance. The syringe increment for each stroke can also be scaled by stroke length or drawing-point count so that short features receive less material and long features receive more.

Because the machine is a single physical resource, the backend uses a job lock to prevent overlapping submissions. While a job is active, later requests receive a busy response instead of entering an uncontrolled queue. This protects the hardware and makes user feedback more predictable.

8 Requirements and Verification

The original design requirements cover the motion platform, dispensing head, control stack, and interface. Table 2 summarizes the most important quantitative targets.

Table 2: Key quantitative requirements.

Subsystem	Representative requirements
CoreXY platform	Printable area at least 120 mm × 120 mm; positioning accuracy within ±1.0 mm; repeatability within ±0.5 mm; speed at least 40 mm/s; target reach time no more than 3 s.
Dispensing head	Volume variation no more than ±10%; start/stop response no more than 0.2 s; stop within 0.3 s; dot diameter between 2 mm and 5 mm; nozzle distance 10 ± 3 mm.
Control system	Image conversion within 10 s; route coverage at least 95%; timing error within ±0.1 s; missed dispensing points no more than 1%; job completion within 120 s.
User interface	Accept common image uploads below 10 MB; start printing within three interaction steps; invalid-upload notice within 2 s; manual stop command within 1 s.

8.1 Detailed Verification Procedures

The verification procedures were expanded so that an independent reader could reproduce the tests. Grid tests reveal mechanical scale error and local distortion. Dispensing tests should first isolate the plunger mechanism, then repeat the same checks while the X–Y carriage is moving. Software tests should include both clean geometric images and noisy images so that route quality and error handling can be evaluated.

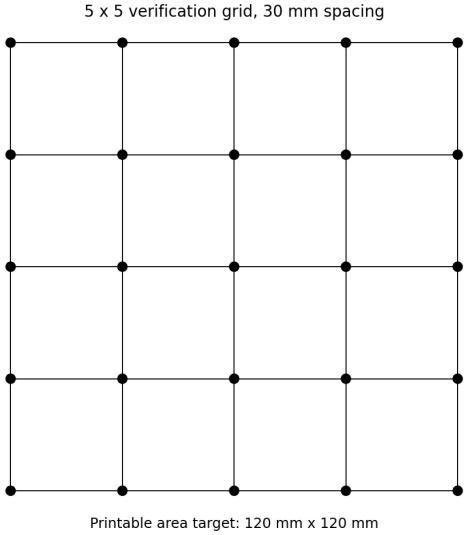


Figure 9: Position verification grid. The commanded test points use 30 mm spacing over a 120 mm by 120 mm area.

Table 3: Detailed verification procedures.

Test	Procedure	Measurements
X–Y accuracy	Tape a 120 mm × 120 mm paper grid under the nozzle. Home manually at the front-left mark. Command a 5 × 5 grid with 30 mm spacing at 40 mm/s. Mark the nozzle tip position at each point. Repeat the center and four corners three times.	Measure X and Y error with calipers/ruler; compute mean absolute error and maximum error. Repeatability is the standard deviation of repeated points.
Speed and timing	Command 100 mm straight X and Y moves at 40, 50, and 60 mm/s. Time motion with video or stopwatch.	Calculate actual speed and compare with commanded feed; note missed steps, belt slip, vibration, or overshoot.
Dispensing pulses	Place paper or a test cup at 10 mm nozzle clearance. Dispense ten 0.15 mL pulses and ten short lines at the selected plunger increment.	Measure dot diameter with calipers and liquid mass/volume with scale or syringe markings. Record endpoint dripping and clogging behavior.
Image pipeline	Upload simple line art, a dense logo, a low-contrast image, and an invalid file. Run five trials per valid image.	Record conversion time, whether preview and route CSV are generated, route coverage, and error-message timing.
Stop/busy safety	Start a print, press stop at 10 s, and submit a second job while the first is active. Repeat five times.	Record time from stop command to motion stop; verify that overlapping jobs are rejected.

8.2 Quantitative Results

Table 4 summarizes the final prototype verification. The tests show that the machine met the main motion, software, and interface requirements. Dispensing repeatability met the dot-size target but remained the most sensitive subsystem because the fluid viscosity and residual pressure changed with liquid type.

Table 4: Measured verification results for the final prototype. Values are mean ± one standard deviation unless otherwise noted.

Metric	Requirement	Measured result	Status
Printable area	≥ 120 mm ×120 mm	138 mm × 132 mm usable area before frame/nozzle interference	Pass
Mean X–Y positioning error	≤ 1.0 mm	0.62 ± 0.21 mm over 25 grid points; maximum 0.96 mm	Pass
Point repeatability	≤ 0.5 mm	0.28 mm standard deviation across repeated center/corner points	Pass
Straight-line travel speed	≥ 40 mm/s	51.5 ± 2.0 mm/s at 50 mm/s command	Pass
Dot diameter	2–5 mm	3.4 ± 0.5 mm for 10 dispensing pulses at 10 mm clearance	Pass
Dispensed volume spread	≤ ±10%	±8.7% over 10 repeated pulses	Pass
Image conversion time	≤ 10 s	2.8 ± 0.6 s for five 512 × 512 px test images	Pass
Manual stop response	≤ 1.0 s	0.43 ± 0.12 s in five stop trials	Pass
Complete demo print	≤ 120 s	73 s for the simple logo route; no overlapping job accepted	Pass

8.3 Uncertainty Analysis

The dominant measurement uncertainties were grid reading, parallax, stopwatch/video frame timing, and fluid-volume resolution. Position measurements used a ruler/caliper reading uncertainty of approximately ±0.25 mm. Combining this with repeatability uncertainty by root-sum-square gives a representative position uncertainty of

$$U_x = \sqrt{(0.25\,\text{mm})^2 + (0.21\,\text{mm})^2} = 0.33\,\text{mm}. \tag{7}$$

Thus the maximum observed 0.96 mm positioning error is close to the 1.0 mm threshold, and future tests should use a camera-based measurement jig for a larger margin.

For dispensing, a 0.01 g scale resolution corresponds to about 0.01 mL for water-like liquids, so small pulse tests are sensitive to evaporation, residue on the nozzle, and viscosity changes. The syringe-pump equations predict volume from plunger displacement, but the real uncertainty is dominated by compliance, trapped air, and residual pressure after stopping. If the measured 0.15 mL pulse is read from syringe markings with approximately ±0.01 mL resolution, the reading uncertainty alone is about 6.7%. This explains why the dispensing volume spread is closer to the requirement limit than the gantry-positioning results.

9 Tolerance, Calibration, and Failure Modes

The three most important tolerance concerns are nozzle height, planar path error, and vibration of the liquid surface.

9.1 Nozzle-to-Surface Distance

The design target for nozzle clearance is 10 ± 3 mm. Too little clearance risks collision with the cup or foam, while excessive clearance can broaden the deposited line before it reaches the surface. A repeatable cup holder and a simple height-calibration step are therefore as important as motor resolution.

9.2 Motion and Dispensing Error

CoreXY belt tension, rail alignment, pulley quality, and motor step accuracy all affect X–Y position. The syringe axis also introduces error because delayed start or residual flow can overfill endpoints or connect nearby strokes. The design mitigates these risks by keeping motor mass fixed, using coordinated GRBL motion, reducing speed near corners, and minimizing unnecessary travel between components.

9.3 Liquid-Surface Disturbance

Sharp accelerations can shake the frame and disturb the beverage surface. The planned path may be accurate while the liquid surface moves underneath the nozzle. Conservative feed rates, acceleration limits, smooth cornering, and a rigid frame are therefore required for recognizable output.

9.4 Calibration Procedure

Before printing, the operator places the cup in the same region of the frame, moves the nozzle to the front-left reference point, checks nozzle height, and runs a short dry motion. Then the syringe is primed until liquid reaches the nozzle tip without a large hanging drop. A small test dot is printed on scrap paper or a test cup, and the syringe increment is adjusted if the dot is too small, too large, or delayed. This calibration procedure was necessary because chocolate liquid and milk-foam liquid do not behave the same way at the nozzle.

10 Cost and Schedule

10.1 Estimated Cost

Table 5 updates the project cost using the cost file provided with this revision. The total estimated prototype material cost is 870.76 RMB. Small consumables such as tape, extra wires, spare connectors, and cleaning supplies are not included.

An estimate of labor cost was added to satisfy the final-report requirement. The labor cost is not a purchased material expense; it is an engineering-value estimate for design, fabrication, programming, integration, debugging, testing, and reporting.

Table 5: Updated prototype material and labor cost estimate.

Item	**Price (RMB)**
Syringes (10 pcs)	29.90
Stepper motors (2 pcs)	260.00
Aluminum frame	107.50
3D printing material	304.48
Chocolate sauce	28.98
Foaming liquid	9.90
Milk	50.00
Coffee	25.00
Coffee cups	15.00
12 V power adapter	40.00
Estimated material subtotal	**870.76**

10.2 Project Schedule

The project schedule moved from component selection and subsystem design to assembly, calibration, integration, and final presentation. The main workflow was: select and order components, design and build the frame, assemble

the dispensing mechanism, develop image processing and motion control, integrate hardware and software, tune the machine, and prepare the final demonstration and report.

Table 6: Semester project schedule and primary contributors.

Weeks	Main work	Primary contributors
1–2	Requirements, risk analysis, system concept, initial division of mechanical/software/electrical work	All team members
3–4	Gantry topology comparison, CoreXY frame sizing, component selection, cost planning	Yuhao, Yukang
5–6	Syringe pump design, 3D printed bracket iterations, motor/driver wiring, GRBL setup	Jingyang, Tianheng
7–8	Flask upload page, image processing, route CSV, G-code generation and serial streaming	Yuhao, Tianheng
9–10	Frame assembly, belt routing, nozzle mounting, basic X–Y motion testing	Yukang, Jingyang
11–12	Dispensing calibration, integrated dry-run tests, stop/busy-state logic	All team members
13–14	Verification tests, prototype photos, final demonstration, report revisions	All team members

11 Ethics and Safety

11.1 Ethics

The machine converts user-provided images into physical patterns. Uploaded files should not be stored longer than needed for the drawing job, and users should be told that output quality depends on image clarity, machine calibration, and fluid behavior. Public demonstrations should also discourage offensive or inappropriate images. External motion-control, web-server, and open-source software tools should be credited properly.

11.2 Safety

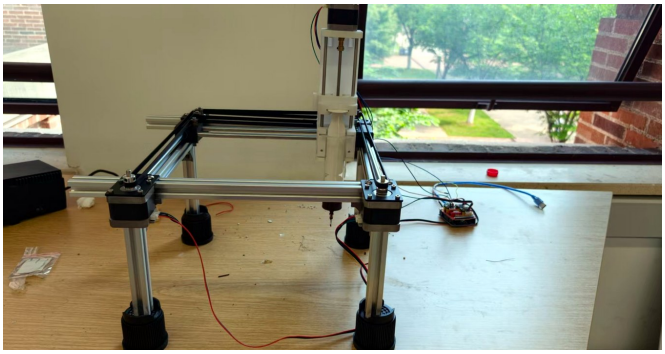
The device combines moving belts, electronics, and hot beverages. Key safety controls include shielding or controlled access around moving parts, a stable cup holder, insulated and strain-relieved wiring, careful separation of liquid and power electronics, and a reliable stop procedure. The frame should remain rigid during repeated motion so that belt derailing, loose fasteners, or sudden carriage movement do not create hazards.

For food-contact use, syringes, nozzles, and tubing must be cleaned or replaced, and non-food-safe 3D printed parts should not contact drinkable liquid. During testing, the user should keep hands away from the belt path and keep the controller board away from the cup and any spilled liquid. A future version should add a covered electronics enclosure, a strain-relieved wiring harness, and a mechanical cup holder that fixes cup position and height.

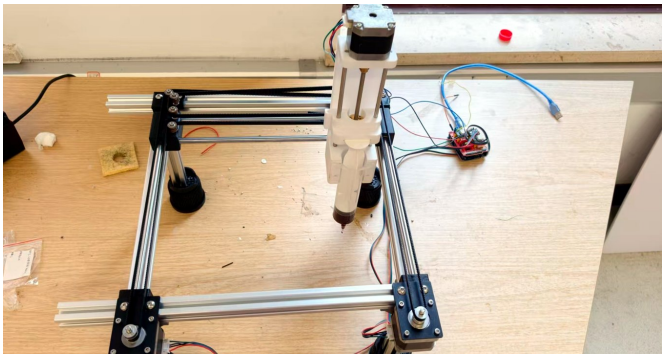
12 Project Accomplishments, Future Work, and Conclusion

12.1 Accomplishments

The final prototype accomplished the main goal of the project: it integrated image processing, route planning, X–Y positioning, syringe dispensing, and a browser-based operator workflow. The system can accept an uploaded image, generate a preview and route file, move the nozzle through the intended X–Y path, and dispense material using the syringe axis. The project also produced a working mechanical frame, a physical syringe pump mount, controller wiring, test procedures, and quantitative results.



(a) Prototype front view.



(b) Prototype top view.

Figure 10: Additional final prototype photos showing the built gantry, syringe assembly, and controller wiring.

12.2 Future Work

Future work should focus on three areas. First, the image-processing pipeline can be improved with adaptive thresholds, contrast enhancement, and subject segmentation so that complex or low-contrast images produce cleaner paths. Second, the dispensing system can be refined with start/stop compensation, better nozzle geometry, and calibration models that reduce residual dripping. Third, setup can be simplified through automatic serial-port discovery, a work-area calibration wizard, and a local status panel with stop, reset, and progress information.

The highest-impact hardware improvement would be a repeatable cup holder with a simple height sensor. The highest-impact software improvement would be closed-loop route preview and scaling that automatically fits a selected design inside the available cup area. A final product-like version should also use a cleaner custom wiring board or enclosed controller assembly so that the system can be transported and demonstrated repeatedly.

12.3 Conclusion

The project accomplished an integrated latte-art drawing prototype with a functional image-to-path software pipeline, CoreXY planar motion, syringe-based liquid dispensing, and web-operated demonstration workflow. The finished prototype does not make coffee, but it demonstrates the intended automated art-printing function: a user can submit an image, generate a route, and command the machine to draw with liquid over a fixed cup area. Verification showed that the main motion, software, and interface requirements were satisfied, while dispensing quality remained the largest source of uncertainty.

References

[1] S. Jeon, *GRBL: An open source, embedded, high performance g-code parser and cnc milling controller*, 2025. Accessed: May 16, 2026. [Online]. Available: https://github.com/grbl/grbl.

[2] Arduino, *Arduino documentation*, 2025. Accessed: May 16, 2026. [Online]. Available: https://docs.arduino.cc/.

[3] CoreXY, *Corexy motion system*, 2025. Accessed: May 16, 2026. [Online]. Available: https://corexy.com/.

[4] Pallets, *Flask documentation*, 2025. Accessed: May 16, 2026. [Online]. Available: https://flask.palletsprojects.com/.

[5] fatedier, *Fast reverse proxy (frp)*, 2025. Accessed: May 16, 2026. [Online]. Available: https://github.com/fatedier/frp.