Journal of Applied Research and Technology

www.jart.icat.unam.mx

Journal of Applied Research and Technology 24 (2026) 575-584

Original

Design of a Low-Cost-Computer Braille System for Visually Impaired People Based on Open-Source Hardware

Sendey Vera-González* • Bremnen Véliz
Lidice Haz • Franklin López-Silva

Universidad Estatal Península de Santa Elena, Facultad de Sistemas y Telecomunicaciones, La Libertad, Ecuador

Received: 20 02 2025; Accepted: 17 08 2025
Available: 31 08 2026

Abstract: This paper describes the creation of a low-cost Braille printer using readily available components and open-source software. Existing Braille printers are expensive for many visually impaired individuals, particularly those with limited economic resources. This project aims to address this issue by providing both a functional printer and a practical construction guide. The system utilizes a CNC machine, controlled by a CNC shield board and an Arduino UNO microcontroller. The open source GRBL firmware, loaded onto the Arduino flash memory, translates G-code commands into Braille patterns. The open-source software Visual Studio and Braille G-code Generator are used to create the user interface that converts alphanumeric texts into instructions to control the two stepper motors. One motor is in charge of moving the sheet, and the second one for the displacement of the reel housing an electromagnetic coil (solenoid) that creates the raised Braille dots. The resulting printer achieves a maximum relative displacement error of 8% and a printing speed of approximately 1/36 characters per second. These results demonstrate the printer's ability to generate Braille text, thus increasing accessibility for the visually impaired community.

Keywords: **Braille printer, GRBL firmware, Arduino, CNC machine, Visual Studio.**

*Corresponding author.
E-mail address: svera@upse.edu.ec (S. Vera-González).
Peer Review under the responsibility of Universidad Nacional Autónoma de México.

1. Introduction

Access to education is a fundamental human right, yet individuals with disabilities and those from low-income backgrounds, often face significant barriers to inclusive learning. This necessitates differentiated educational support and access to specialized Information and Communication Technologies (ICTs) that foster independent learning. International organizations such as the United Nations Educational, Scientific and Cultural Organization (UN ESCO) (UNESCO, 2021) advocate for the development of accessible educational materials to ensure equitable access to education for all, including people with disabilities. In Ecuador, as of 2021, 2,625 students with visual disabilities were enrolled in basic, secondary and high school education. A significant proportion (40.58%) of these students were children between the ages of 4 and 12 years (Ministry of Education, 2021), highlighting the critical need for accessible educational resources. This work is motivated by the challenges faced by visually impaired youth in the La Libertad area of Santa Elena province, which include high rates of social exclusion (Mazzini Lindao & Carrera Rodríguez, 2018) and the lack of access to Braille printers for rent or purchase. For visually impaired individuals, a crucial tool for literacy is Braille. As illustrated in Figure 1, Braille code is based on a six-dot matrix, arranged in two columns of three rows. This system enables the representation of a wide range of characters, including letters, numbers, punctuation, mathematical symbols, and even musical notation.

The dimensions of the standard Braille matrix for writing Braille code characters are: 2.5 mm for the horizontal and vertical distances between dots of the same cell, the horizontal distance between cells (first column, cell 1 and first column, cell 2) is 6.0mm, the vertical distance between each cell is 10 mm, and the vertical distance between the first dot and the third dot of the cell is 5.0 mm.

The manual process of writing Braille systems is done character by character with a stylus and a classic slate, making it difficult, slow, and hindering learning. A more sophisticated method for writing Braille is the Braille printer. This system translates digital text into Braille, creating raised dots on a sheet of paper (thicker than the one used for printing in ink) using a series of hammers or blunt pins or combination with a print head to produce both ink and Braille (Dogs, 2023).

Braille printers use the mechanism called Computer Numerical Control (CNC). CNC is a system for precise

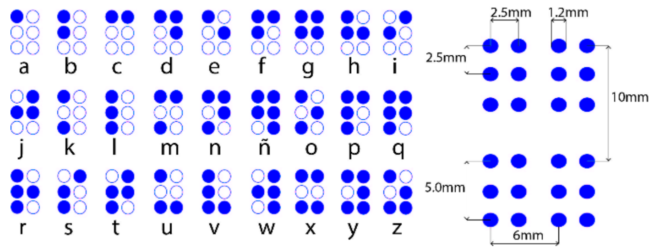

Figure 1: Braille Alphabet and dimensions ((CBE), 2024).

control of tools such as hammers, milling machines, grinders, lathes, drills, water jets, and lasers along the X, Y, and Z axes. CNC uses preprogrammed computer software written in a language called G-code, which is a numerical programming language of coordinate points with which you can establish parameters to operate the tools that include feed rates, spindle speeds (threaded bar), clockwise and counterclockwise rotation, tool changes, among other parameters, which serve to carry out a specific machining operation. A major drawback to widespread Braille literacy is the high cost of Braille printers. For example, a low-end printer from the Perkins brand costs around $1800 (Dogs, 2023), not including import costs. For that reason, researchers are developing inexpensive Braille modules such as a Braille communicator (Shubhom et al., 2017) that used a hex-keypad as an input interface or a printer with an Arduino Mega as the main controller (Gürevin & Güney, 2023).

To address the need for affordable Braille printing solutions, this paper details the design of a Braille printer leveraging the open-source GRBL firmware for CNC control and the readily available Arduino UNO hardware. This approach enables the creation of embossed Braille text on standard thick printer paper, offering a practical and easily accessible alternative for educational institutions and individuals that aims to significantly improve access to educational materials for visually impaired individuals. This work offers a novel approach to low-cost Braille printing by integrating CNC technology with GRBL firmware and solenoid-based embossing, achieving high positional accuracy and automated G-code generation from text input. Unlike previous designs that relied on less precise actuators or lacked full automation, this system combines hardware and software innovations that improve print quality, reproducibility, and accessibility. The inclusion of real-world validation with visually impaired users further reinforces its practical impact.

2. Materials and Methods

2.1 Materials

The physical structure materials are: A base designed in 3D in SolidWorks and made of 3 mm thick medium-density fiberboard (MDF), 3D-printed supports in PLA material, 6mm/400 mm steel shafts, 6.35 mm 20-tooth GT2 pulley, 6.35 mm to 5 mm flexible coupling, 6 mm GT2 timing belt, 6 mm LM6UU linear bearing.

Figure 2 shows the diagram of the main hardware components of the Braille printer, which are: the Arduino board, a CNC shield control board, the A4988 drivers, the stepper motors, a solenoid, and a 24 VDC power supply. Table 1 shows the electronic components used with their respective models.

Table 1: Hardware components for CNC control.

Amount	Component	Model
2	Stepper motor	NEMA17 42STH40
1	Solenoid	24 VDC
1	Arduino UNO	Atmega 328P
1	External Source	TPS-2405
2	Driver	A4988
1	Controller Board	Shield CNC
1	Transistor	TIP110
1	Diode	1N4002

The stepper motors are bipolar, operating at a current per phase of 1.2 A with a step angle of 1.8°. A motor controls the movement of the reel, while the other motor drives the roller that drags the sheet. The solenoid is an electromechanical actuator of linear movement that requires a 24 V power supply and is used to make the relief points. The solenoid circuit actuation was designed in the Altium Designer software. The Arduino UNO is a programmable device with an Atmega microcontroller with 32 KB internal flash memory, which is powered at 5 V and has a clock speed of 16 MHz. The external power supply, manufactured by EVL, delivers a continuous voltage of 24 VDC and a maximum current of 5 A to power the solenoid and the CNC shield V3 card. The A4988 drivers are used to control the rotation direction and step resolution of the stepper motors. These drivers are powered at 5 V and include a potentiometer to adjust the output current up to a maximum of 2 A. The CNC shield board is powered from the external source, it is used to supply power to the two stepper motor drivers and is controlled by the Arduino UNO. The TIP110 transistor manages power to activate

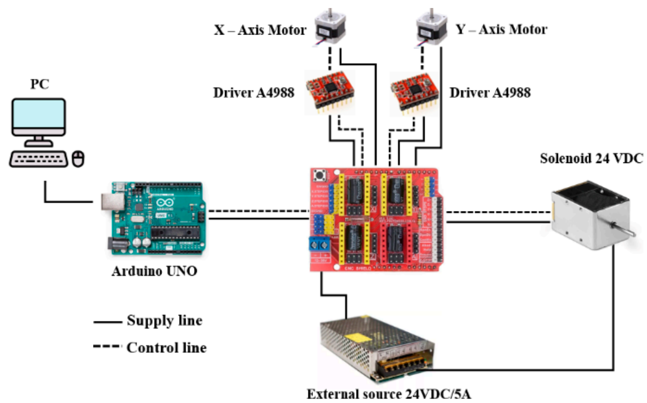

Figure 2: General hardware connection diagram.

or deactivate the solenoid (coil). The diode in antiparallel to the solenoid performs the function of protecting the transistor from the high back-electromotive force of the coil when the transistor is turned off.

2.2 Programming Method for the CNC Machine

The programming language of the CNC is the G-code. Using the G-code (Linux-CNC, 2024a) you can produce the movements of the CNC machines, such as where to start, how to move, and when to stop the actuating piece. On the other hand, the M-code (Linux-CNC, 2024b), is a set of auxiliary commands that allows the control of all non-geometric operations of the machine, such as stopping programs or changing tools. The software for supervising the CNC control is the Universal G-code Sender (UGS) version 2.1.2, which is available for free (Universal G-Code Sender, 2023). While the program to print Braille texts is Braille G-code Generator, the user enters the texts by keyboard from an interface of this program that has been previously configured. For Arduino programming, the open-source GRBL v0.9 library was used, which is downloadable from GitHub. First, the library must be added to the library from the program menu of the Arduino IDE, and then the example code called GRBLUpload must be compiled. The GRBL library includes the firmware for controlling the CNC machine.

2.3 Control Circuit Design

The control circuit consists of the Arduino UNO board that connects to the CNC shield, along with two A4988 drivers for each stepper motor, and the solenoid actuator will also be connected to the CNC shield. The CNC shield board is mounted on the Arduino UNO board making the 3V 3, 5 V and GND pins of both boards match. The actuator is a solenoid that performs the function of making the raised dots. Since the solenoid is activated with 24 V, a power conditioning circuit is required. The design of the power electronic circuit is based on a Darlington transistor to turn the solenoid coil on and of The solenoid coil is connected to terminal P3, as shown in Figure 3.

Transistor Q1 (TIP 110), which can handle a current of 2 A and power of 50 W, is used as a switch activated by means of a digital signal from pin D11 of the Arduino UNO, which is connected to resistor R1. The 220-Ohm value of resistor R1 limits the outgoing current of the Arduino to 20 mA.

A diode D1, which supports a current of 1 A, is connected between the external source and the collector of the transistor. This diode protects the transistor from the high voltage generated in the solenoid coil when the transistor switches to the cut-off state.

The detailed control circuit diagram is shown in Figure 4. The implementation is done by wiring between components as seen in Figure 5.

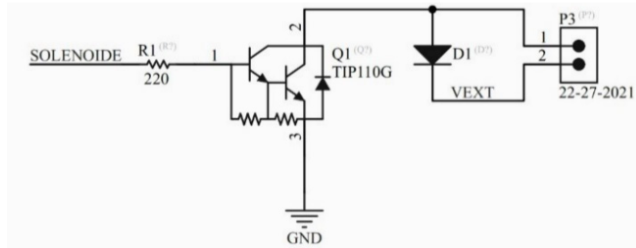

Figure 3: Power circuit for the solenoid.

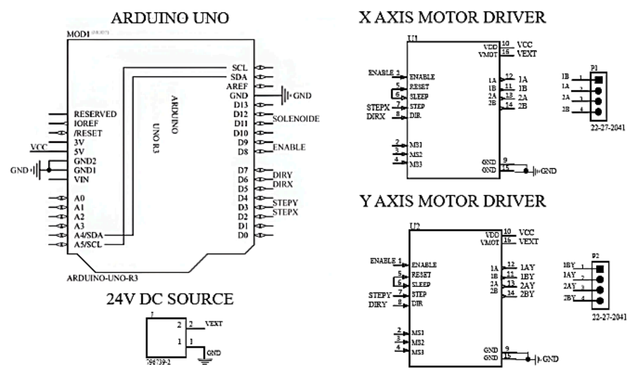

Figure 4: Schematic design of the control circuit.

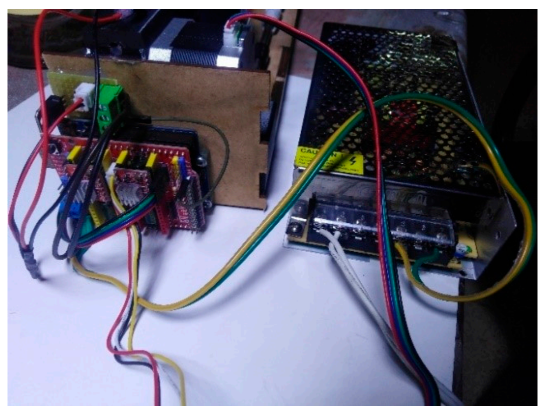

Figure 5: Implemented control circuit. CNC shield board (left) and voltage source (right).

3. Integration of Hardware and Software

3.1 MDF Base and Supports

The base of the structure was manufactured to hold all the components and allow the paper to move firmly. It has a length of 43 cm, a width of 10 cm, and a height of 2 cm. The profile cuts were made using laser cutting technology to have the precise dimensions obtained from the 3D design as seen in Figure 6.

The right-side bracket and the left-side bracket are shown in Figure 7. Each bracket has precise holes where the linear bearings on which the shafts for the reel are installed fit, as well as the holes where the timing belt pulleys are installed and the holes for adjusting the motors with screws.

A specific part is manufactured using 3D printing to contain the solenoid so that it can move parallel to the axes. The part has holes through which the linear bearings and the axes pass, and it also has a slot through which the timing belt fits (see Figure 8).

3.2 Stepper Motors

The stepper motor for the X-axis is located on the left side support. A GT2 pulley is attached to the motor and is used to mount a GT2 timing belt, which is responsible for moving the reel on the axes that support it. The movement of the motor produces the precise movement of the reel. The belt is mounted with a correct fit between the two ends (see Figure 9).

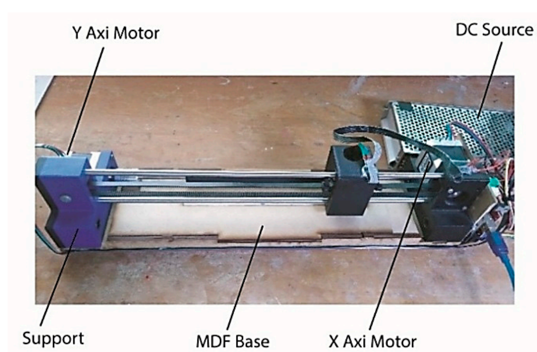

Figure 6: Top view of the Braille printer.

The second motor is installed on the right-side support and is connected to the roller shaft with a flexible coupling. This motor is responsible for moving the blade through the rotation of the rollers built into the shaft. The coupling prevents alignment problems between the motor shaft and the roller shaft, which helps to have a constant and stable movement of the blade on the base, as seen in Figure10.

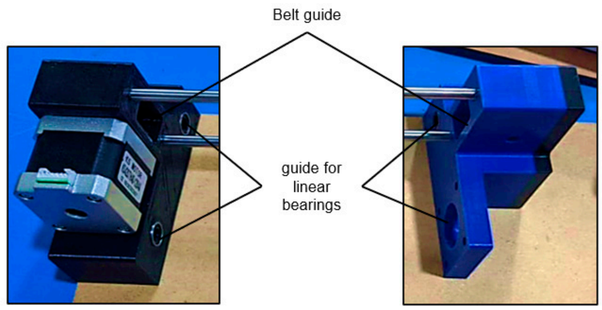

Figure 7: Left-side support and right-side support.

3.3 Calibration of Initial Parameters of the CNC Machines.

G-code Sender software is used to send commands from the computer to the CNC machine. This software is compatible with version 0.9 of the GRBL.h firmware, which runs on the Arduino board to load calibration parameters such as speed and step resolution of the motors. Our Braille printer has two movements on the X axis and the Y axis, the movement of the reel and the roller respectively, the Z axis only has the activation of the tool (solenoid).

In the parameters \$100 (x, step/mm), \$101 (y,step/mm), and \$102 (z, step/mm) the number of steps of the X, Y, and Z axes, required to move one millimeter is calibrated. The additional parameters do not undergo any change, they remain with the factory settings. The calibrated parameters are the following: \$100 = 4.796 (x, step/mm) \$101 = 4.782 (y, step/mm) \$102=250,000 (z, step/mm).

Finally, the offset values, or the zero values, are set to start the path. To establish these points, it is necessary to make a visual inspection of the Braille printer and define the point where the solenoid will start to puncture to generate the Braille characters.

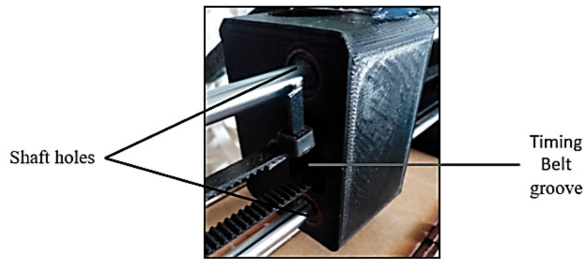

Figure 8: Solenoid structure.

3.4 Communication Between the Computer and the CNC Machine.

To print Braille characters, the Braille G-code Generator program is used, which is designed in C++ with Visual Studio to communicate with the CNC machine. In the program's interface, the user must select the communication port with the Arduino board (COM port). Text, numbers, or special characters are entered via the keyboard. By means of a button, the Braille characters are sent to be printed. Additionally, a window displays the G-codes generated from the entered text.

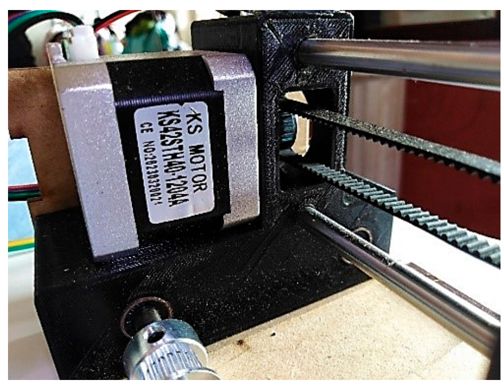

Figure 9: Engine installation on right support.

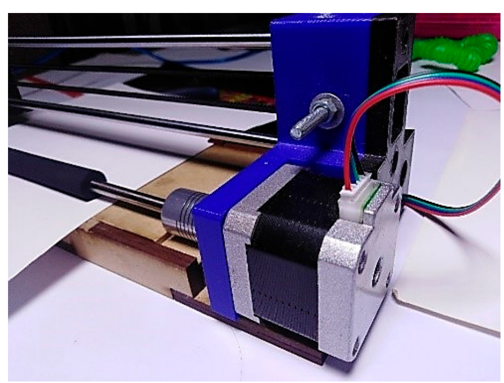

Figure 10: Engine installation on left support.

The flowchart algorithm for computer-CNC communication is observed in Figure 11. The communication initiates by creating a G-code list based on the Braille matrix dimensions. The X-axis reel starts at zero (xoffset = 0), and the initial Y position is 10 (yoffset = 10). Cell spacing is set to 6 mm (xIncrementalWord = 6), row spacing to 5 mm (yIncrementalRow = 5), and a limit of 29 characters per row (limitWordRow = 29) is enforced. The interdot distance within each Braille cell is 2.5 mm (increment = 2.5). Subsequently, the G-code parameters are configured (Figure 12). These initial steps constitute the core of the pre-programmed Braille G-code Generator interface communication algorithm (Figure 11).

Next, the algorithm enters a "foreach" loop (a special type of loop in C) waiting for the user to input text via the keyboard. Then, it verifies that the text has the Braille representation for each character using a collection data structure called a Dictionary. If the character's representation exists, it iterates over the character's Braille matrix. Then, the final coordinates are calculated based on the matrix elements equal to 1, resulting in the G-code coordinates that are stored in the G-code list.

Each letter has a matrix representation stored in the Dictionary. The letter is traversed with "foreach". Then the letter is linked to the corresponding Braille matrix. The matrix is extracted and traversed with a "for" cycle between the matrix indices, [6] = [0, 1, 2, 3, 4, 5]. To evaluate the indices, a Switch statement is used. It evaluates each case (0), case (1), case (2), case (3), case (4), and case (5). If the index is equal to 1, the value for finalX/finalY is assigned using the values obtained from the instructions for each case. Finally, the finalX / finalY values are added to the G-code list using the instructions shown in Figure 13.

These are the G-codes to generate the Braille character: the position of the G1 X/Y axes, activation of the M3 S1000 solenoid, G4 P0.2 pause, and shutdown of the M3 S0 solenoid. This is followed by the process of updating the positions after processing each character (xoffset += xIncrementWord), as well as limiting the number of words per row to 29 characters. When the limit is not reached, the positions are updated and -1 is reduced from the value of the limit of words per row (limitWordRow –).

When the word count limit per row is equal to zero, the position in X/Y is updated. The variable yoffset stores the value of the constant variable yIncrementRow (5.00) and adds the value of 10.00 that was assigned during initialization (yoffset += yIncrementRow).

The new value for xoffset is set to 0.00; returning it to its starting state and restarting the character count at 29 characters per row (limitWordRow =29). As long as the number of characters is equal to or less than 29, the process ends by sending the order for the motor to output the sheet with the printed Braille text.

When the number of characters exceeds 29, the first 29 characters are output first, which correspond to the

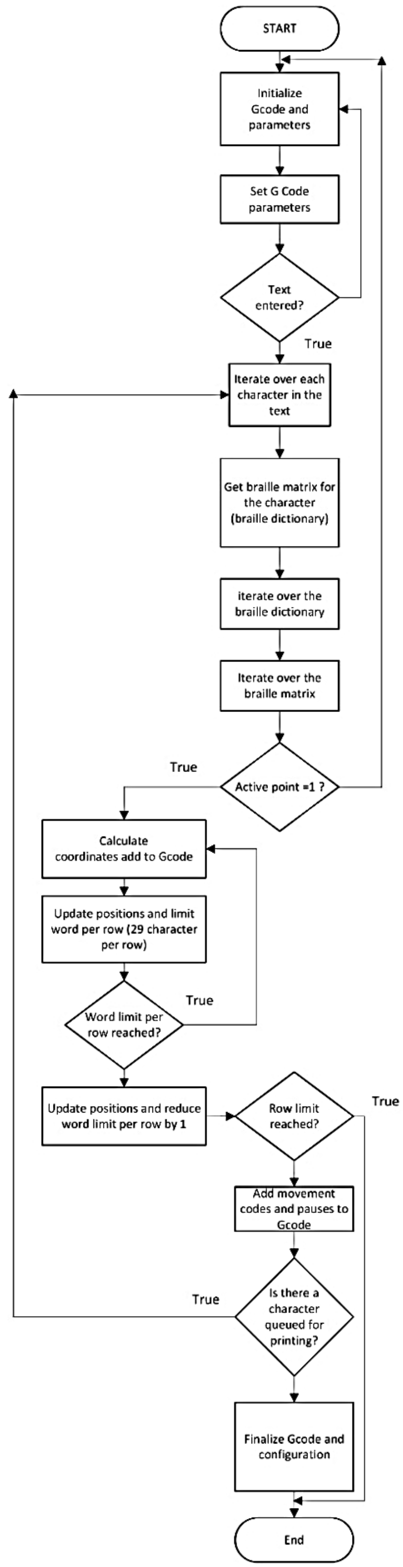

Figure 11: Flowchart (general Braille G-code Generator).

first row. The remaining characters are output in the following rows according to the text size. To output the remaining characters, the motors move simultaneously in a coordinated manner until the reel is in the starting position, and the Y-axis motor pulls the sheet until it is positioned 5 mm apart between the first row and the second row.

The code concludes with the assignment of the G-codes, to drag the sheet until it finally comes out of the roller. The total distance of the Y displacement of the roller is sent and is assigned a value of 310 mm, which is the value of the calculation of the sheet dimension and X = 0.00 to move the reel to the starting or resting point. Then the offset position is written at coordinate 0, expressed by the code L20 Y0. Finally, the X/Y position is reset to zero, and they are sent to the G-code list:

gcode.Add (“G1 X0.00 Y310.00;”);
gcode.Add (“G10 P0 L20 Y0;”);
gcode.Add (“G0 X0 Y0;”);
return string.Join (“”, gcode);

These last instructions are stored in the G-code list and sent to the Arduino via the COM port; the data transfer speed is 115200 baud, used in CNC machine communication. Upon receiving the G codes, the GRBL firmware installed on the Arduino interprets the instructions and translates them into motor movements and solenoid actuation to obtain Braille texts.

The complete design files, including 3D models, circuit schematics, and source code for the Braille G-code Generator software, are available for download at the following link (Vera et al., 2025).

4. Results

4.1 Test of the Stepper Motors

Two stepper motors were needed, which perform the function of moving the reel and the sheet in a coordinated manner. To generate the Braille matrix, the measurements in Figure 1 are used. The distance that the motor travels when taking a step is calculated by:

$$D = 2\pi rn(360°) \tag{1}$$

D is the distance traveled by the motor in one step, r is the radius of the motor pinion, and n is the number of degrees per step.

The spool travels via the motor on the X-axis, has an angle of 1.8 degrees per step, and the sprocket diameter

```
// parametros iniciales de GCODE
gcode.Add("G21;");
gcode.Add("G90;");
gcode.Add("G17;");
gcode.Add("G94;");
gcode.Add("G0 X0 Y0;");
gcode.Add("G1 X0 Y0 F1000;");
```

Figure 12: G-code initial parameters. C Visual Studio code.

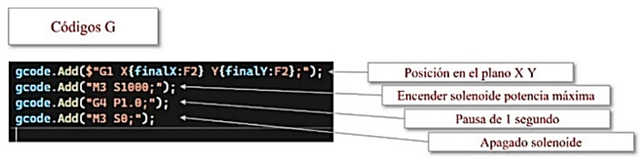

Figure 13: G commands to activate the solenoid.

is 12 mm (GT2 pulley). Applying equation 2, the distance the spool will travel is 0.188 mm per step.

The movement of the roller to drag the sheet is carried out by the motor on the Y-axis. This motor, according to its technical specifications, makes a 360° turn with 200 steps, the distance that drags the paper is taken experimentally to be 52.6 mm when making one turn. The roller therefore moves the sheet 0.263 mm. The number of steps, and the degrees for the Braille matrix are detailed in Table 2.

Table 2: Steps, degrees by distances traveled from Braille matrix.

Parameter	Distance	Steps	Degrees
X axis	1.0mm	5	9.0°
	2.5mm	13	23.4°
	6.0mm	32	57.6°
Y axis	1.0mm	4	7.2°
	2.5mm	10	18°
	10.0mm	38	68.4°

To check the displacement of the motors on the X and Y axes, a sheet of carbon paper was used, which was placed on the cardboard to mark the dots.

4.2 CNC Shield Controller Board

Before mounting the drivers, it was necessary to calibrate them in order to reduce the nominal current supplied to the stepper motors. The driver has a screw-like rheostat that must be turned to establish a reference voltage value. According to the data sheet (ALLEGRO-Microsystems, 2022) of the A4988 driver, this reference value is 0.96 V calculated using the equation:

$$V_{ref} = 8I_{max}R_S \quad (2)$$

Where I_{max} is 1.2 A, which is the stepper motor current per phase. R_S is 0.1 ohms, which is the A4988 surface-mount resistor. To perform the calibration, connect the driver via the VDD and GND pins to the 5 V and GND pins of an Arduino board, then measure with a multimeter via the terminals located on the GND of the driver and the positive terminal of the multimeter connected to the screw of the driver. In this way, the driver must be calibrated. The process is repeated on the two controllers. Once calibrated, they can be mounted on the CNC.

4.3 Test of Printing

The solenoid prints with a force of 3.5 N to form the embossed dots. The dot printing is done by programming the G commands to the Arduino to turn on, pause, and turn off the solenoid (see Figure 13); however, they were not enough to form the well-defined embossments. By making adjustments to the G-codes, it was determined that through four repetitions of turning on and off, the Braille dots with better relief are obtained. If a very high number of repetitions are added for turning on and off, the solenoid breaks the paper. In Figure 14 can be observed the printing of the alphabet in lowercase letters that has been entered by the Braille G-code Generator interface.

The Braille symbol measurements are checked by comparing the values obtained from the printing tests with the actual values observed in Figure 1. As can be seen in Table 3, the maximum absolute error does not exceed 0.3 mm and the highest relative error value is 8%, demonstrating good printing accuracy with the standard Braille dimensions.

Table 3: Errors measured in the distances traveled by the Braille matrix.

Detail	Distance	Absolute Error	Relative Error
A	2.5mm	0.2mm	8.0%
B	2.5mm	0.1mm	4.0%
C	6.0mm	0.3mm	5.0%
D	6.2mm	0.3mm	4.8%
E	10.0mm	0.2mm	2.0%

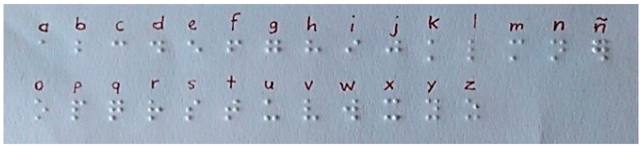

Figure 14: Lowercase alphabet printed in Braille code.

Where A is the horizontal distance between points in the same cell, B is the vertical distance between points in the same cell, C is the horizontal distance between cells, D is the distance between the first point and the third point in the cell, and E is the vertical distance between cells.

To test the printing time, a total of 696 characters were printed on a sheet of cardboard measuring 210 mm by 297 mm, distributed in 24 rows, where each row of printing consisted of 29 Braille characters. The printing times are shown in Table 4. The maximum time per sheet (696 characters) is one hour, ten minutes, and fifty-eight seconds. The time is quite high because there is only one punch (actuator) for the dots.

Table 4: Braille printing times.

Impression	Time (h:m:s)
A character	0:00:36
A row 29 characters	0:04:40
24 rows of 696 characters	1:10:58

The cost of implementing the printer was $186, not including labor, a similar value to the fast Braille printer proposed by Chowdhury (Chowdhury et al., 2018) or (Ismail et al., 2020), however, our printer has better accuracy, with a maximum relative error of 18% compared to the value standard format for Braille codes. While another low-cost printer, implemented with gear DC motors as a linear actuator, obtained a maximum error in precision of up to 100% as can be observed in Table 5.

Table 5: Comparison of Braille printing systems.

Detail	This Work	Chowdhury Work	Ismail Work
Arduino	Uno	Mega-Nano	Mega
Actuator	Solenoid	Linear Motors	Solenoid
Char/sec	1/36	1	Not
Unicode	Six dots	Six dots	Six dots
Size paper	A4	A4	A4
Char/line	29	16–18	Not
Cost	$186	$150	$57.53
Max error	2%-18%	50%-100%	Not

The printer's accuracy is slightly lower than that of commercial printers, but it is sufficient for comfortable reading. Future work will focus on increasing printing speed.

Finally, the printer was loaned to a Foundation specializing in education for children with visual impairments to be successfully evaluated due to the good legibility of the codes on the A4 Bristol card interpreted by blind people.

Conclusions

This project successfully demonstrates the feasibility of developing a low-cost, accessible Braille printer using CNC technology and open-source hardware. The implementation achieves several significant outcomes. First, by utilizing readily available components and G-code commands, the system provides accurate text-to-Braille conversion at a fraction of the cost of commercial solutions, making it more accessible to educational institutions and individuals with limited resources. The printer demonstrates acceptable performance metrics, achieving a Braille dot printing accuracy of ±0.3 mm (8% relative error), and a printing speed of approximately on characters every 36 seconds. User testing conducted through the Ambos Foundation confirmed that the output maintains sufficient quality for comfortable reading and practical use.

The project's significance extends beyond its technical achievements. By providing both a functional printer and detailed construction guidelines, this work contributes to the broader goal of improving accessibility to Braille literacy tools. The positive evaluation by the Melvin Jones Foundation particularly highlights the system's potential as an educational resource, offering new opportunities for teaching and learning among visually impaired individuals. In contrast to earlier designs that used lower precision actuators or did not offer complete automation, this system integrates both hardware and software enhancements that increase the quality of Braille output, ensure greater reproducibility, and enhance user accessibility. Moreover, its evaluation with actual visually impaired users highlights its real-world applicability and practical value.

Future work could focus on improving printing speed and further reducing the error margin, though the current implementation already represents a viable alternative to more expensive commercial solutions.

Conflict of Interest

The authors have no conflict of interest to declare.

Funding

The authors received no specific funding for this work.

Acknowledgements

We thank the members of the Monitoring group, the Faculty of Systems and Telecommunications of the Universidad Estatal Península de Santa Elena, and the Melvin Jones Foundation. We also thank the projects INCYT-PNF-2024-S76-185 and INCYT-91870000.0000.390370.

References

ALLEGRO-Microsystems. (2022). A4988 dmos microstepping driver with translator and overcurrent protection.
https://www.allegromicro.com/-/media/files/datasheets/a4988-datasheet.pdf

(CBE), S. B. C. (2024). Technical document b1. dimensional parameters of braille.
https://www.once.es/servicios-sociales/braille/comision-braille-espanola/documentos-tecnicos/documentos-tecnicos-relacionados-con-el-braille/documentos/b1-parametros-dimensionales-del-braille-2.

Chowdhury, D., Haider, M. Z., Sarkar, M., Refat, M., Datta, K., & Fattah, S. A. (2018). An intuitive approach to innovate a low cost Braille embosser. *International Journal of Instrumentation Technology*, *2*(1), 1-17.
https://doi.org/10.1504/IJIT.2018.090858

Dogs, G. (2023). Braille printer.
https://www.guidedogs.org.uk/getting-support/information-and-advice/how-can-technology-help-me/braille-printer/

Ecuadorian-Education-Ministry. (2021). Disability statistics.
https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/

Gürevin, B., & Güney, E. (2023). A Study on the Design of a Braille Printer Working on a Microcomputer. In *International Conference on Recent and Innovative Results in Engineering and Technology* (pp. 17-21).
https://doi.org/10.59287/icriret.1364

Ismail, A. M., Yuvaraj, K., Vigneswaran, S., & Nagarajan, P. (2020). Development of Low cost 6 Dot Unicode Braille printer for aiding visually impaired persons. In *IOP Conference Series. Materials Science and Engineering* (Vol. 995, No. 1). IOP Publishing.
https://doi.org/10.1088/1757-899X/995/1/012001

Linux-CNC. (2024a). G-codes.
https://linuxcnc.org/docs/html/gcode/g-code.html

Linux-CNC. (2024b). M-codes.
https://linuxcnc.org/docs/html/gcode/m-code.html

Mazzini Lindao, M. Y., & Carrera Rodríguez, M. E. (2018). Exclusión social de los jóvenes con discapacidad visual barrió 7 de septiembre del cantón La Libertad, período 2016.
https://repositorio.upse.edu.ec/handle/46000/4413

Shubhom, V. T., Keerthan, S., Swathi, S., Abhiram, G., & Shashidhar, R. (2017). BRAPTER: Compact braille transput communicator. In *2017 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)* (pp. 164-168). IEEE.
https://doi.org/10.1109/ICCE-ASIA.2017.8309319

UNESCO. (2021).
https://www.unesco.org/es/articles/la-unesco-hace-un-llamado-para-atender-las-necesidades-educativas-de-las-personas-con-discapacidad

Universal g-code sender. (2023).
https://universalgcodesender.com/

Vera, S., Véliz, B., Haza, L., & López, F. (2025). Braile1 - low-cost cnc braille printer project [Accessed: 2025-08-07].
https://github.com/Tecnotronick1/Braile1