A Digitally Automated Text to Braille Device for the Visually-Impaired

Jean Gozon[1], Apple Li[2], Andrea Odulio[3], Julianne Sadullo[4], Hiroki Asaba[5], and Clement Ong[6]
[1]De La Salle University - Manila
Jean Gozon: jean_gozon@dlsu.edu.ph
Apple Li: apple_li@dlsu.edu.ph
Andrea Odulio: andrea_odulio@dlsu.edu.ph
Julianne Sadullo: julianne_sadullo@dlsu.edu.ph
Hiroki Asaba: hiroki_asaba@dlsu.edu.ph
Clement Ong: clem.ong@delasalle.ph

Abstract: Convenience, portability and accessibility are key factors in the declining usage of Braille reading materials. While alternative devices such as screen readers, refreshable Braille displays, and mobile apps are being developed, the demand for better implementation of technology in said devices has yet to be met. These devices are often bulky, inaccessible, overly robotic and artificial, and also lack the required context which can cause stress to its users as well as multiple misunderstandings. With this in mind, a device was designed, with a 3 by 2 series of actuators, representing the "dots" in Braille, to raise the six pins under a finger-sized platform which represent a single Braille character. The coding and data transmission of the device uses Arduino Uno with specific components being 3D printed while the Braille code is based on Unified English Braille. The prototype is able to display Braille characters at approximately 180 words per minute.

Key Words: Visually-Impaired; Braille Reading Device; Unified English Braille; Arduino Uno

1. INTRODUCTION

Standard Braille books are often bulky and inconvenient to carry around. These are also often larger and bulkier than regular books due to the nature of Braille such as its spacing, its thickness, etc. In addition, Braille books are prone to the natural wear and tear of a book as it is used over time, causing the embossed pins to flatten and become less tactile. This means Braille books wouldn't be of use anymore, especially to the visually impaired Braille readers who heavily rely on their sense of touch to be able to read. As the advancement of technology continues to prosper, Braille books are also being produced in fewer numbers each year due to the digitalization of most literary works and academic sources.

In a 2012 report of the World Health Organization, 285 million were found to have visual impairments. Thirty-nine million of those 285 million people are completely blind while the rest have low vision (Santos, 2017). As for the visually impaired, the number one alternative for hearing is the use of tactile sense.

Braille is a tactile writing system developed in 1824 for helping the visually-impaired read (Roth, 2011). Characters, which include numbers, letters, and special characters, are represented by a series of dots which are embossed in a thick paper. Each character is represented by a certain pattern of a set of 2 x 3 dots.

It is often thought that the rapid advancement of technology in modern society eliminates the need for Braille. However, while technology such as text-to-speech software and audio-based toolkits produce human-like voices and could be easier to utilize, there is always a difference in terms of stress and intonation. In other words, it still lacks the complexity of naturally occurring speech (Kilickaya, F). Studies also show that compared to visually impaired students who only rely on hearing, Braille students have a higher literacy rate on average (Transforming Braille Charter Project, 2012). By learning how to read using Braille instead of relying on hearing, visually-impaired people can improve their spelling and literacy in language.

This study addresses the problems posed by traditional Braille books by providing a portable and convenient text-to-Braille device which can automatically translate digital text to Braille. This is a quantitative study which focuses on the effectiveness of the developed text-to-Braille device when compared to the average reading speed, 200 words per minute, of Braille readers (Ford & Walhof, n.d.). The study is also limited to tests which target the quantitative aspect of the device, such as its speed and accuracy.

2. METHODOLOGY

2.1 *Arduino Program*

An Arduino UNO board, a microcontroller, was used to serve as the main control in handling the program which will translate digital text to its corresponding Braille code and designates which specific actuators, that serves as the Braille pins, are protruded based on the projected Braille code.

Initially, the program gets an input text from the user. This input is to be converted into its six-bit Braille equivalent. The program starts off by checking if one of the input characters belong to the set of 'Special Characters'. Special characters include numeric characters such as 1,2,3, etc. and certain symbols such as question mark, ampersand, caret, etc. These Special Characters are composed of two sets of Braille code with the first one as the 'Special Character Indicator'. Identification of the number of Special characters in an input is necessary, so that the length of the output text can be adjusted to compensate the number of indicators.

Special Character Indicators are the preceding set of Braille code before the Special character which helps the user understand that the next set of Braille code is a Special character as there is a resemblance between some characters without the presence of the indicator which makes it important. Then, each character is converted to its six bit Braille code through a lookup table based on the Unified English Braille (UEB) code (PharmaBraille, 2015). This is then outputted through the digital output pins of the Arduino.

When the program starts, the Arduino will read the input from the first character. Two push buttons, which represent the forward and the backward buttons, are connected into the Arduino in order to let the user read back and forth throughout the text. Whenever either button is pressed, the Arduino pulses a digital output pin that is used to trigger the high voltage of the solenoid driver circuit. Triggering the high voltage of the solenoid driver circuit is needed to pull the Braille pins into the solenoids which creates an actuation process.

2.2 *Solenoids*

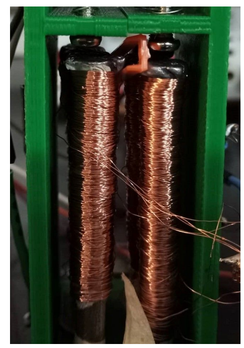

Figure 1. 2x3 set of Solenoids

Figure 1 shows the set of Solenoids which were made to serve as the electromagnet in the Braille device. This electromagnet will be able to pull the pins back to its initial position from rest through electromagnetism to produce an actuating mechanism in the Braille device.

The solenoids were made by coiling 39 AWG copper wires around a 6.3cm iron nails. To wind the copper wires around the nails, a stepper motor was programmed in order to accurately wind the solenoids with the desired number of rotations. This stepper motor is controlled by an A4988 stepper motor driver and an Arduino board (Nedelkovski, n.d.; see also How To Mechatronics, 2015, 9:34). The nails were attached unto the stepper motor via tape then the program to count the desired number of turns was uploaded to the Arduino board.

For the Arduino program, it first gets an input to how many rotations is needed. Then, it produces pulses based on the 1:48 ratio meaning 1 rotation is equal to 48 pulses before the actual rotation of the stepper motor begin. For this study, the stepper motor was programmed to make 2000 turns. As for the winding procedure, one end of the copper wire was taped unto the end of the nail before the program was uploaded to the Arduino. Then, the copper wires were manually guided along the nail making two layers of coil with 1000 turns for each layer. After the winding process, the ends of the wires were soldered to remove the thin insulation layer coating it.

2.3 *Actuator*

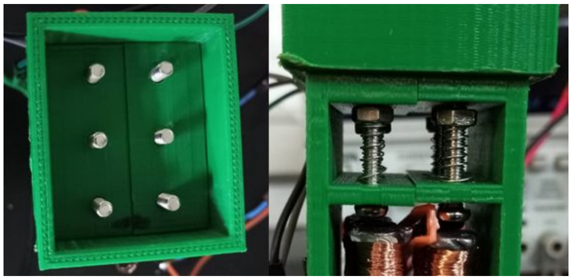

Figure 2. 2x3 set of Actuators as the Braille Pins

Upon successfully creating the component which would serve as the electromagnets for the actuator (i.e. solenoids), a 3D model case with two planes for the actuator and a base for the nail or the solenoid was modeled via FreeCAD and then 3d printed. This was done so as to test the success or failure of the actuator as well as to better visualize the device.

For the "Braille pins", six 3.7cm screw, nuts and springs were used. These materials allowed for the Braille pins to be easily adjusted. The screw was inserted into the drilled hole of the 3D model, with the tip serving as the Braille pin (Figure 2). The screw was secured by placing a spring below the nut in between the two planes. This facilitates the amount of force required to pull the screw down and how quickly it can be released. The distance between the screw and the nail can be adjusted by either loosening or tightening the nut. The ideal distance would be the distance that's small enough so the solenoid would be able to pull it down at high voltage without effort and a distance that's large enough that when pulled down, it would be noticeable from far away.

2.4 *3D Model*

The final model of the digitally automated Braille device's casing was modeled using FreeCAD.

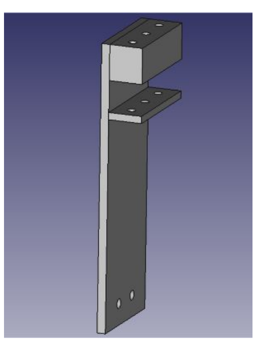

Figure 3. Braille Device Casing (Side)

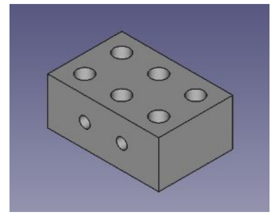

Figure 4. Braille Device Casing (Base)

Figures 3 and 4 show the final 3D design which will be used as the case of the Braille device. It is separated into two parts in order to facilitate the assembly and placements of parts such as the solenoids and the spring-tensioned pins.

After carefully taking the measurements of the solenoids and the actuators (e.g. length, width, depth, required distance, and circumference), a final verification of the taken measurements was conducted before printing the overall case for the entire device containing all six actuators and solenoids to ensure the accuracy of the measurements. After printing the case, the six actuators and solenoids were secured in place and final adjustments were done to guarantee the functionality of the actuators and the solenoids. Thus, making sure that there is enough distance between the electromagnets to avoid overlapping, and verifying if the holes in the case are loose enough so that the Braille pins can pass through smoothly avoiding any roughness between the transitions as much as possible.

After placing the actuators and solenoids in the case, it was found that the distance between the solenoids and actuators were lacking, so the length of the solenoids were reduced, and the holes for the actuators needed further drilling using a slightly larger drill for a smoother actuation. Finally, a roll of masking tape was used to secure the nails and gently hammering it down into each respective slots to keep it in place.

2.5 *Assembly*

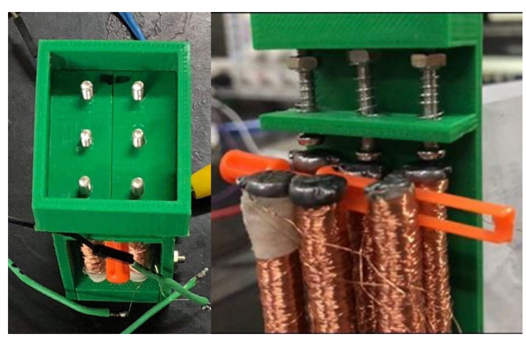

Figure 5. Braille Model Assembly

Figure 5 shows the Braille model of the final device. It is composed of six actuators, the 3D printed case, and six solenoids.

The six screws were used as a substitute in place of the actual Braille pins due to its flexibility and adjustability. All of the Braille pins were raised by default position when there is no power supplied. Once the DC power supply is powered, the pins will be pulled down by the solenoids and create the actuation. This is done by connecting the soldered tips of the copper wires to the alligator clips which is connected to the power supply. hus, supplying electricity to the solenoids.

The springs were placed in position, so the pins can be pulled down and revert back to its initial position when there is no power supplied to the solenoids. The uncompressed springs which results to raised pins serve as the dots in a Braille character.

In every actuator, a nut is placed above the spring to control the tension in it. Greater tension in the springs will result to a greater push force in the pins making it easier to fly back to its original position, but would require greater pull force from the solenoid. Likewise, a lesser tension in the spring will result to a lesser pull force from the solenoid making it easier to pull down, but will produce an unnoticeable actuation.

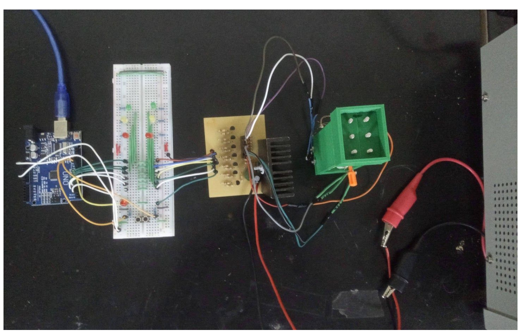

Figure 6. Complete Braille Device Setup

Figure 6 displays the complete setup for the Braille device. This setup comprises of a laptop, DC power supply, Arduino UNO, prototype board, solenoid driver circuit, and the assembled Braille device. as shown in Figure 5.

For the entire setup, a laptop containing the Arduino program is connected to the Arduino UNO device which receives the digital text input, translates it into ASCII text, and then translates it to its equivalent Braille code. The Arduino UNO is then connected to a prototype board displaying the six-bit Braille output through LED lights to assist in the verification of the projected Braille code output in the Braille device. The LED lights are switched OFF if the pins is supposedly actuated or pulled down and OFF if certain pins are left in its default position or protruded. The solenoid driver circuit, which regulates input voltage, is then connected to the prototype board. This part of the device regulates the voltage coming from the DC power supply connected to it before the electricity is supplied to the electromagnets within the actual device, causing the solenoids to pull the screws down, thereby forming the desired Braille character through the process of actuation. The solenoid driver circuit is connected to the DC power supply, and then to the actual Braille device.

2.6 *Solenoid Driver Circuit*

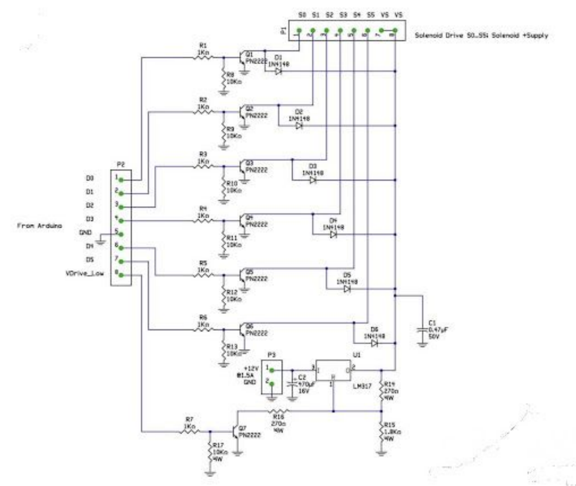

Figure 7. Braille Interface Circuit Schematic

Figure 7 shows the schematic diagram for the solenoid driver circuit responsible for powering the six actuators with the needed voltages.

Circuit input is given by the six-bit Braille output and a voltage drive pin. Each of the six Braille inputs are connected to the base of a 2N2222 NPN transistor. The emitter of each transistor is connected to ground, while a 1N4148 diode is connected between the collector of the transistor and the voltage drive output. The voltage drive output could be set to either high voltage (8.5V) or low voltage (2.5V). This is controlled with a LM317, an adjustable voltage regulator which requires two external resistors to set the output voltage. To switch the output voltage from high to low, a similar set up using the 2N2222 transistor is used to activate the resistors. For each solenoid, this voltage drive output is connected to one side of the coil, while the other side of the coil is connected to a transistor. The voltage was measured via VOM at high voltage is 8.5V and 2.5V at low voltage.

3. RESULTS AND DISCUSSION

Data is recorded to determine the electrical characteristics and response times of the device. In tables 1 and 2, each solenoid is tested to determine the minimum voltage needed to pull the pin, and the minimum voltage needed to maintain the attracted position of the pin. After determining the minimum required voltage to be able to pull all of the pins, the solenoids were tested at this voltage to get the amount of current each solenoid needs.

Table 1. Minimum Solenoid Pull Voltage

Pin number	Voltage
1	8.00 V
2	8.00 V
3	8.48 V
4	8.00 V
5	8.50 V
6	7.80 V

The data on table 1 show that each of the solenoids have different minimum voltage requirements when pulling the pin. This is caused by the distance between the nail and the screw. If the distance between the nail and the screw increases, the spring would then get more compressed causing the minimum pull voltage and minimum sustain voltage to go even higher. On the other hand, if the distance between the nail and the screw decreases, the spring would get more uncompressed resulting to a lower minimum pull voltage and lower minimum sustain voltage, but might increase the time it takes for it to go back to its original position. Based on these results, the minimum voltage required so that all of the pins will be able to be pulled is 8.5V, while the minimum voltage required to maintain the pull of the pin is a collective 2.3V.

Because the equivalent minimum voltage required is 8.5V in order to pull every pins, each solenoid is powered at this voltage to determine how much current each solenoids consume, results obtained are as shown in table 2.

Table 2. Solenoid Power Consumption

Pin number	Current
1	0.43 A
2	0.30 A
3	0.35 A
4	0.37 A
5	0.34 A
6	0.34 A

Table 2 shows that the current flowing through the solenoids also differ from one another. This is caused by the inexact patterns of the coils, varying distances between the solenoid and the screw, and the different sizes of the nail heads which contact the pins. Based on the results, the highest current that a pin requires is 0.43A consumed by Pin 1, while the lowest current that a pin requires is 0.3A consumed by Pin 2.

Overall, there are differences in the electrical characteristics between the pins due to numerous factors such as the distances between nail and screws, the compression and tightness of the springs, the inexact pattern of the coil in each nail, and the dissimilar size of the nail heads.

The next tests show the delay times that it takes for the device to change between Braille characters. Too long a delay time may disrupt the Braille reader's flow of reading, so it is best that this delay is as small as possible. Using an oscilloscope, the delay time between pressing the button and the powering of the solenoid is measured to be 84 µs. Then, using a slow motion camera at 240 fps, the time it takes for the pins to pull in, release, and stabilize were taken.

Table 3. Pin Pull In and Release

Pin number	Pull In	Release
1	12.5 ms	16.67ms
2	12.5 ms	12.50 ms
3	16.6 ms	16.67 ms
4	12.5 ms	12.50 ms
5	8.33 ms	20.83 ms
6	16.67 ms	16.67 ms

The data in Table 3 shows the time it takes for a pin to pull in and release. This was recorded using a slow-motion camera. The camera captures frames at 240 fps. Each frame is equivalent to 4.17ms. It takes longer for other pins to pull in and release because of the varying spring compression per solenoid.

Table 4. Total Time for the Pull In and Release of the Pins

Pin number	Total Time
1	29.17 ms
2	25.00 ms
3	33.27 ms
4	25.00 ms
5	29.16 ms
6	33.34 ms

Table 4 displays the total time for the pins to fully pull in and release. As seen in the data, pin 6 takes the longest out of all the pins. This is because pin 6 has the most compressed spring, making the distance from the screw to the nail bigger than the others. Pins 4 and 5 have the least total time because their springs were tightened perfectly. For Pin 5, the times for the pin to pull in and to release differs significantly because the spring was inadequately compressed. This can be seen in the quick time it takes for the pin to pull in, signifying the distance between the screw and the nail is small, and the slow time it takes for the pin to release, signifying the lack of compression of the spring.

Braille users read at an average speed of 200 words per minute. Those who start begin from childhood however, develop a reading speed as fast as 200 to 400 words per minute (Ford & Walhof, n.d.). If the time it takes for a pin to pull in and release is too long, it may also affect the flow of reading for users of the device. To reduce this, the time for a pin to pull in and release should be as short as possible to avoid disturbing the flow of reading of the Braille reading causing an uncomfortable feeling to the Braille users.

According to a study, the average reading speed of a Braille reader is about 200 words per minute (Ford & Walhof, n. d.). To calculate the reading speed of a Braille reader in words per second, the average reading speed of 200 wpm should be converted to its equivalent in seconds which is 3.33 words per second. Then, to calculate how many characters a Braille reader can read per second, supposed that there are about 5-6 characters in a word. The average reading speed of 3.33 wps is multiplied by the assumed average character count for one word which is about 5-6 words. This results to an approximate value of 20 characters per second. To get the speed at which a Braille reader can read one character per second, a second is divided by 20 which is the aforementioned number of characters a Braille reader can read in a second. Thus, this would generate an average Braille reading speed of 50 ms per character. As for the Braille device, the time of delay is computed by getting the average of the total pull in and release time of all the pins shown in Table 5 which will generate a 29.16 ms delay time. Comparing this to the data gathered, if the computed delay time is subtracted to the computed Braille reading speed, the delay time of the device would still leave about 20.84 ms left for the reader to feel the full character formed. This means that the users of the device would still have to read in a slower pace than what is usually done. The pull in and release of the pins also differ because of the varying spring compressions causing different tensions between the springs.

4. CONCLUSIONS

A prototype device that takes input ASCII text and produces a user-controllable sequence of Unified English Braille characters on a single 6x2 matrix has been developed. The system is capable of producing the full range of Braille characters, including so-called Special characters. Specially-wound electromagnets are used to actuate the Braille pins, and the whole system is capable of being operated from a 5V supply. Tests indicate that the system is capable of producing Braille patterns at a speed of about 180 words per minute.

Possible future development include specifically-manufactured electro-mechanical components, particularly the pins and electromagnet cores, will allow for a significant reduction in size. Fine-tuning of the pin force and height, by careful choice of spring tension and pin length respectively, will have to be undertaken to ensure that the Braille characters formed provide a comfortably felt, easy to read character for the user. A sliding encoder may also provide for a more intuitive interface, allowing the user the freedom to move his/her hand over a virtual Braille page. Such an input system would pave the way for the system to be interfaced to a computer where anything displayed, such as a browser newspage, could be instantaneously rendered in Braille at every character point on the page.

5. REFERENCES

Bochkarev, V.V., Shevlyakova, A.V. & Solovyev, V.D. (n.d.). Average word length dynamics as indicator of cultural changes in society [PDF]. Retrieved from https://arxiv.org/ftp/arxiv/papers/1208/1208.6109.pdf

Ford, S. & Walhof, R. (n.d.). Braille Reading Speed. Are you ready to do what it takes? Retrieved from https://nfb.org/images/nfb/publications/bm/bm99/bm990604.htm

How To Mechatronics. (2015, August 16). *How To Control a Stepper Motor with A4988 Driver and Arduino* [Video file]. Retrieved from: https://www.youtube.com/watch?v=5CmjB4WF5XA

Kilickaya, F. (2006, January 2). Text-to-Speech Technology: What does it offer foreign language learners?. Retrieved from http://callej.org/journal/7-2/Kilickaya.html

Nedelkovski, D. (n.d.). How To Control a Stepper Motor with A4988 Driver and Arduino. Retrieved from https://howtomechatronics.com/tutorials/arduino/how-to-control-stepper-motor-with-a4988-driver-and-arduino/

Norvig, P. (n.d.) English Letter Frequency Counts Mayzner Revisited or ETAOIN SRHLDCU. Retrieved from http://norvig.com/mayzner.html

Percentages of Letter Frequencies per 1000 Words. (n.d.). Retrieved from http://www.cs.trincoll.edu/~crypto/resources/LetFreq.html

PharmaBraille. (2015, November). Unified English Braille (UEB) Code. Retrieved from https://www.pharmabraille.com/braille-cdes/unified-english-braille-ueb-code/

Roth, G. A. & Fee, E. (2011). The Invention of Braille https://www.pharmabraille.com/braille-co des/unified-english-braille-ueb-code/. *Am J Public Health, 101*(3), 454. doi: 10.2105/AJPH.2010.200865

Santos, T. G. (2017, August 7). Doh Tells Pinoy: Avoid Blindness, Have Your Eyes Checked. *Philippine Daily Inquirer.* Retrieved from https://www.pressreader.com/philippines/philippine-daily-inquirer/20170807/281603830556394

Transforming Braille Project Charter. (2012). Retrieved from http://www.daisy.org/projects/transforming-braille

Wright, Wormsley & Kamei-Hannan. (2009). *Hand Movements and Braille Reading Efficiency: Data from the Alphabetic Braille and Contracted Braille Study.* Retrieved from http://www.afb.org/afbpress/pubnew.asp?DocID=jvib031008)