Proxicane:

a low cost, open source, wearable, proof of concept obstacle avoidance system for blind/partially sighted users.

Hussein Farkhani

Department of Computing

Goldsmiths, University of London

2018-2019

IS53007D: Computing Project Final Report

Project Repository: https://github.com/Hussein-Ben/Proxicane

Abstract

The Proxicane project is a proof of concept device that can provide low-cost object avoidance for blind/partially sighted people. The Project focuses on using the lower legs of a blind/partially sighted person rather than the hands and arms (which most other projects focus on) thereby freeing the user's hands and arms for other purposes. The Proxicane project is also intended to replace the cane a blind/partially sighted person typically uses. The entire project including the hardware and software is fully open source.

License

Table of Contents

Introduction 5
How does Proxicane Work? 6
What is ultrasound and how does the ultrasound sensor work? 8
What is the coin cell vibration motor and how does it work? 10
Why was haptic feedback chosen instead of sound or other outputs? 11
Why has the lower leg been chosen? 13

Proxicane .....
Parts list and technical specifications 15
Features 21
Limitations 23
Source code 26
Testing 28
Development process 34
Building instructions 39
Future Improvements 41

Before Proxicane – an attempt at an open source refreshable braille display 44
Specifications 49
Experiments 50
What if braille itself needs to be updated? 62

Conclusion 65

Appendix .....
Bibliography 67

Introduction

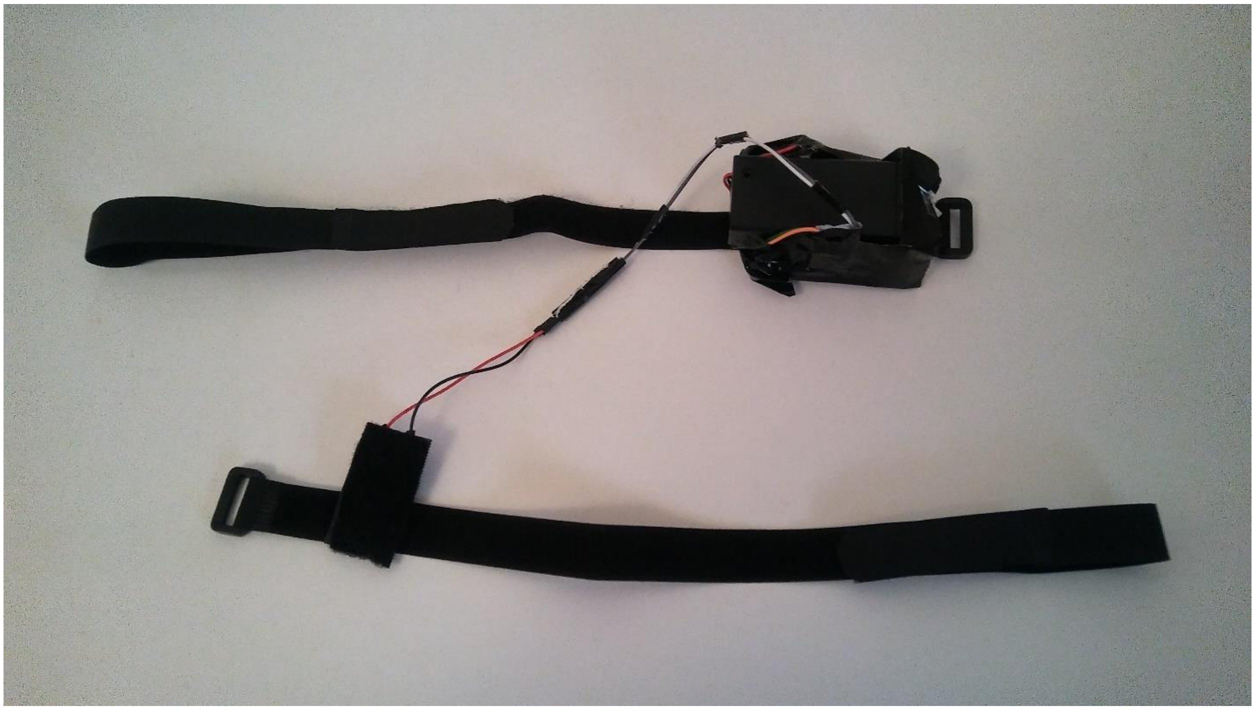
Typically, a blind/partially sighted person would use a cane to avoid obstacles while navigating a built environment. Depending on the level of impairment, different types of canes are used. All are visible and require the user to continually move the cane sideways as they walk.[1] Proxicane is an alternative to using the cane, it uses haptic feedback and ultrasound sensors to alert the user when they are near an obstacle, lowering the cognitive load of the user as this method is passive, the user only needs to move naturally. Proxicane will alert the user when they are near an obstacle. Although a concept, if mass manufactured and reengineered commercially, could provide a blind user with the independence to navigate a built environment without a need to interact with the built environment actively, they could move discretely (hitting a cane on a surface is noisy) and confidently.[2]

Most devices that provide object avoidance for the blind all involve the device being either attached to or held by the hand or attached to the arm. With this PoC I hope to demonstrate that focusing the device being attached to the leg would be more efficient for the user.

[1] https://www.rnib.org.uk/cane-explained
[2] https://www.bbc.co.uk/news/disability-34855311 cane users have issues with the cane hitting into people. A primary school in Bristol, England banned the cane over Health and Safety issues.

How does Proxicane Work?

A 9v battery powered Arduino has an ultrasound sensor connected to it, and a detachable coin cell sized vibration motor. Both the vibration motor has a Velcro fastening and a Velcro strap, the Arduino also has its own Velcro strap. This consists of one Proxicane unit. The vibration motor provides haptic feedback.



A single Proxicane Unit

For each leg, only 2 Proxicane units are required. One unit faces forward, and another unit will face 90 degrees to the side of the user, thereby providing full front and side collision detection. When a user moves a set distance near an obstacle (which is any hard/soft surface), the Arduino turns the vibration motor on, which vibrates, letting the user know that they are near an obstacle and if they move any further to it, they may collide.

Each unit faces 90 degrees to each other, this covers both Front and Side obstacles.

The Arduino continuously measures the surrounding distance via the connected ultrasound sensor.

If the user moves more than a set distance (adjustable by the user) from an obstacle, the Arduino switches off the vibration motor. Thus, the user knows they are *not* going to collide.

The detection distance is currently set at 30cm, but this can easily be modified by plugging the Arduino into any computer via a USB cable and changing the *collision_distance* variable, the program code is provided and is fully open-source.

One leg is sufficient for full frontal object avoidance with one Proxicane unit. However, both legs are needed (two Proxicane units per leg) for full front and side object avoidance.

What is ultrasound and how does the ultrasound sensor work?

Ultrasound is sound wave frequencies that are above 20,000 Hz, which is the highest frequency audible to the human ear. Just like sound waves, ultrasound can travel through gas, liquids and solids but not vacuums. Most importantly, Ultrasound waves travel in a very straight line (longitudinal). When ultrasound waves hit the surface of an object they are reflected, an ultrasonic sensor can detect the reflected wave.

Since ultrasound travels at a speed of approximately 340m/s through the air (at room temperature), the distance from a sensor to an object can be calculated by using the formula:

[3] $Distance(m) = speed(m/s) \times time(s)$

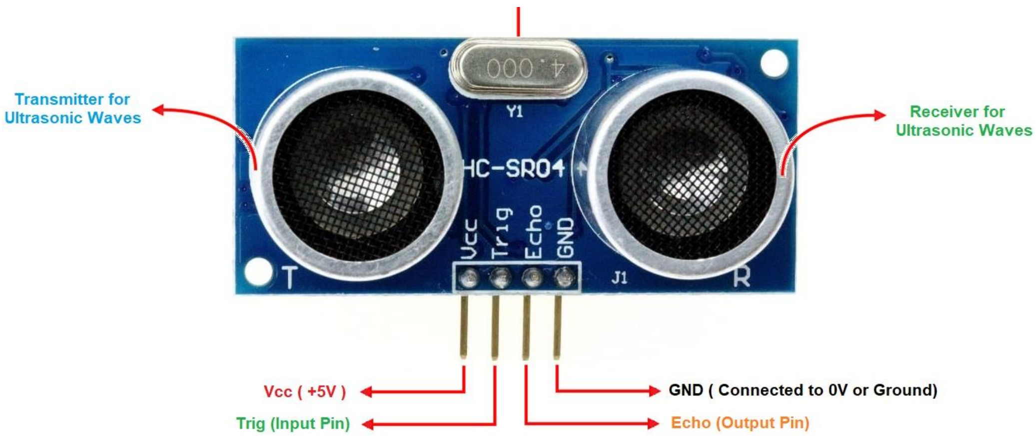


Figure 1 a HC-SR04 sensor, common cheap ultrasound sensor

An ultrasound sensor such as the HC–SR04[4] can use its two speakers which act as a transmitter and receiver; it can calculate the time it takes for a single pulse of ultrasound waves to be

[3] Diagram from: https://www.theengineeringprojects.com/wp-content/uploads/2018/10/Introduction-to-HC-SR04.jpg
[4] https://www.piborg.org/sensors-1136/hc-sr04

transmitted and received. With time calculated by the sensor (the sensor has onboard circuits that are designed for this), the distance can, therefore, be calculated by the Arduino.

What is the coin cell vibration motor and how does it work?

The coin cell vibration motor used in Proxicane is the same type of low-cost vibration motor generally found in modern smartphones.

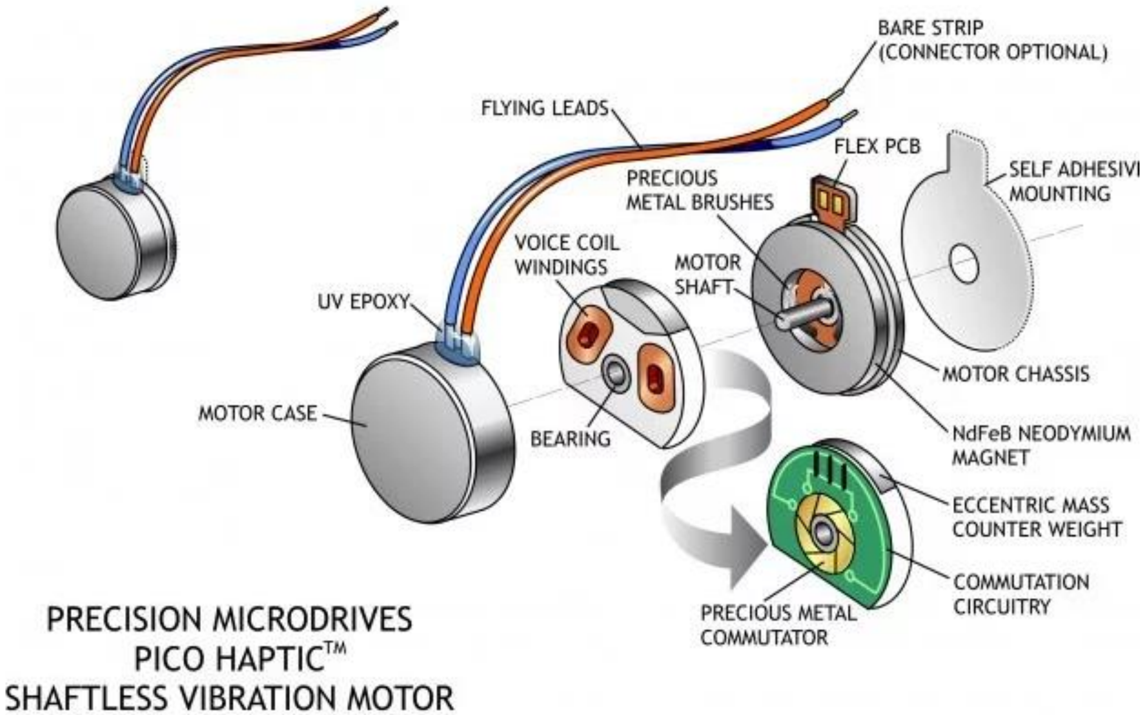


Figure 2A cross-section of a Vibration Motor, the same type used in a Proxicane unit. [5]

Coin cell vibration motors are small DC motors that spin an unbalanced weight. Spinning the weight creates vibrations that can be felt by the user. This is one of the cheapest methods to achieve haptic feedback.

---

[5] Diagram taken from: https://www.precisionmicrodrives.com/wp-content/uploads/2018/05/coin-shaftless-vibration-motor-exploded-view.original.jpg

Why was haptic feedback chosen instead of sound or other outputs?

Sound was considered as feedback to the user if they were near collision. However, there is an increased cognitive demand for the user because the user as well as using their hearing to navigate their environment, they would also have to focus on hearing a specific sound from the Proxicane device. By using haptic feedback instead, the user is free to focus all of their hearing on navigating a built environment (for example, listening for nearby cars).

Also, sound feedback would cause Proxicane to be less discreet for the user; it would draw unnecessary attention from a passer-by, which would reduce the user's confidence in navigating a built environment.

Moreover, a built environment is ordinarily loud, the sound output of the Proxicane would have to be loud just for the user to hear it outdoors, but then a severe problem arises if the user enters a quiet indoor environment where the loud sound from the Proxicane would not be appropriate.

Furthermore, the user in any built environment will come across many obstacles at a regular rate; a constant repeating sound will eventually become a nuisance to the user.

Haptic feedback avoids all of these problems, the vibration alerting the user is almost inaudible, it is direct and natural (haptic feedback is similar to being guided by another human by touch)

Due to the user being blind or partially sighted, light feedback cannot be used. Haptic feedback is the only form of feedback suitable in a built environment.

Although Proxicane can be attached to any limb or even equipment, why has the lower leg been chosen?

Other ultrasound location devices for the blind have all focused on the device being attached to the arm or hand. One of the purposes of Proxicane is to replace the cane so that blind and partially blind users can navigate more discreetly and with a lower cognitive load.

The cane has the role of sweeping what is in front of the user in a built environment, the endpoint of the cane which hits the lower part of any environment (such as the floor or lower part of a wall) is what provides haptic feedback to the user. By attaching Proxicane to the leg, the Proxicane ultrasound sensors would also replace this role.

Logically, the majority of obstacles a user will come across in a built environment would have to be “connected/attached” to the ground. So, sweeping the lower part of the environment using ultrasound would be more efficient in this situation. For example, all chairs and tables must have a leg that touches the ground, however, most chairs, tables and other objects do not reach a height that is taller than an average person, so a user could not detect a low table with a cane unless they swept with a cane and detected the table legs. So, unlike other collision detection devices for the blind which are attached to the arms and hands, it is more sensible if the device was attached to the lower leg.

Also, it is more discreet for the user and less to noticeable to others that the device is attached to the lower leg. The upper arms of the user are also free for the user to use in navigation and other activities.

Furthermore, the lower legs can only be used for walking or standing based locomotion activities whereas the arms and especially the hands of the user are needed for more finer control and touch. Proxicane being attached to the leg allows the user's hands and arms to be free to be used in other activities.

It is also apparent that the general public does not understand that there is a wide range of different forms of sight loss. For example, this user[6] can see without impairment in bright light, but at night or in slight darkness the user cannot see anything, the user would need to use a cane to walk at night, and they would be able to read any text in bright light, to an ignorant bystander the person may not seem blind, the blind person may be confronted and lose their confidence. This one of the reasons why a more discreet collision avoidance system, an alternative to the cane is needed.

[6] https://www.reddit.com/r/Blind/comments/99oigl/has_anyone_with_low_vision_been_harrassed_for/

Parts list and technical specifications

- **Elegoo UNO R3 Board (any Arduino UNO based board can also be used):**[78]

The Arduino project has fully open-sourced both its hardware and software, so similar cheaper boards such as the Elegoo will operate in the same manner as an official Arduino.

The source code of Proxicane will work with the official Arduino Uno and other Arduino based boards, so the user is free to choose any Arduino based board.

Average cost: £7.99 to £14.99

Microcontroller ATmega328

Operating Voltage 5V Input Voltage (recommended) 7-12V

Input Voltage (limits) 6-20V

Digital I/O Pins 14 (of which 6 provide Pulse Width Modulation output)

Analog Input Pins 6

2 electric current outputs from the 5V and 3.3V pin:

- 5V Electric current: 500MA
- 3.3V Electric current: 50MA

Flash Memory 32 KB of which 0.5 KB used by bootloader

SRAM 2 KB

EEPROM 1 KB

Clock Speed 16 MHz

---

[7] https://www.elegoo.com/product/elegoo-uno-r3-board-atmega328p-atmega16u2-with-usb-cable/ - the exact microcontroller used in this part of the Proxicane project

Working Temperature: -40°C to 85°C

Net weight: 25g

Dimensions:

- Length: 6.8cm
- Width: 5.3cm

The serial interface chip is different from the official Arduino Uno R3 board. The Elegoo board uses an ATMega16U2 instead of the ATMega8U2 chip.

Programmable using the official Arduino IDE, connect to a computer using a standard USB cable.[9]

[9] https://components101.com/sites/default/files/component_datasheet/Arduino%20Uno%20Datasheet.pdf – detailed technical specifications

- **Ultrasonic Distance Sensor (HC-SR04)**[10]

Average cost: £3

Input Voltage: 5V

Current Draw: 20mA (Max)

Digital Output: 5V

Digital Output: 0V (Low)

Working Temperature: -15°C to 70°C

Sensing Angle: 30° Cone

Angle of Effect: 15° Cone

Ultrasonic Frequency: 40kHz

Range: 2cm - 400cm accurate to the nearest 0.3cm

Net weight: 20g

Dimensions:

- Length: 43mm
- Width: 20mm
- Height (with transmitters): 15mm

Will require four pins on the Arduino.

[10] https://www.piborg.org/sensors-1136/hc-sr04

- **Flat Vibrating Vibration Motor**[11]

Average cost: £6-7 for five pieces

Material: Iron

Working Voltage: 2.5 - 4.0V

Output Speed: 12000RPM

Net Weight: 10g

Dimensions: 10mm x 2.7mm

Maximum start voltage is at an average 2.3v; this is important because without a voltage higher than 2.3v the vibration motor will not start. The unbalanced weight in the motor requires a high voltage to move it. Proxicane needs to be able to deliver power fast, for the haptic feedback to be useful to the user in real-world collision avoidance.[12]

[11] https://www.amazon.co.uk/gp/product/B079121QJ7/ref=ppx_yo_dt_b_asin_title_o07_s00?ie=UTF8&psc=1 the vibrating motor used in a Proxicane unit

[12] https://www.precisionmicrodrives.com/vibration-motors/coin-vibration-motors/ gathered from here, it was difficult finding detailed specifications.

- **Jumper wires**[13]

Average cost: £6 for 120pcs

Only eight jumper wires are needed to build the Proxicane

Jumper wires are used to build and prototype rapidly. The user who wants a more permanent solution can use any wire and solder the connections between components.

- **9v battery holder box with on/off switch**[14]

Average cost: £3 each

Dimensions: 6.8 x 3.3 x 2.1cm

Net weight: 10g

- **9v battery**

Can be found in almost any supermarket/ electrical store

The battery must be Alkaline, preferably a long-life battery due to the real-world application of Proxicane and long durations (16+ hours) that it needs to be operating.

[13] https://www.amazon.co.uk/Elegoo-120pcs-Multicolored-Breadboard-arduino-colorful/dp/B01EV70C78/ref=sr_1_3?keywords=jumper+wires&qid=1556015743&s=gateway&sr=8-3 jumper wires used in a Proxicane unit

[14] https://www.amazon.co.uk/HALJIA-Battery-Holder-Cover-Switch/dp/B071FJ14P2/ref=pd_rhf_ee_s_rp_0_4/261-1889573-1109536?_encoding=UTF8&pd_rd_i=B071FJ14P2&pd_rd_r=e46d7d97-3d7a-4e2b-9cd1-c9aa3cdda95e&pd_rd_w=0eNhj&pd_rd_wg=THG7r&pf_rd_p=be6d562a-9dbf-4e22-ab1d-83792c9c988d&pf_rd_r=CXM8FC77TZ4H3WQPYCDJ&psc=1&refRID=CXM8FC77TZ4H3WQPYCDJ this is what the Proxicane unit used

These are the essential core parts required for Proxicane to work. The user is free to choose the ideal case for Proxicane. Duct tape was chosen as this is the cheapest and lightest solution which provides high strength and waterproof ability. Velcro straps and stickers were used so that Proxicane can be mounted to the lower leg or any equipment.

The total average cost of a single Proxicane unit (without buying in a bulk order): £18.50

Features

- Fully open source design and source code. Any user can contribute suggested improvements or bug fixes after this project is complete; the repository of Proxicane will be made publicly available at https://github.com/Hussein-Ben/Proxicane
- A different approach to echolocation for the blind/partially-sighted users: a focus on the device being attached to the lower legs rather than the hands or arms.
- Focusing on the lower legs, the user is free to use their hand and arms to enhance navigation further or for other activities.
- Allows blind/partially sighted users to travel in a built environment more naturally, to react to obstacles passively, i.e. without having to use a cane to hit an obstacle or surface.
- An emphasis on discretion, a more discreet device draws less attention to the user, possibly increasing their confidence in navigation.
- Only requires a 9v battery; future work on Proxicane will reduce the power consumption.
- All components are low-cost and easy to obtain or even salvaged (vibration motors can easily be found in disused smartphones) There are better sensors such as laser range finders, but this was avoided as it was expensive (above £30 for a single sensor)[15]
- Low cost allows rapid changes to made to the design without fear of financial burden.
- Simple to operate, the user only turns on the device via a switch and attaches the device to lower leg, they can start navigating immediately.
- Changing the object distance detection can be done by editing the source code, the Arduino only needs to be plugged into a PC.

[15] https://www.amazon.co.uk/MakerHawk-Single-Point-Ranging-Pixhawk-Compatible/dp/B0778B15G7

- Accessible source code, only 60 lines of source code which is well commented. With little coding knowledge, anyone can understand the entire source code of Proxicane.
- Haptic feedback, more discrete than a cane (which must by design be loud, as the user hits/sweeps across surfaces), if the user has deafness or reduced hearing haptic feedback would not impair the navigation abilities unlike using a cane.
- Ultrasound is not affected by environmental lighting, can work in any lighting condition, unlike a laser range finder or any other sensor that relies on light.
- Haptic feedback provided by the vibration motor is the same feedback used in modern smartphones, Users would most likely be familiar with the vibration provided by the motors.
- Designed with the smallest number of wire connections to build a working unit, all unnecessary connections are kept to a minimum, this is to ensure that in the future it will be easier to scale down the physical size and power requirements of a Proxicane unit.
- Low cost, cheaper than a cane. Standard canes on average cost £ 20-24.[16] Even without further optimisations and without buying parts in bulk, Proxicane on average costs £18.50.

---

[16] Examples of current cane prices:

- https://shop.rnib.org.uk/mobility/canes/100cm-folding-guide-cane-pencil-tip.html
- https://www.amazon.co.uk/White-Aluminum-Mobility-Folding-Sections/dp/B019LOFY5G/ref=sr_1_2?keywords=cane+for+blind&qid=1556033970&s=gateway&sr=8-2

Limitations

- Different surface materials reflect ultrasound waves differently; this affects the accuracy of the ultrasound readings. For example, fabric materials will absorb some of the ultrasound waves reducing the accuracy of object detection (distance to detect an object will be lower). Solid flat surface typically reflects best.
- Due to obstacles having different materials, the detection distance for different obstacles will not be uniform. For example, a sofa will be detected slower than a closed wooden door, because the material of the sofa will absorb ultrasound waves more.
- Each Proxicane unit has its own haptic feedback end. When two Proxicane units are attached to a user’s leg, the haptic feedback ends need to be at least 15cm away from each other; this is to ensure that the user is clear which unit detected an obstacle, the user needs to differentiate between two different vibrations.
- The problem with multiple haptic feedback outputs on the body is that a user could mistake or misinterpret the vibration emitting from the haptic feedback end.
- Some users may be sensitive to haptic feedback, although no research can be found to support this. It is unclear what the effects of long-term haptic feedback is, does the user have reduced sensitivity to it? Does it distress the user?
- Proxicane is still a concept; it is not intended to be used in day to day real-world collision avoidance yet.
- Not suitable for accurate object collision detection, designed for day-to-day movement at a walking pace. Should not be used when cycling or running. Proxicane at this stage cannot support individuals who do sports or lead more active lifestyles.

- To keep costs low as possible (ensuring accessibility), only cheap ultrasound sensors can be used, it is unclear how long these sensors can endure in real-world situations. More testing is needed.
- Collison detection is not “smart” if the user was talking to a person in front of them, Proxicane would register this as a Collison, there needs to be functionality added that allows the user to determine if a collision is a false alarm. However, even this functionality would become annoying to the user, the vast number of false collisions the user will come across in a typical day, it is clear that a machine learning model would be needed. A machine learning model can be trained to recognise when collisions are false, thereby reducing the load on the user. In summary, Proxicane needs artificial intelligence, specifically machine learning; adding accurate location data that can be stored is also needed for this approach.
- The current power consumption and battery life of Proxicane does not allow all day collision detection; approximately 16 hours of battery power is needed. Proxicane can only provide up to 2 hours of battery life.
- Jumper wires allow rapid changes to be made to the design but are not ideal as a permanent solution as the wires can become loose. Soldering connections together is another feature that users will be encouraged in further iterations of Proxicane.
- Changing the detection distance requires the user to have access to a computer; it currently cannot be changed immediately. Proxicane needs to be completely standalone. Users should be able to change the detection distance on the device rapidly.
- Climbing up a flight of stairs will cause Proxicane to continually vibrate as it would assume that there is a Collison in front of the user. The user needs a way to shut down

temporarily, any false readings. Currently, the user can switch off Proxicane manually via a switch on the battery pack.

- Two units are required for each leg to achieve full side and front obstacle avoidance; this is too costly. One unit for each leg is what is needed.
- Proxicane will provide haptic feedback when in range of an object but it will not be able to provide feedback on the type of object that is being collided, for example, the user would not know if the object is a human or a piece of furniture.

In terms of intelligence of the Proxicane device, having real-time intelligent object detection (for example, recognising stairs etc..) would require processing power that not even the high-end tier smartphones of today could provide. When Augmented Reality becomes a regular part of the general population lives, it will require full 3D mapping, both indoor and outdoor; this data can be used to provide genuinely accurate collision avoidance as well as object detection. There is already progress in this field.[17]

[17] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4732050/

Proxicane source code

Programs that provide basic functionality for setting up an Ultrasound sensor[18] was needed as a base to start development from as there is only one standard way of operating an Ultrasound sensor (this is typical in Arduino projects). Total source code memory size is 3kb. There is no use of external libraries, so there are no dependency issues.

```
int trigPin = 9;    // Trigger - ultrasound waves beamed out
int echoPin = 8;    // Echo - listening for the wave that was beamed out
int vibrate_power = 10; // pin that supplies power to the vibrate function
int collision_distance = 90;  // the distance from an object, the user could collide with

long duration; // time in microseconds since firing a pulse and receiving a pulse
long distance; // continuously measured - this value is not 100% accurate, but it is sufficient for object avoidance

void setup() {
  //Serial Port begin
  Serial.begin (9600);

  //Define inputs and outputs
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(vibrate_power, OUTPUT);
}

void loop() {
  // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);

  // The sensor is triggered by a HIGH pulse of 10 or more microseconds.
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the signal from the sensor: a HIGH pulse whose
  // duration is the time (in microseconds) from the sending
  // of the ping to the reception of its echo off of an object.
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
```

[18] https://raw.githubusercontent.com/RuiSantosdotme/Random-Nerd-Tutorials/master/Projects/Ultrasonic_Sensor_HC-SR04.c used as a base

```

  // Convert the time into a distance
  // distance = (time/2) x speed of sound
  // The speed of sound is: 343m/s = 0.0343 cm/uS

  distance = (duration / 2) * 0.0343;

  // duration is divided by two first because the wave was sent,
  // collided with the object, and then returned to the echo sensor

  Serial.print(distance);
  Serial.print("cm");
  Serial.println();


  if (distance < collision_distance ) {  // warn user there going to collide with an object
    Serial.println("collision");
    digitalWrite(vibrate_power, HIGH); // physically warn user
    delay(100);
  } else { // stops false reading
    digitalWrite(vibrate_power, LOW);
  }
```

Only basic loops and if statements have been used to avoid any runtime errors and to increase simplicity. The source code only needs to be loaded onto the Arduino once, updates are not expected to the source code, through use there has been no bugs discovered. If updates are needed the source code will need to be loaded onto Arduino again using a USB cable and a PC.

Testing

Proxicane testing criteria:

Note: each Proxicane device with one haptic feedback end is referred to as a unit.

1. Drop test: Can a Proxicane unit function after a drop of 1.5 meters?
2. Detection distance: Can Proxicane detect obstacles (walls, people) within 1 metre?
3. Detection reaction time: how long does it take for Proxicane Unit to provide vibration feedback to the user?
4. False readings: what kind of items and surfaces in the real world that can create a false reading for a Proxicane unit?
5. Impact test: how much force can a Proxicane unit withstand before failure? How much pulling force can the wires endure?
6. Humidity and waterproof, water splash test: Can a Proxicane unit operate in a humid environment accurately, does humidity and water damage affect the accuracy of the ultrasound sensor? A Proxicane unit is generally attached to the leg, will sweat cause any issues?
7. Temperature: highest and lowest temperature a Proxicane unit can operate. Does temperature affect the Ultrasound sensor readings?
8. Does wind affect ultrasound ability?
9. Do loud sounds in the environment interfere with Proxicane?
10. Interference: do other ultrasound sensors and electromagnetic emitters affect the Proxicane ability to detect collisions?
11. UX: general user experience with Proxicane, the difficulty level experienced by the user. Does the user need the training to start using Proxicane?

12. Battery life: in the most demanding use how long is the Proxicane battery life expected to last?
13. User movement speed: what is the fastest speed a user can move before the Proxicane unit object detection ability is affected?
14. Object avoidance: are users avoiding collisions with objects? The efficiency of avoiding objects in different types of built environments.
15. Source code: are there any compilation errors or runtime errors? Any unused variables?

Tests carried out against criteria:

Proxicane has not been tested with blind/partially sighted users yet, but it has been tested with users who were blindfolded to block out their vision.

1. The Proxicane unit dropped from a height of 2 metres; is fully operational. Object detection still works as normal. However, after two drops the battery covers fall off. Suggested Improvements: currently as a proof of concept duct tape is used, the battery compartment needs to be less exposed. The user is encouraged to design their form of casing.
2. The source code variable for detection distance is inaccurate by an average of 100% when applied to real-world object detection. For example, if the variable for object detection distance is set at 100cm (so if a user is less than 100cm from an obstacle), in real-world applications the actual distance a user needs to be before Proxicane alerts them is 50cm. This is due to the different materials ultrasound waves reflect off, and the low cost of the Ultrasound sensor, more expensive laser range finders avoid this issue. A user wanting to be alerted to obstacle a metre away would need to set the variable to 200 cm.
3. Reaction time: Proxicane reacts to a Collison in under 1 second. A Collison is defined as a Proxicane unit detecting the distance calculated by the ultrasound sensor as less than 100 cm (or any value defined by the user in the source code). Under 1 second is sufficient at a walking pace in a built environment without moving obstacles such as people. More testing is required to see if the reaction time is sufficient in built environments that have large crowds of people moving around a user.
4. False readings occur only if an object is placed in front of sensor momentarily, for example, waving a hand in front of it. The problem is that Proxicane does not consider

the thickness of the material, so if a piece of paper or even a leaf is in front of the sensor, Proxicane will detect this as a Collison.

5. Five hundred grams of weight can be applied to the wires without any connection break. However, reinforcement is needed by having all connections soldered. Currently, Proxicane does not have a case to protect the Arduino. Without a case at the current state of this project, Proxicane should not be used for mission-critical object avoidance.
6. Proxicane will not work and be fully damaged if submerged in water. Cannot be used for obstacle avoidance in water or near water (walking beside a beach tide for example). As a proof of concept, I covered a Proxicane unit in duct tape to secure it. Duct tape is waterproof, but because the ultrasound sensor emitters and receivers need to be exposed in order to work, Proxicane cannot be splash proof. Research needs to be done to find a way to make low-cost ultrasound sensors waterproof.
7. Speed of sound is temperature-dependent and changes by approximately 0.17% with each degree Celsius according to the Pepperl+Fuchs a company that specialises in electronic sensors[19] Proxicane is confirmed to fully operate in temperatures ranging from 10°C to 27°C, the only temperature range possible for testing with the resources I have. The Arduino has a working Temperature between -40°C to 85°C, but the ultrasound sensor has a working Temperature between -15°C to 70°C. The required resources needed to test working temperatures to the extremes is not available. However, it is safe to assume that Proxicane will have no problems working indoors or in average weather conditions

[19] https://www.pepperl-fuchs.com/great_britain/en/25518.htm according to the Pepperl+Fuchs a company that specialises in electronic sensors

outdoors. The temperature must be compensated for; a suggested improvement is that Proxicane now needs a temperature sensor as well.

8. Using a hairdryer to simulate wind near the ultrasound sensors has not shown any noticeable differences to the readings, this is the most stress that can be simulated. Wind speed[20] below 50-61.5 km/h should not affect ultrasound readings.
9. Proxicane has been placed in loud environments of 120 decibels and operated as usual.
10. Other ultrasound sensors, if in the line of sight of a Proxicane unit will cause severe interference and produce false readings.
11. Proxicane is still a proof of concept; the casing is not available yet. Currently, the only input to the Proxicane unit is an on/off switch on the battery pack. The only method that a user can modify the detection distance is via connecting the Proxicane unit to a PC using a USB cable. A short training period is required in order to explain what Proxicane is to a new user. Proxicane user experience is not intuitive nor is it ready. Many improvements are needed; the Proxicane unit needs to be smaller with a hard case. There needs to be buttons on the case that allow a user intuitively adjust collision detection distance immediately. A user also needs to alert the unit, a temporary sleep button to stop false readings when climbing stairs for example. Priority needs to be given to making a Proxicane unit standalone, requiring no PC for modification to Collison distances. Ideally, Proxicane needs intelligence to adapt to a user's current situation, the processing power and cost required for this is the main setback.

---

[20] https://www.pepperl-fuchs.com/great_britain/en/25518.htm

12. Battery life was tested whereby the Proxicane unit was always in a state of collision; the haptic feedback end was continually vibrating. The Proxicane unit lasted for a maximum of 2 hours.
13. The Proxicane unit has been tested to successfully detect objects when a user is at a walking pace in a non-crowded built environment. In a crowded environment, Proxicane will most likely alert the user more than three times per second as they move through a crowd. More testing is needed with blind users to see if this level of alert is annoying or satisfactory to the user. At a running pace, Proxicane is not ready for use. More testing is needed at a walking pace in crowded environments before running can be explored.
14. Through testing, at a walking pace, objects can be avoided, but Proxicane should not be the only source of object avoidance that a user should rely upon. Users should of course also use their sense of hearing, smell, touch (for example, bus stop buttons in braille) and sense of place (for example, a user generally has a mental map of the place they have been before)
15. Source code is kept as simple as possible; there have been no compilation errors or bugs detected. No unused variable or dependency issues. The source code of Proxicane does not rely on any external libraries. As the source code is hosted on GitHub, automatic vulnerability alerts have been enabled[21].

[21] https://github.blog/2017-11-16-introducing-security-alerts-on-github/

Proxicane development process

Development focused on experiments with as many different sensors; these sensors are the only inputs that can possibly detect a Collison. When deciding which sensors to use, only the cheapest were experimented with. Proxicane needs to be low cost in order for it to be accessible.

The initial concept was to use the smartphone sensors to detect and warn the user of Collison with objects. Experiments were first carried out with ambient light sensors, that all modern smartphones have. Currently, there are more than 4 billion smartphones on earth[22] , and most people around the world have access to a smartphone, so this is the most accessible hardware.

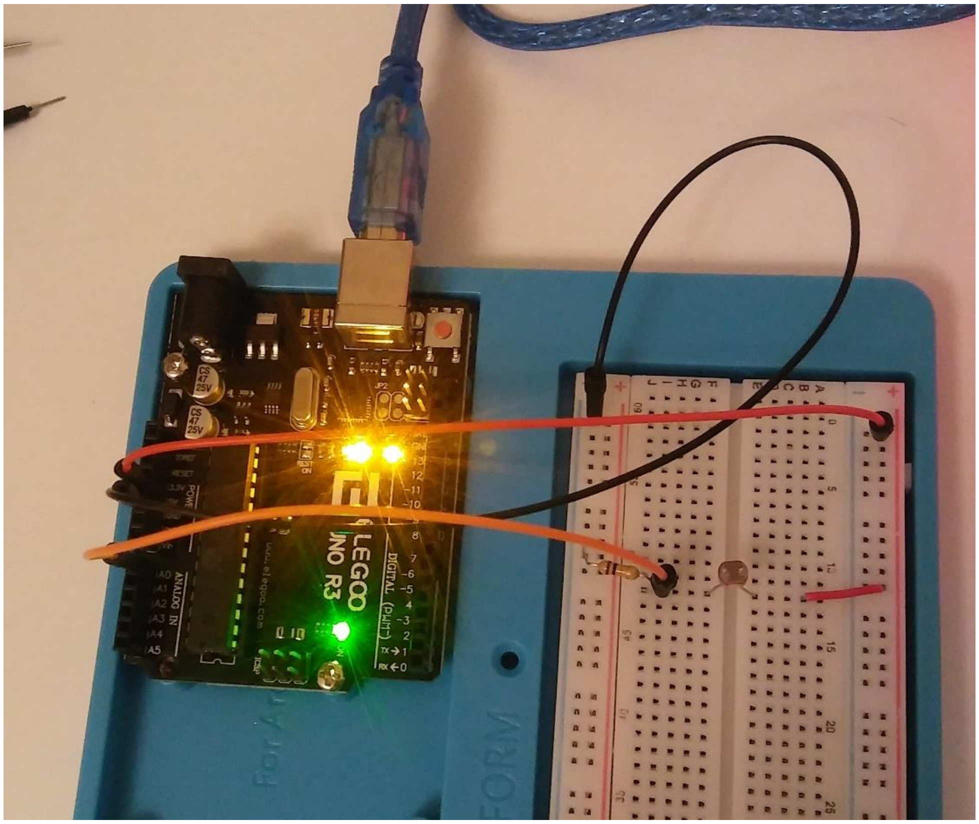


Figure 3 Experiments with a raw ambient light sensor

[22] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

What was discovered is that ambient light sensors can only detect a significant difference in light (a reduction of 30-40 lux) when the sensor is placed 10cm away from an object, this is in a well-lit room in daylight. The problem with ambient light sensors is that it would not be able to detect significant changes in light from a near obstacle in a dark room or at night. This is the only suitable sensor available on a smartphone; therefore, this pathway was disregarded.

Experiments were also carried out using the ambient light sensors using the HTML 5 ambient light event API[23]. The results were similar to the experiments carried out previously. However, the HTML 5 ambient light API has been disabled by every web browser except Microsoft edge, even with Mozilla Firefox, ambient light events have to be manually enabled. There are two reasons why it was discontinued. Firstly, from telemetry data, the API was not being actively used by developers probably due to smartphones and tablets having their own ambient light API (from either Android or iOS SDK). Secondly, there are many security issues with ambient light sensors[24]. For example, light readings from the ambient light sensor can reveal information about the user's environment to the extent that even bank PINs can be discovered as demonstrated in this research paper[25].

[23] https://w3c.github.io/ambient-light/
[24] https://www.w3.org/TR/ambient-light/#security-and-privacy
[25] Spreitzer, R. (2014). PIN Skimming. *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices - SPSM 14*. doi:10.1145/2666620.2666622

As a countermeasure sensor reading will intentionally be made less accurate, thereby it will not be fit for purpose in the near future for Proxicane.

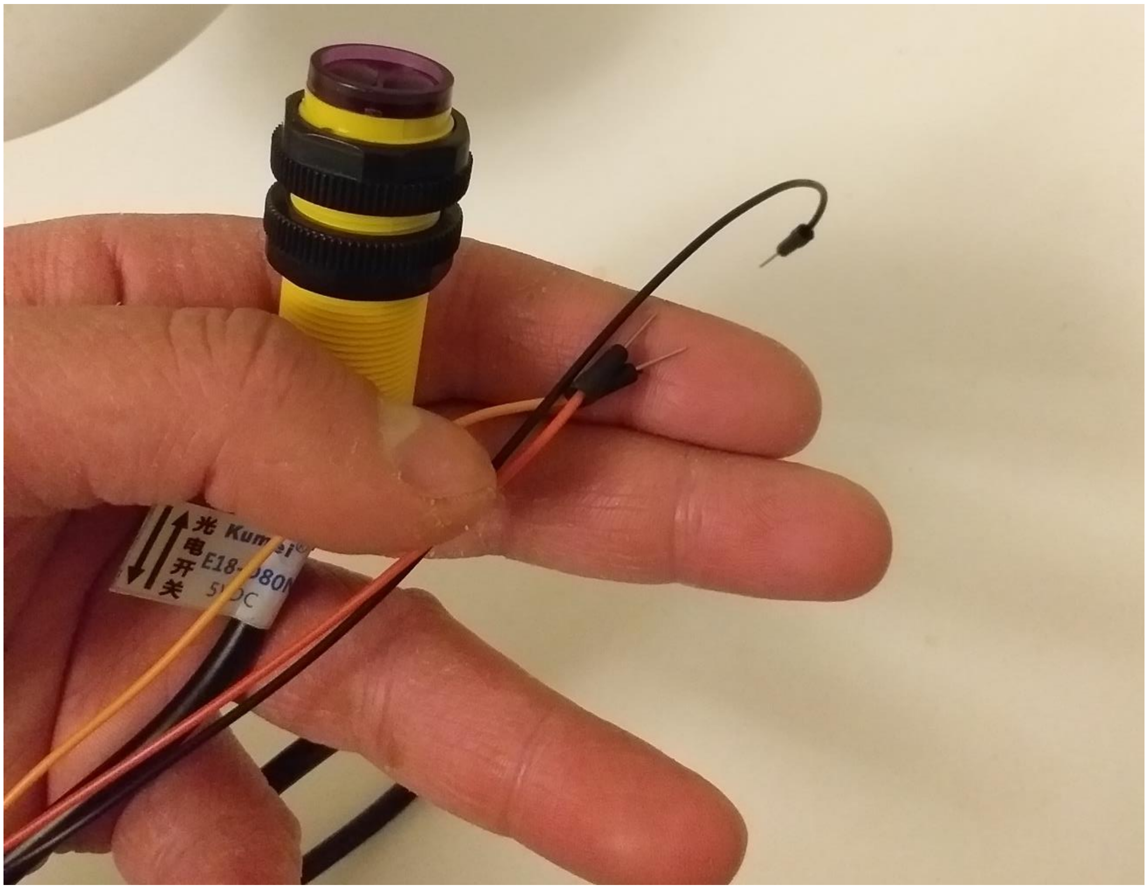


Figure 4 Infrared sensor, more expensive than the ultrasound sensors

Experiments were now carried out on the feasibility of Infrared sensors. All the tests that were carried out show that although infrared sensors are less affected by the lighting conditions indoor and outdoor, the range is still similar to the previous ambient light experiment. It can only detect objects from a range of 30cm. Infrared sensors are also more expensive than ultrasound sensors; the cheapest sensor was £9.89 for a single sensor. The higher costs discourage experimentation and rapid prototyping, so it was discontinued.

Ultrasound sensors are already a popular choice of sensors for object avoidance with robots[26]. So, this pathway was chosen. Using ultrasound sensors is the cheapest option that produced a working solution. It was also natural, as bats and dolphins and many other animals use ultrasound for communication and especially with the case of bats, object avoidance[27].

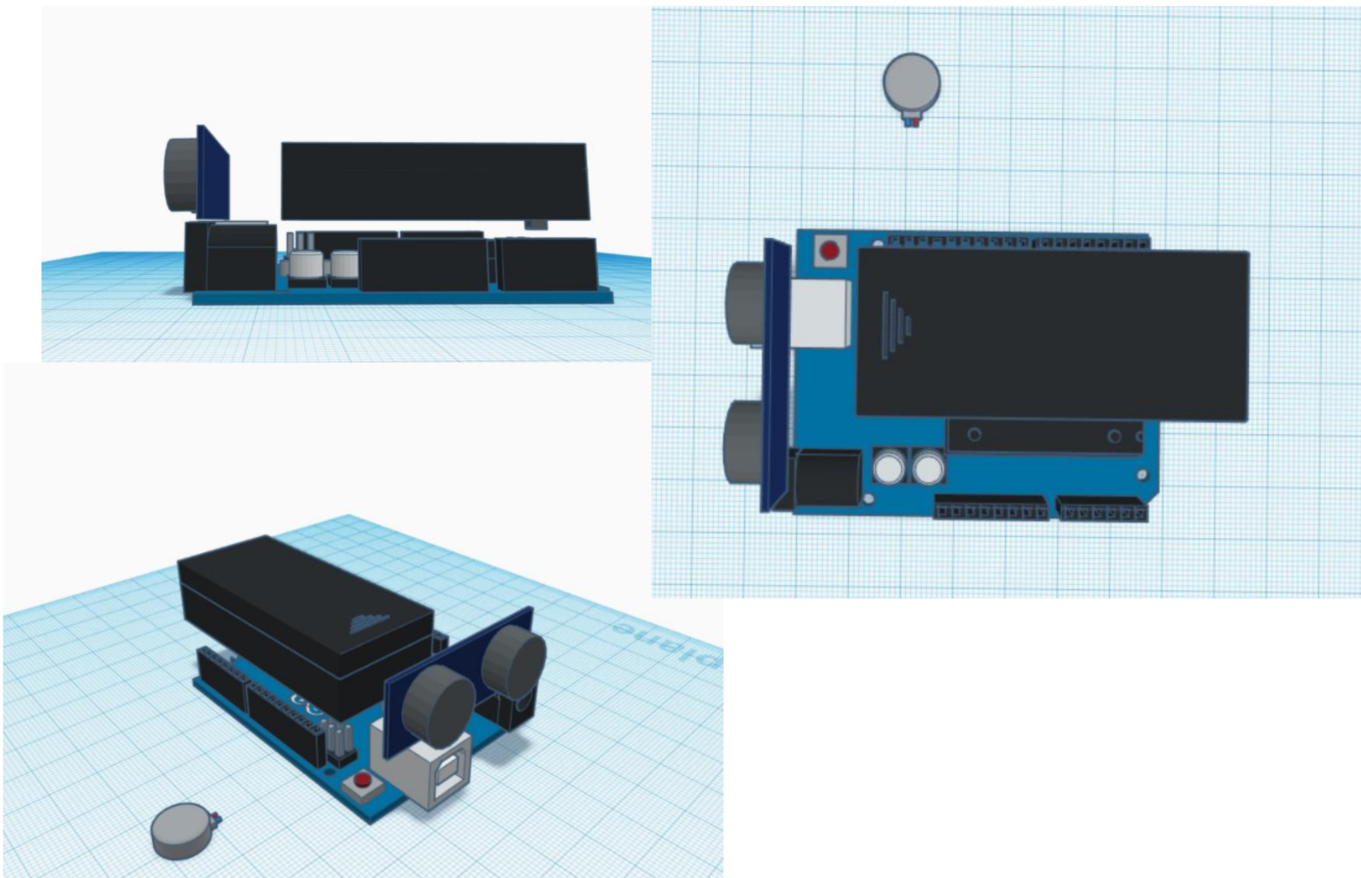


Figure 5 the minimum number of components needed to build one unit

[26] Yamada, Y., Ito, K., Oka, A., Tateiwa, S., Ohta, T., Kobayashi, R., . . . Watanabe, Y. (2015). Obstacle-Avoidance Navigation by an Autonomous Vehicle Inspired by a Bat Biosonar Strategy. *Biomimetic and Biohybrid Systems Lecture Notes in Computer Science,*135-144. doi:10.1007/978-3-319-22979-9_14

[27] Goerlitz, H. R., Genzel, D., & Wiegrebe, L. (2012). Bats' avoidance of real and virtual objects: Implications for the sonar coding of object size. *Behavioural Processes, 89*(1), 61-67. doi:10.1016/j.beproc.2011.10.018

Experiments were carried to find the minimum number of components to build one unit of a collision avoidance system. The lower the number of components, the cheaper the system, the less maintenance required and the lower the risk of the unit failing.

Instructions for building a Proxicane Unit:

1. Gather parts:

1 x Arduino

1 x USB cable to connect the Arduino to a PC

1 x Battery pack with switch (should have two wires connecting to the battery pack)

1 x Ultrasound sensor

1 x Cell Vibration motor

2 x [Optional] male-to-male jumper wires (to extend the length of the vibration motor)

6 x female-to-male jumper wires

[Optional] Duct tape

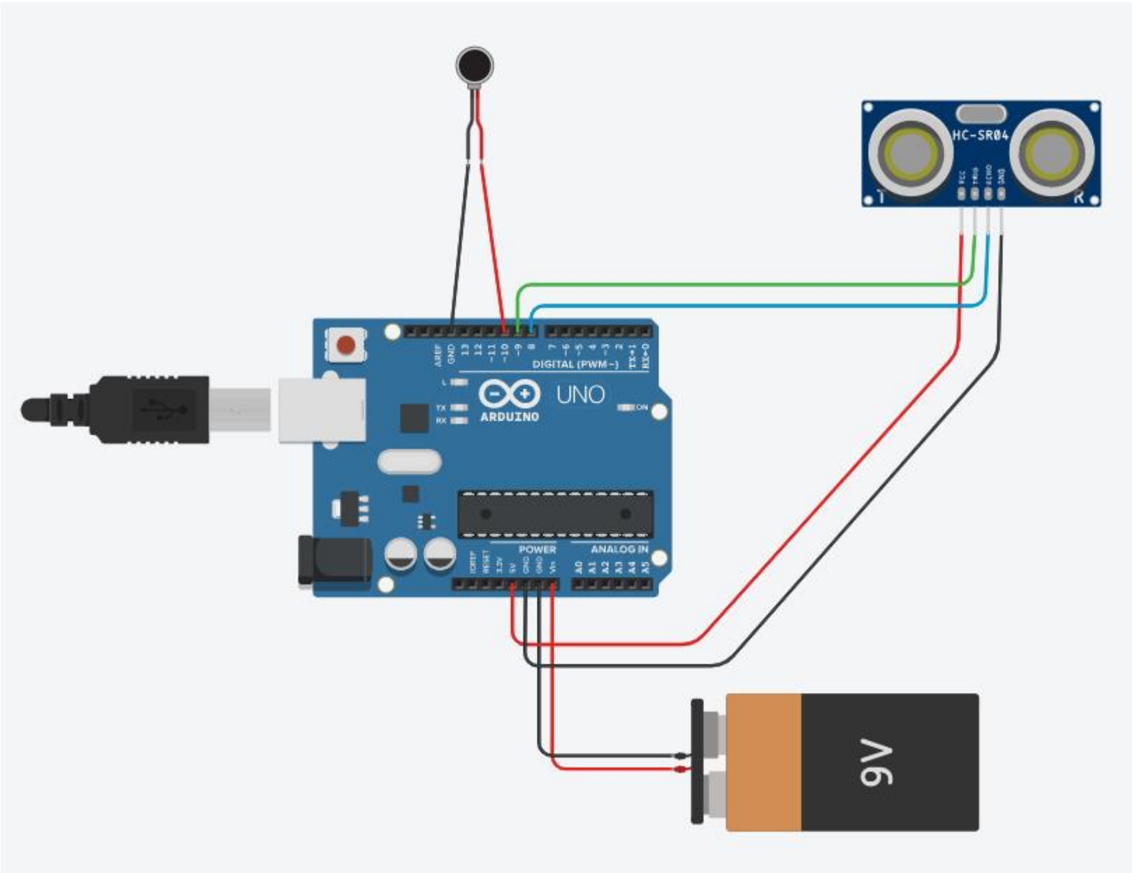
[Optional] Velcro tape

2. Upload the source code from https://github.com/Hussein-Ben/Proxicane/blob/master/Project%200.9/FINAL/final_code/final_code.ino onto the Arduino board by connecting the Arduino board to a computer using a USB cable. Download and install the official Arduino IDE from https://www.arduino.cc/en/Main/Software, or the web editor can be used[28]. There is an upload button in IDE. Ensure that the right COM port is chosen.

---

[28] https://create.arduino.cc/

3. Once the code is upload, disconnect the Arduino. Using the female-to-male jumper wires connect wire according to this schematic[29]



4. Insert a 9v battery into the battery pack and turn the switch on the battery pack. The Proxicane unit should be ready, a place near a wall and the vibration motor should vibrate.
5. [Optional] you can use Duct tape to secure Proxicane and Velcro tape to secure the Proxicane unit to clothing and equipment.

[29] https://github.com/Hussein-Ben/Proxicane/blob/master/Project%200.9/FINAL/final_code/Schematic.png?raw=true

Future Improvements

Proxicane is still a Proof of concept; it is not intended at this stage to be used full-time by a user. There are many improvements needed, by ensuring that the Proxicane project is open source, anyone can apply incremental improvements and fixes.

- Reducing the physical size of the Proxicane Unit:

This can be done by switching the Arduino for a smaller microcontroller such as the Arduino nano, Cheapduino or the LilyPad Arduino. For a smaller size, the jumper wires need to be replaced by soldering connections between components permanently; a PCB would be needed for this. A PCB design can also be open source and shared. A smaller size is essential as it will allow Proxicane to be cheaper and more discreet for the user.

- Reducing power consumption:

The current 9v battery is large, and the unit can only sustain power supply of 2 hours. It is essential that the user can have collision detection available throughout the hours when the user is awake; this is approximately 16 hours. Another issue is that the 9v battery is not rechargeable, replacing the battery on a regular occurrence is both financially expensive and not environmentally friendly. A rechargeable lithium-ion battery is needed. Ideally, this battery could be charged wirelessly.

- 3D printed case/cardboard case

A Proxicane unit needs a case to protect its internal components from dust, water and shock damage. A 3D printed case is cheap to print and is accessible to anyone with a 3D printer or a 3D print on demand service. However, there is a possibility that a cardboard case could be used. These cases would have also had to be tested for their ability to withstand shock, water splash and submersion (the user would object avoidance while walking outdoors in the rain).

- UI/UX (making it easier for the user to change the object detection distance)

Currently, the user has no method of adjusting the distance that a Proxicane unit detects a collision or to notify the unit of false readings. These features can be added with tactile buttons on the unit. However, the cost of the Proxicane unit will increase. It would also be useful if the Proxicane unit could provide noise/voice guidance for training the user and helping the user locate the device if they have removed it from the body.

- Standalone Proxicane Unit

The user currently has to connect the Proxicane unit to a computer in order for the detection distance to be changed via changing a variable in the source code. This is not practical and opens possibilities for the source code to have errors or be corrupted. (the user could accidentally change the wrong variable, add an unnecessary semicolon) It is essential that Proxicane does not rely on any external devices when setting up, to remove any barriers a user may have; some users may not have access to a PC. The problem with turning a Proxicane unit standalone is that it will require more processing power, it would need a voice-guided setup as the user may be blind or partially sighted.

- Collison detection

Smarter collision detection is needed for the Proxicane unit. Ideally, the unit could by using location data and feedback from the user, adapt to the user's environment to prevent false readings. The unit could recognise when the user is climbing up a flight of stairs or when they are in a lift. False readings occur when the user is climbing stairs or sitting down in certain types of seats in a bus for example. Adding the ability to record location data and requesting the user to tag this data by voice, will require considerably more processing power and storage on the Proxicane unit. Any type of Arduino microcontroller would be unsuitable; the Raspberry Pi computer is currently the minimum smallest computer that can be considered.

Before Proxicane – an attempt at an open source refreshable braille display

Primarily, building an open source refreshable braille display was the original intention, building upon my other braille related work https://github.com/Hussein-Ben/braille-machine. However, due to a lack of funding, I did not have enough financial resources to carry out experiments to see whether the idea is feasible.

What I had aimed to demonstrate was a more efficient design of a refreshable braille reader, current designs of braille readers are plagued by the paradigm of building one cell and then scaling up, which has resulted in excessive financial and engineering cost of braille readers. It is far more efficient instead to take a divide and conquer approach.

Most electronic braille readers range from £400 to over £3000+, which is out of the price range of the majority of blind people, an estimated 36 million people in 2015. £400[30] is considered cheap and revolutionary in the field of refreshable braille displays, but even this price, this product will still be inaccessible to a majority of blind users, especially in less economically developed countries. There was a great attempt to make an affordable and accessible refreshable braille display[31] however the technology is patented, there is no information on how it works, and moreover there has been no significant updates in the last 6 months[32] which indicates as with other braille reader attempts that they had problems securing funding.

---

[30] https://www.visionaware.org/blog/visionaware-blog/low-cost-refreshable-braille-at-csun/12
[31] see (http://blitab.com/ approx. cost £300 - 400)
[32] By looking at latest tweets and news on their website there is clearly problems. https://coolblindtech.com/looks-like-blitab-is-back/

Current braille reader mechanisms (how the product raises and lowers a single braille dot) have not changed since the piezoelectric mechanism was first introduced in the 1980s[33]. The Versabraille released in 1982 was the first portable refreshable braille display, current displays still use the same design principle and mechanism, which is why they are still expensive.

34

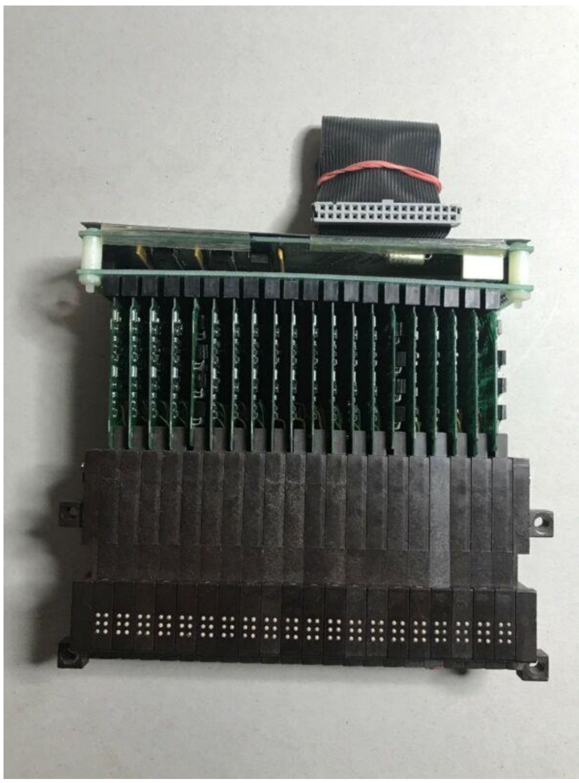


Figure 7 the Versabraille (1982) piezoelectric mechanism, current braille readers still use this same mechanism

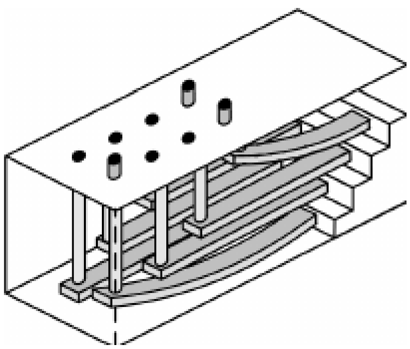
35

Figure 6 the Humanware braille reader (2019) the same piezoelectric mechanism is used

33 https://nfb.org/images/nfb/publications/fr/fr28/fr280109.htm
34 https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/FAIAAOSwR4hbihge/$_3.JPG?set_id=880000500F
35 https://store.humanware.com/heu/brailliant-14-braille-display.html

The Piezo-electric mechanism works by the effects of voltages on certain crystals. The crystal expands when a voltage is applied and contracts when there is no voltage. When the crystal expands, this force is used to push a pin upwards which in turn raises a braille dot.

[36] [37]



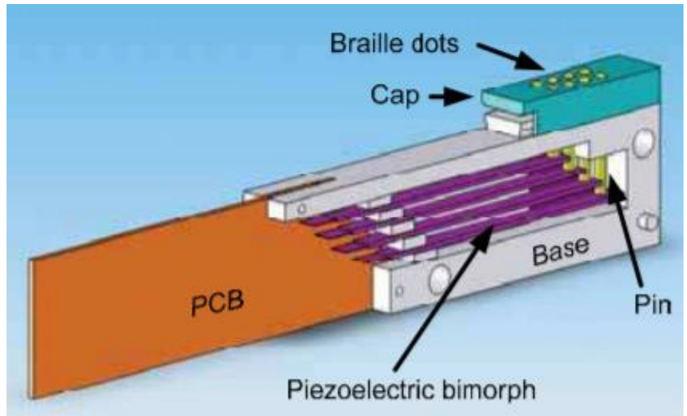


Figure 8 A cross-section of a typical braille cell

Unfortunately, high-cost designs require substantial capital for the product to become a reality and enter the market.

Why do we need refreshable braille readers? The chicken and the egg problem. A typical braille book needs to be at least three times larger than a conventional book; when a blind person reads braille they need space to brush their fingers over the braille dots, the spacing between dots is required to distinguish letters, numbers and punctuations (there is even braille music notation)

---

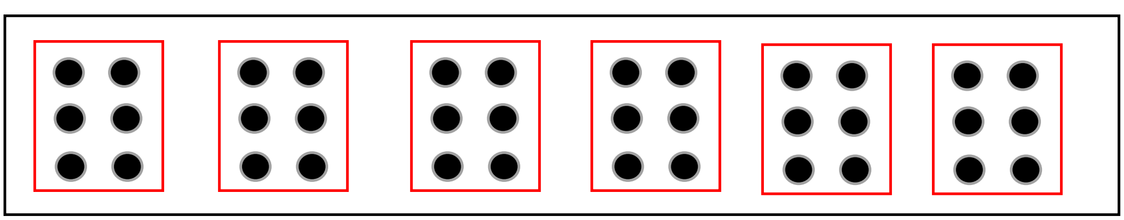
[36] Smithmaitrie, P. (2009). Analysis and Design of Piezoelectric Braille Display. *Rehabilitation Engineering*. doi:10.5772/7391
[37] Lévesque, V., Pasquero, J., Hayward, V., & Legault, M. (2005). Display of virtual braille dots by lateral skin deformation: Feasibility study. *ACM Transactions on Applied Perception*,*2*(2), 132-149. doi:10.1145/1060581.1060587

A typical novel of 300 pages converted into braille would take at least 6 to 10 large volumes and be non-portable (carried in cardboard boxes). Because of the large physical size, braille books are expensive to print and transport, because they are expensive, less blind people can access braille books as well as less blind people can learn braille so therefore there is fewer books and higher costs - the chicken and the egg problem, a vicious cycle.

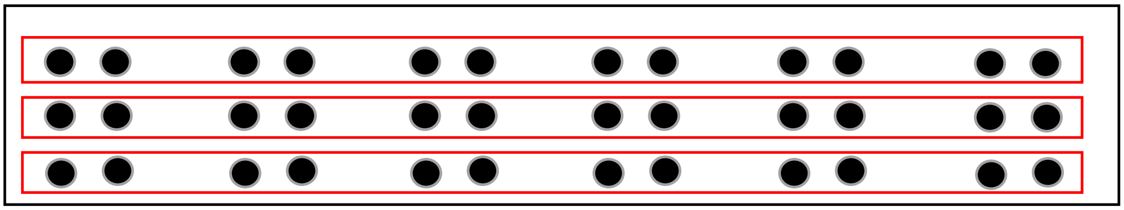
With a refreshable braille display, as many books, including books that do not have a braille format, can be read with just one device which is portable and only needs to be built once.

I created numerous realistic Proof of Concepts (PoC) of a new approach to the design of braille readers. Nearly all braille readers that have been made/designed have all focused-on building one braille cell and then scaling this up to form a line of cells (mimicking a sentence), most designers seem to follow this pattern. The problem with this is that the cost and the electrical/power requirements increase drastically from one cell to twelve or more cells. What I hope I have demonstrated is that it is far more efficient to carry out a divide and conquer strategy.

Normal cell being scaled up, this the current design paradigm of braille readers on the market today.



My PoC is dividing a line of braille cells into three parts, a divide and conquer approach.



I am hoping this PoC can influence new and more efficient attempts at making a refreshable braille display, which is desperately needed.

3D models have already been created, what I discovered through building the models, is that a refined finished product is not possible at the end of this project. It is also unfortunate that there are "patent squatters" as shown with this open source attempt[38]

[38] https://hackaday.io/project/10849-refreshable-braille-display.

Braille Reader Specifications

- What is the minimum required number of resources electronic components required to build a single line of a braille cell?
- Are there DIY alternatives into building a braille display with minimum use of microelectronics?
- Low cost, the whole system must cost under £100 to build.
- Must use physical materials that are available in electronic shops, electronic dumps or salvaged from cheap (under £20) existing electronic products.
- Must only use electronic components that are available in electronic shops, electronic dumps or salvaged from cheap (under £20) existing electronic products
- Must only be built with available consumer tools, not Industrial tools. Therefore, no factories but small 3D printers are allowed.
- Must always consider how a blind person reads braille, ideally low cognitive load in using the reader.
- Must be portable (battery powered), relatively lightweight (under 1.5kg) and durable.

Experiments

Experiments

The Main inspirational Book I read was *Code: The Hidden Language of Computer Hardware and Software*. I have also begun learning how to use blender software and have completed many drawings of PoCs and evaluated them using the Braille Reader Evaluation Criteria; they are ready to be made into 3D models.

I initially started with experiments with ideas that were feasible, but the search space was narrowed down into solutions that are low cost and feasible:

First iteration:

I have decided that after experimenting with DIY electromagnets that there is no way to control the repulsion which is the fundamental mechanism that allows a braille dot to be raised or lowered. The problem I discovered was that after an electromagnet becomes magnetised, it does not become de-magnetised when you switch the electromagnet off.

The electromagnet was replaced with a simple DC motor. Why DC motor? Firstly, they are available everywhere worldwide, there found in many toys and electronics (and electronic landfills), so there accessible in developing countries, secondly there relatively cheap. A user can also build their own DC motor[39].

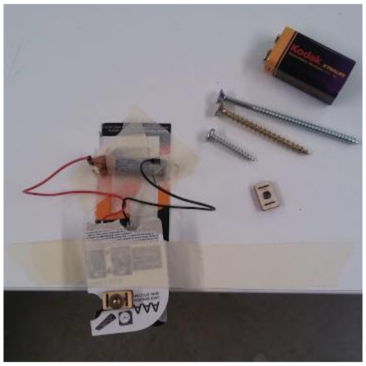
---

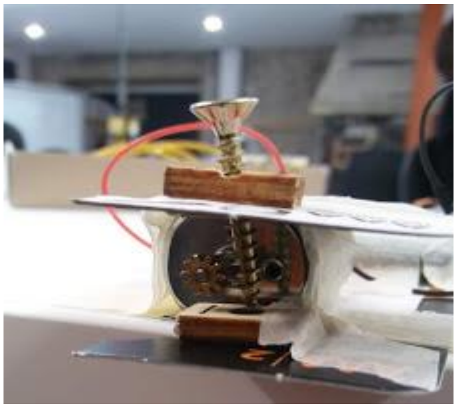
[39] https://www.wired.com/2016/01/how-to-build-a-super-simple-electric-motor-out-of-stuff-you-already-have/

I can still use the metal nails; most dc motors have a gear attached to the rotor. I found that the gear can move upwards and downwards when the DC motor is on, and you change which direction of the screw (clockwise/anticlockwise) when you reverse the direction of the battery. Using the 9V battery was too powerful, I found that the AAA battery was satisfactory and it is much smaller than the 9V battery. For this prototype I used what I could find; the mini wood block is a broken piece I found in the kit and cardboard was from a battery pack.

I now need to introduce some control to the DC motor and build a more secure base for the prototype.

Figure 9 magnetised nail





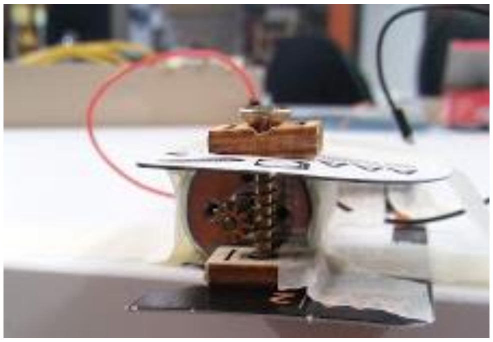
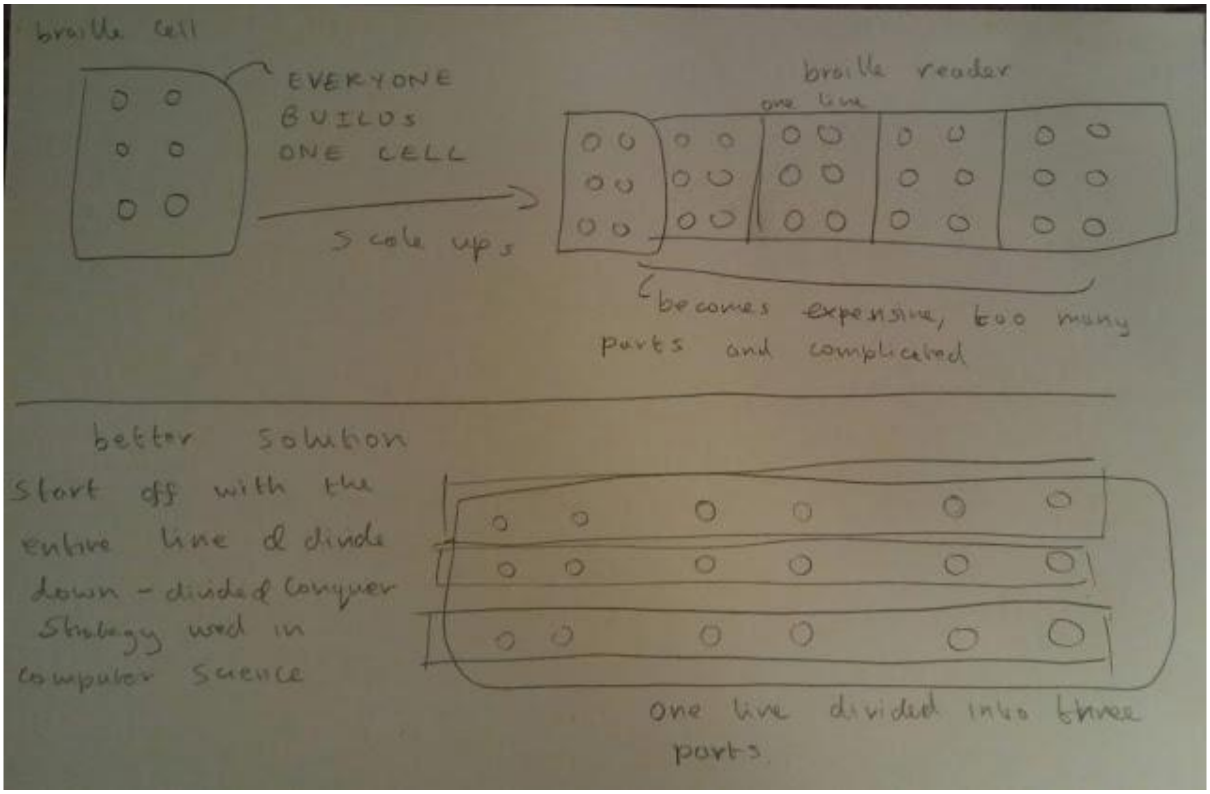


Figure 10 left image: raised position. right image: lowered position. DC the screw would have a dot on the end of it.

Second iteration:



Instead of building one braille cell and then scaling up, it is more efficient just to divide an entire braille line into three parts. The focus is primarily centred on ensuring that one divided part raises a dot or lowers a dot. A rail in which a needle was pushed along by a motor was the solution that required fewer parts.

However, the problem is that there is only one needle per divided part; how do multiple cells get updated? The solution here is to find a way to detect where the user's finger is, in order to know which braille cell their finger is currently on. A blind person would read braille by brushing their index finger over the cells. The simplest way to do that is detecting light changes.

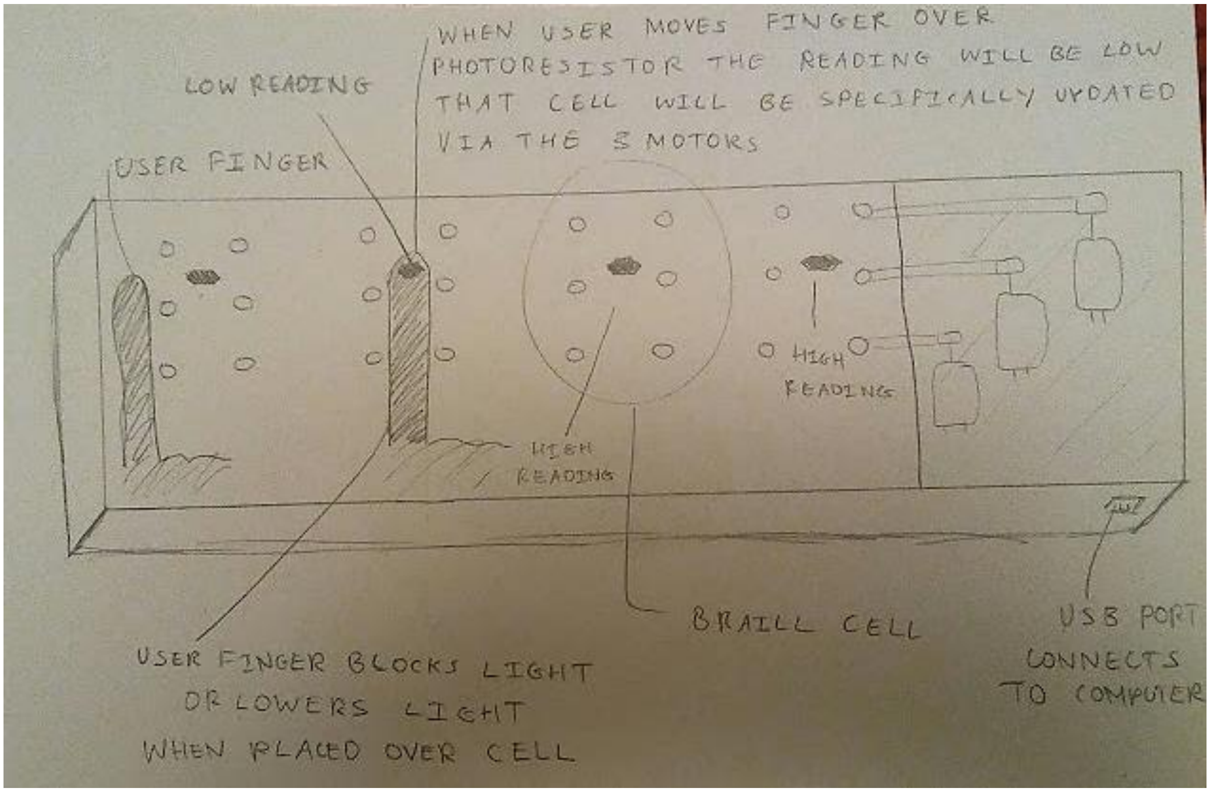


Figure 12 using ambient light to detect where the user finger is

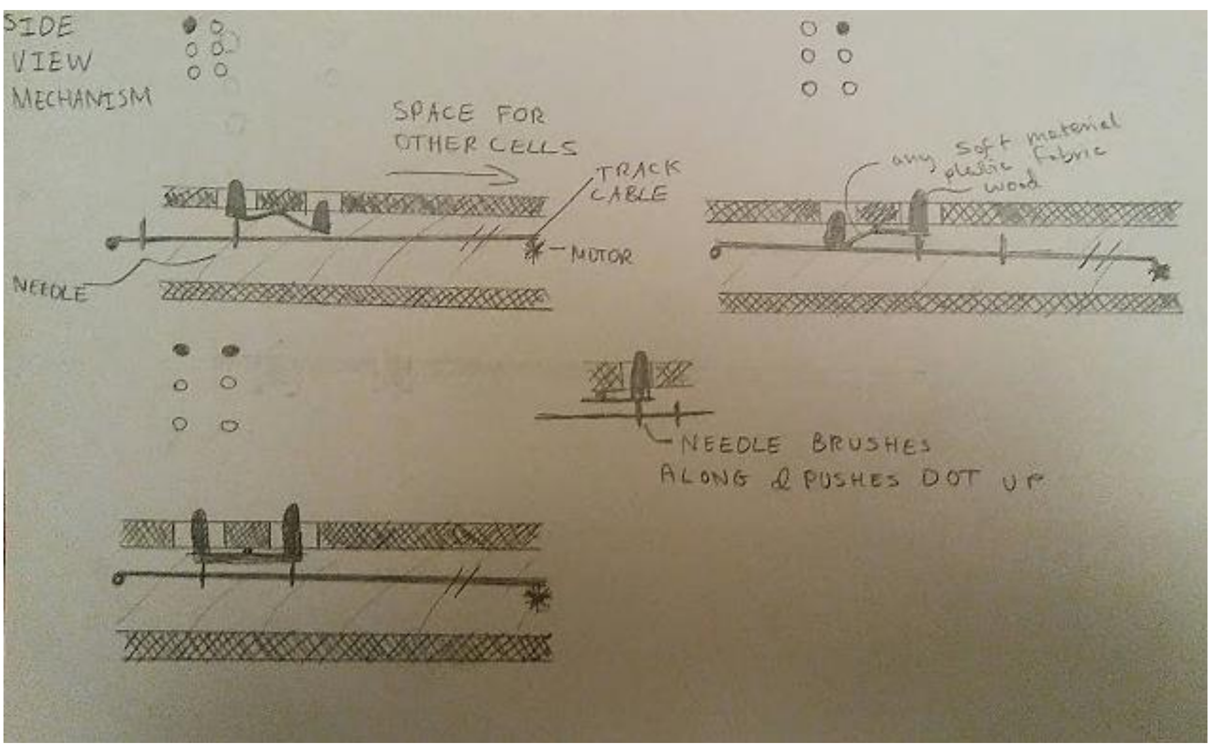


Figure 11 cross-section

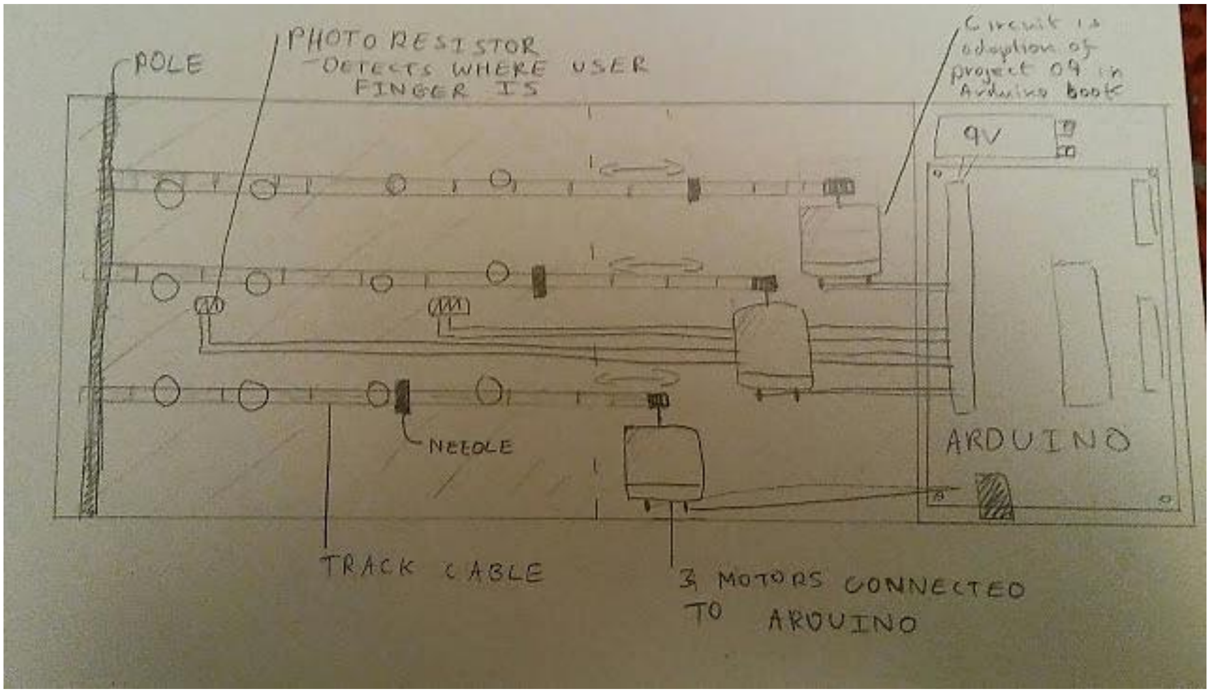
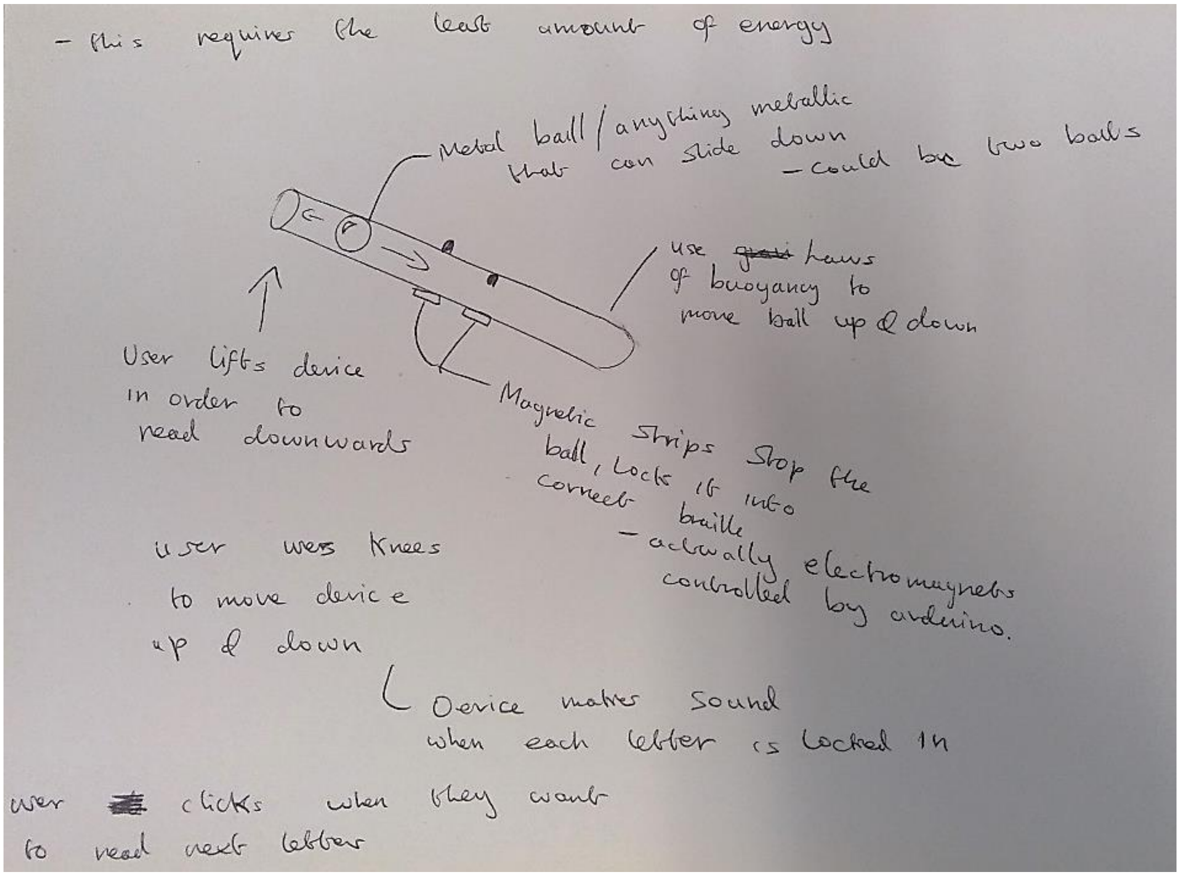
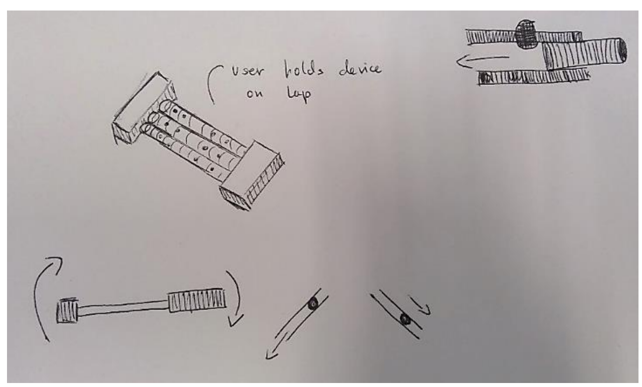


Figure 13 overview of the design

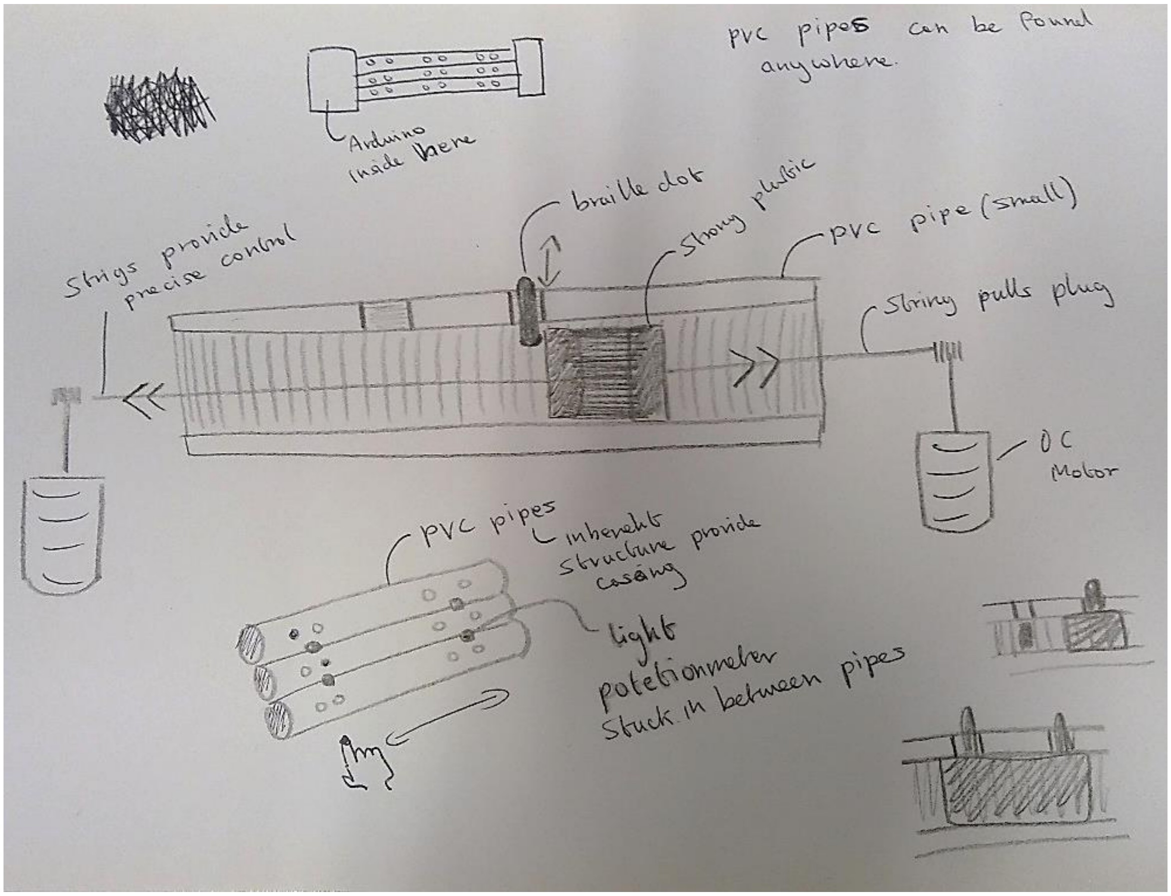
I further improved the design by using two smaller motors. All of the braille designs I intend to use, all use off the shelf parts, materials that can easily be acquired online, in an electronics store or even from electronic dumps and old electronics products.

Third iteration:



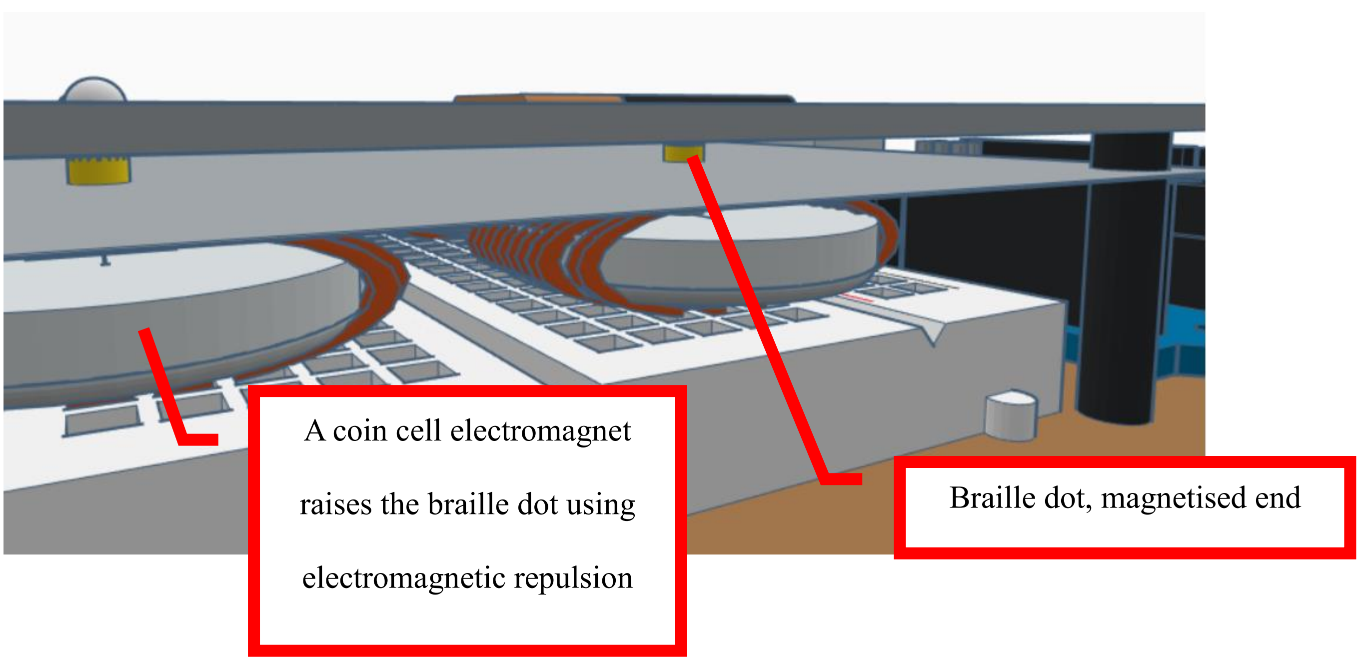


In order to reduce the amount of battery power required by the unit, experiments were carried out using gravitation. A ball inside a PVC pipe when sliding over a specific cell will push a braille dot upwards. By default, the ball will be stopped by magnets (the ball is magnetic), each braille cell has a designated magnet. When the user has read one cell, they use the switch to let the ball move using gravity to the next cell. Power is only needed to turn on and off the electromagnets. The electromagnets are the only components that need electricity. This approach has not been tried out before (no research papers on using gravity in relation to braille technology) since only cheap electromagnets are needed (which are robust and have no moving parts), the entire unit is low maintenance, easier to build and more durable. Durability is essential as the user will be physically carrying a braille reader and be using the reader throughout the day as well.

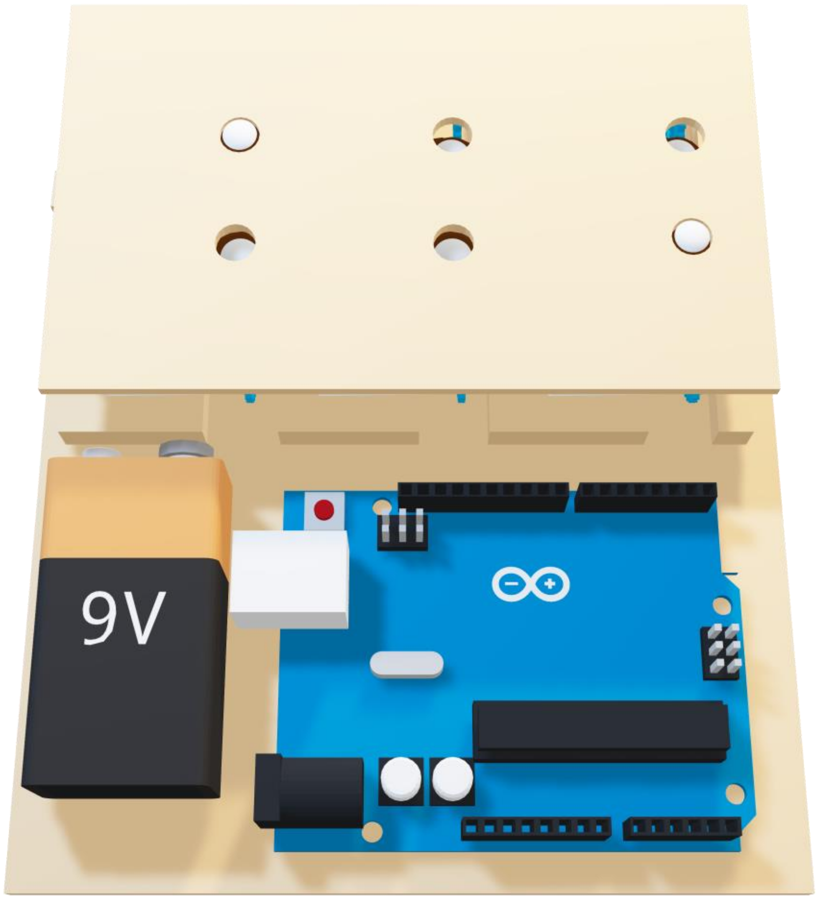


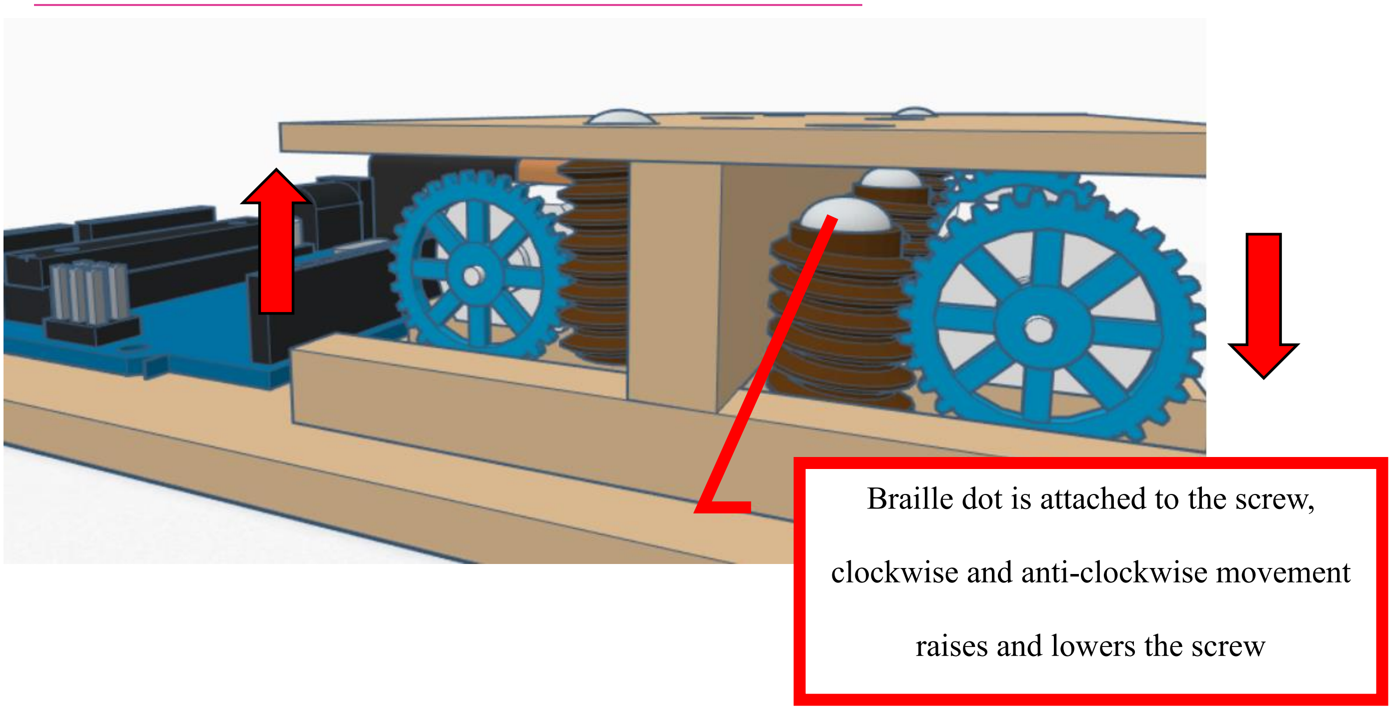
There is also another modification to the design that does not use gravity. Instead, a device similar to the ball is pulled in either direction using a DC motor. The benefit of this is that the user does not need to move the whole device manually. However, the unit will require more power for the DC motors.

All the designs that were developed from the experiments have been 3D modelled. The key focus of these PoC is the mechanisms, the principle that enables a braille dot to be raised and lowered; this is the key problem.

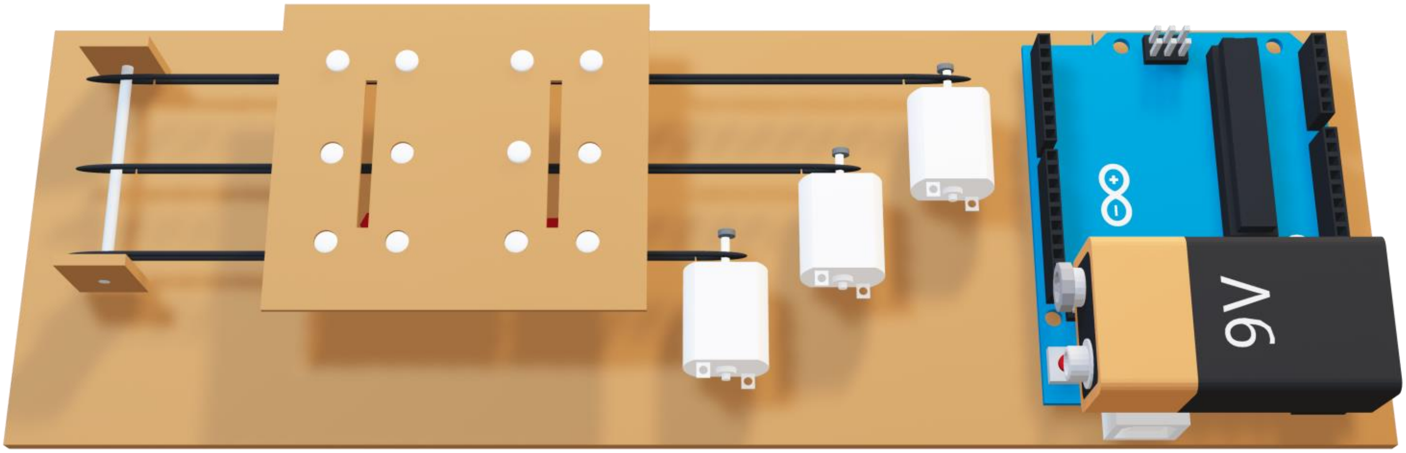


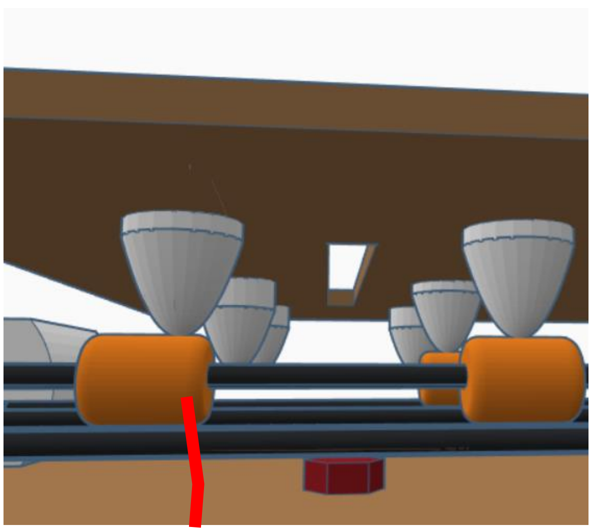
Electromagnet Concept



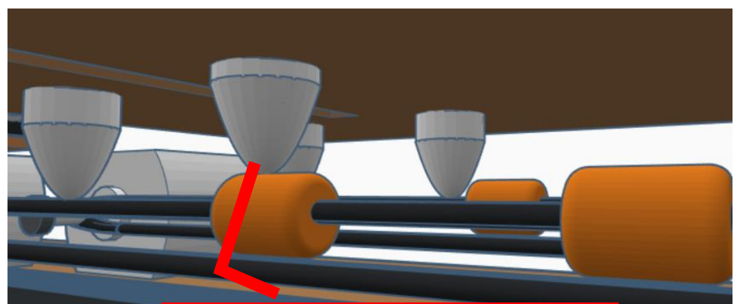


Rail concept revision 1

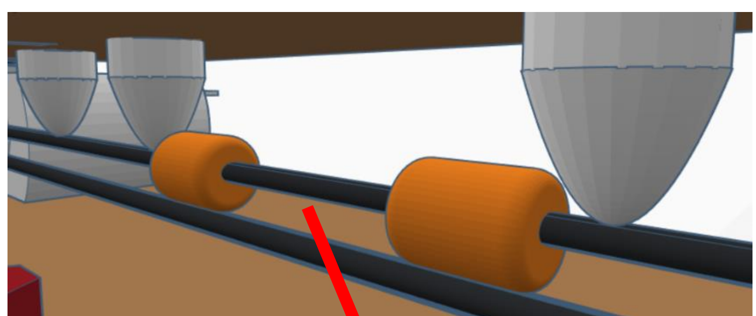




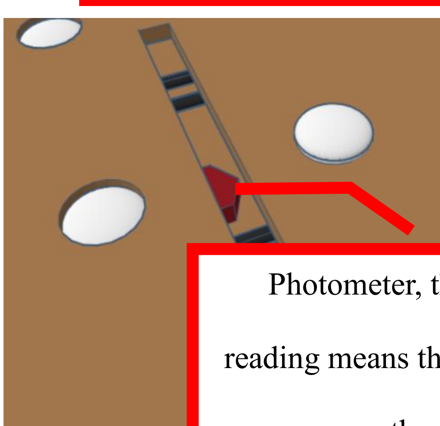
Both Braille dots are raised



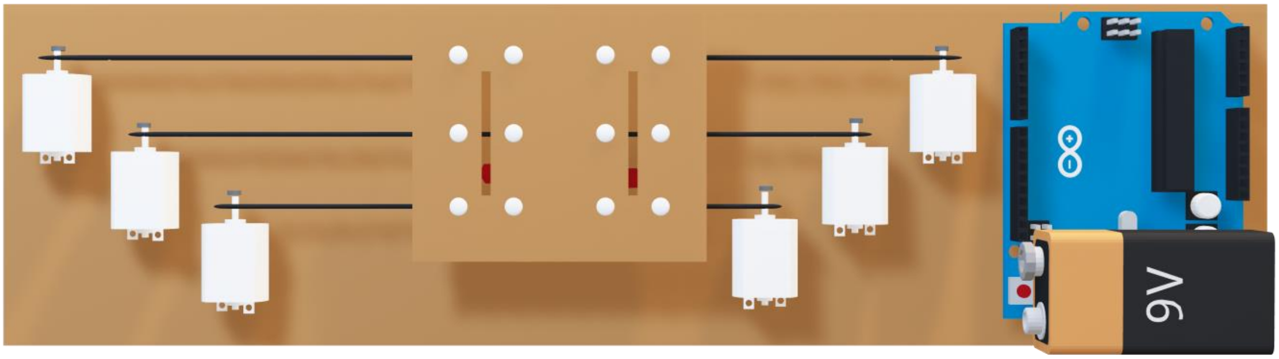
One dot is raised



No dots are raised

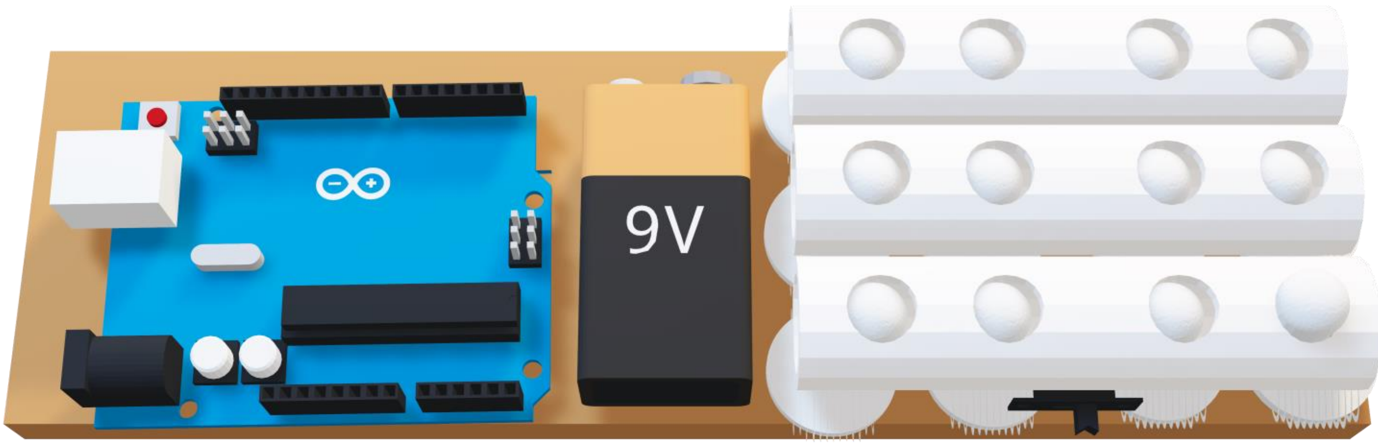


Photometer, the gap lets light in. A low or lower reading means that the user has passed their finger into the vicinity of the braille cell

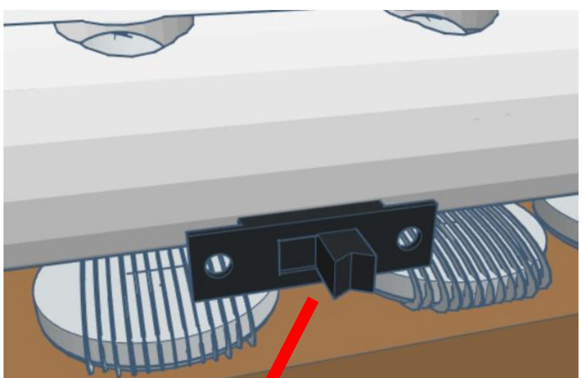


Rail concept revision 2

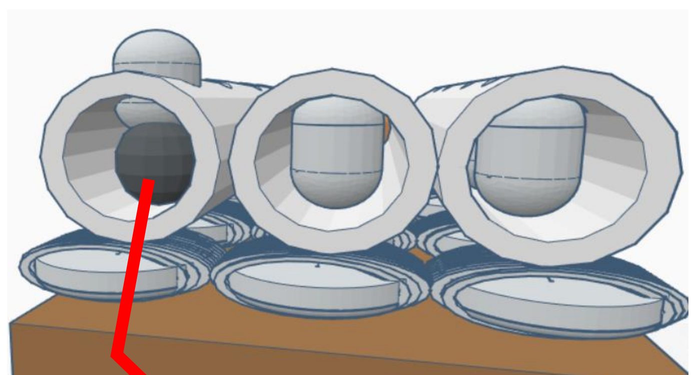
It is similar to the previous design, except there are motors on both sides. This is to increase the speed that the braille cells can be updated. However, there is a more significant cost of power consumption and wiring due to the need of 6 DC motors for each braille line.



PVC pipe concept



Switch for user to change locked position of the magnet ball.



Magnet ball stopped by electromagnets. User tilts the reader to move from one braille cell to the next.

What if braille itself needs to be updated?

Louis Braille invented the Braille system in 1824 with the limitations of current technology. The reason why the braille system was successfully adopted was that it was cheap to produce braille text; all that was required was a Slate and a stylus, simple tools can be built with low cost.[40] There were already other alternative systems during the 19th century such as the Moon system[41].

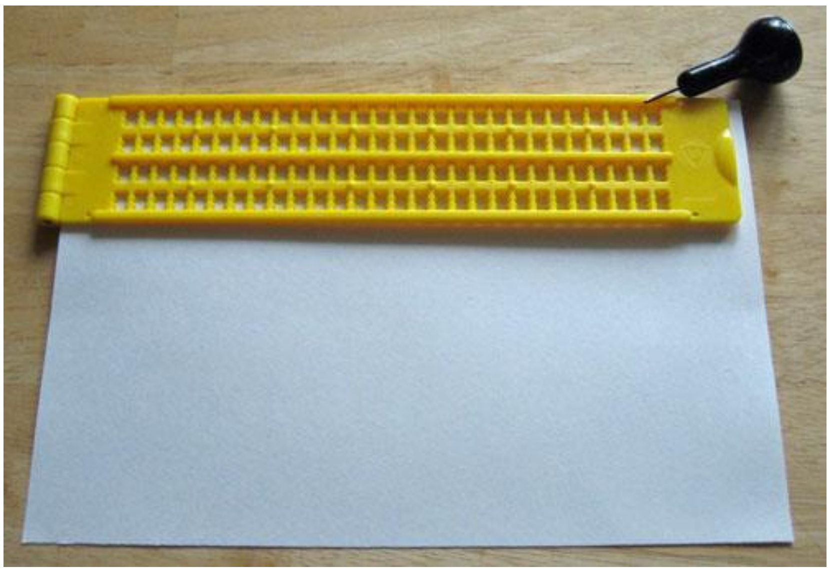


Figure 14 a Cheap Braille Slate and Stylus

If Louis braille were alive today, he would not be limited by the technology of the 19th century. This is an idea that has driven the Elia idea[42], an alternative to braille system that is more similar to the Latin alphabet. Nasa engineering control panels inspire the shapes of letters in the Elia system. It takes roughly 20 minutes to learn the whole system compared to a standard braille system which takes six months to 1 year to learn. One of the biggest problems with braille is that it requires a user to feel the dots; this requires prolonged training over six months to build

[40] http://www.specialneeds.tokoyasu.com/wp-content/uploads/2017/10/Slate-and-Stylus.jpg
[41] https://www.atlasobscura.com/articles/how-a-blind-doctors-moon-code-helped-thousands-read-again - its important to note that users with calloused fingers had difficulty learning other systems
[42] http://www.theeliaidea.com/faq

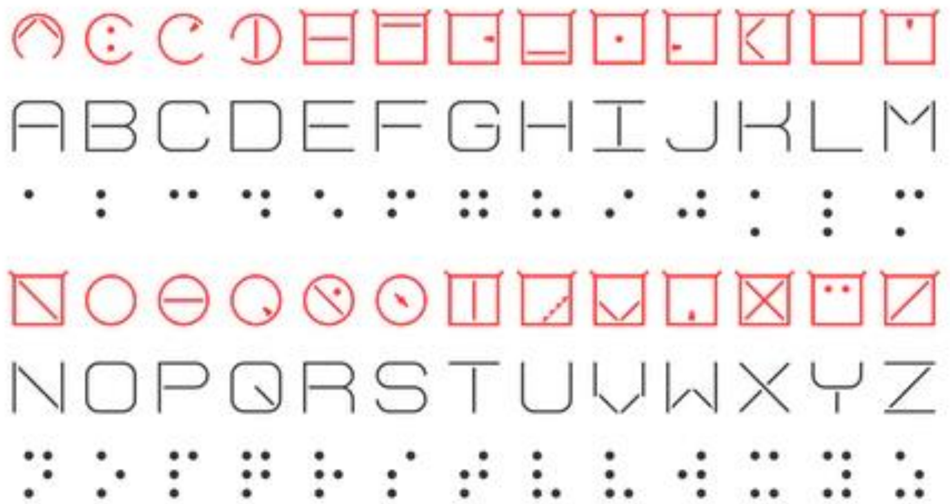


Figure 15 the ELIA frames(alphabet)

[43]

sensitivity in a braille users finger. As the braille user becomes older, they start losing sensitivity in their fingers, which makes reading braille more difficult.[44] Whereas the Elia system design

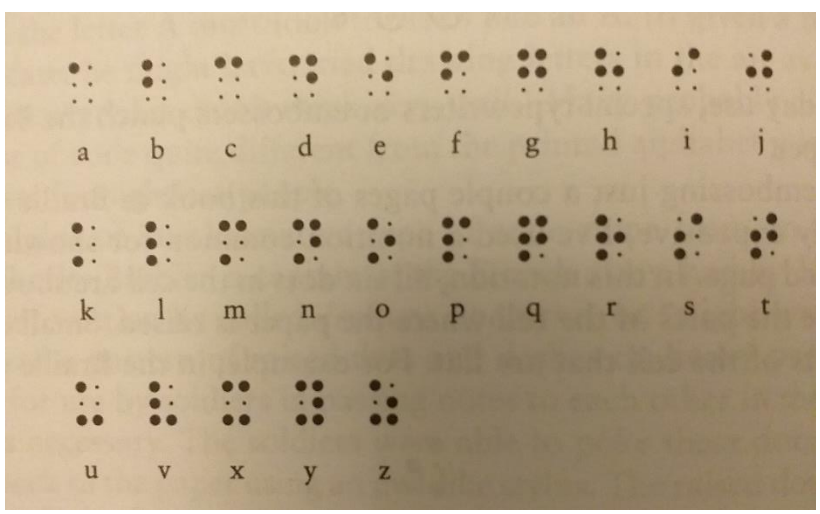


Figure 16 Petzold, C. (1999). Code. Redmond, WA: Microsoft Press. pg 18 the repeating pattern of braille

---

[43] http://www.theeliaidea.com/elia-frames
[44] https://www.fastcompany.com/90136975/the-complicated-quest-to-redesign-braille

circumvents this, as it allows greater spacing between a raised shape, the user has space to move their fingers, (in braille systems a gap between braille dots is only a few millimetres)

Although, the braille alphabet is simple to learn, in order to reduce space and increase reading speed, contractions are used. Common words such as “The” are shortened.

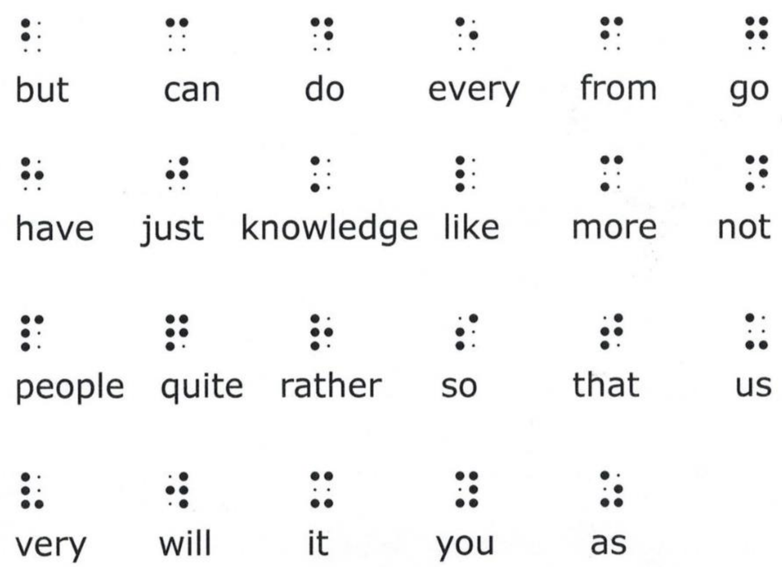


Figure 17 common braille contractions for the Unified English Braille

Each language has its own system of contractions which are updated continuously; these systems create a significant cognitive load on a braille user, as they have to remember all the contractions. This can discourage further learning of braille.

With so many unsuccessful attempts at building a refreshable braille reader and a declining braille literacy, could it be that the braille system itself needs to be replaced?

More public attention and discussion are needed around the issue of blind literacy.

Conclusion

As discussed in the chapter **Future Improvements,** Proxicane is still a proof of concept; more development is needed for Proxicane to become ready for everyday Collison avoidance. Hopefully, as the Proxicane project is open source, anyone can contribute, improve and extend Proxicane.

Proxicane has introduced a different way of implementing object avoidance, by focusing on users' legs, it is hoped that this idea will encourage other researchers and developers to try different approaches too. The entire repository of Proxicane will be available here: https://github.com/Hussein-Ben/Proxicane

Frequently used terms

PoC: Proof of Concept, a term generally used by the hacking community to show that their "hack" is possible. In this project it is used in the same way, to show that low-cost open source object avoidance is possible for the blind as well as the possibility of alternative approaches to building a refreshable braille display.

Proxicane: also referred to as the Proxicane project. The project aims to build a PoC which highlights a different approach to placing object avoidance system on the body; it also showcases a low cost more discreet system that can replace the cane used by blind or partially sighted people.

Proxicane unit: the physical device which provides objection avoidance in one direction, such as the front of the user. This is because there is only one haptic feedback end for each unit; one unit can only alert in one direction.

Bibliography

Petzold, C. (2015). *Code: The hidden language of computer hardware and software*. Redmond, WA: Microsoft Press. The main book that inspired my interest in braille.

Platt, C. (2013). *Encyclopedia of electronic components Volume 1: Resistors, Capacitors, Inductors, Switches, Encoders, Relays, Transistors*. Sebastopol, CA: Maker Media.

Platt, C. (2013). *Encyclopedia of electronic components Volume 2: LEDs, LCDs, Audio, Thyristors, Digital Logic, and Amplification*. Sebastopol, CA: Maker Media.

Platt, C. (2013). *Encyclopedia of electronic components Volume 3: Light, Sound, Heat, Motion, Ambient, and Electrical Sensors*. Sebastopol, CA: Maker Media.

Horowitz, P., & Hill, W. (2018). *The art of electronics*. New York: Cambridge University Press.

Burns, M. F. (2015). *Burns braille guide: A quick reference to Unified English Braille*. New York, NY: AFB Press.

WORLD'S FIRST TACTILE TABLET. (n.d.). Retrieved from https://blitab.com/

Refreshable Braille Display. (n.d.). Retrieved from https://hackaday.io/project/10849-refreshable-braille-displayan example of the traditional one cell approach

Low-Cost Refreshable Braille at CSUN. (n.d.). Retrieved from https://www.visionaware.org/blog/visionaware-blog/low-cost-refreshable-braille-at-csun/12 $700 is considered cheap.

The Holy Grail of Braille. (n.d.). Retrieved from https://www.visionaware.org/blog/visionaware-blog/the-holy-grail-of-braille/12 - Orbit reader 20 was the closest cheapest attempt

Introducing the Orbit Reader 20. (2019, March 07). Retrieved from https://www.rnib.org.uk/orbit-reader-20 its price £538 is considered cheap and revolutionary. The piezoelectric technology is from a patented tech from the 1980s.

Levin, D. (n.d.). OI Engine, an innovation management software built on design thinking. Retrieved from https://challenges.openideo.com/challenge/refugee-education/ideas/the-world-s-first-fab-lab-in-a-refugee-camp/comments A similar project that has inspired me is an echolocation device mounted to the arm rather than the leg

Also the OneMoreMile project from Stanford University, a running aid for the blind: https://web.stanford.edu/class/engr110/2017/pdf/OneMoreMile.pdf