UHL Aid

ARTURO AMAYA*, UCSD, United States
JULIAN RAHIMI, TUM, Germany
SAMMY KOUJAH, UCSD, United States
SCOTT WEBSTER, UCSD, United States

People with Profound Unilateral Hearing Loss (UHL) face difficulties with sound localization, which has been proven to have lifelong consequences on cognitive development and social capabilities. Addressing this problem depends on an individual's specific type of hearing loss: conductive, sensorineural, or a mix of the two. While adequate solutions exist for those suffering from conductive hearing loss, there are no existing hearing aids designed for Unilateral Sensorineural Hearing Loss that solve the issue of sound localization whilst maintaining comparable fidelity to natural hearing. As an alternative to cochlear implants, we present a Sensory Substitution Device (SSD) that utilizes haptic feedback to serve as a sensory transposition for binaural localization. A hasty (yet functional) implementation has been made using a Raspberry Pi microcomputer, two Samsung Galaxy Watch 4 smartwatches, and a Seed ReSpeaker v2 microphone array.

CCS Concepts: • **Human-centered Computing → Accessibility**.

Additional Key Words and Phrases: Unilateral Hearing Loss (UHL), haptic feedback, Sensory Substitution Device (SSD), Sensory Transposition

1 INTRODUCTION

Hearing loss is the third leading cause of disability globally, with approximately 466 million people living with disabling hearing loss around the world[7]. But with so many different forms of hearing loss, it can be hard to find a universal solution to all hearing based disabilities. For this paper we will be talking about a specific category of hearing based disability called Unilateral Hearing Loss, or Single Sided Deafness. More specifically, we will be tackling a unique type of UHL called Unilateral Sensorineural Hearing Loss. We will briefly explain the science behind hearing loss, what UHL is and how it differs from other forms of hearing loss, and what our proposed solution is to this unique problem. The goal of this paper is to walk you through our motivation for our research and the design of our proposed solution. This paper details our CSE 118 class project, from our development process to the architecture and technology we used.

Sound waves propagate through the air and are funneled by the outer ear pinna into the ear canal, hitting the tympanic membrane (eardrum) that borders the middle ear. This flexible membrane then vibrates the auditory ossicles (the malleus, incus, and stapes), which function as a physical mechanism to relay dampened vibrations to the inner ear. The fluid-filled cochlea receives these vibrations, which is then transduced by stereocilia (tiny hair-like cells moved by the fluid within the cochlea) into electrical signals for the nervous system.

If any part of this pipeline is interrupted, then hearing loss will occur. Conductive hearing loss originates from an interference in bone conduction, whether this occurs from blockage of the ear canal, a ruptured eardrum, or ossicular chain dislocation. Sensorineural hearing loss originates from cochleovestibular malformations or Auditory Neuropathy Spectrum Disorder (ANSD). [2] Conductive and sensorineural hearing loss may also be present simultaneously. All of these causes may be congenital, sudden-onset, or progressive with aging. Hearing loss is considered "Profound" when little to no hearing threshold remains. [7]

*All authors contributed equally to this research.

Authors' addresses: Arturo Amaya, a1amaya@ucsd.edu, UCSD, San Diego, California, United States, 92092; Julian Rahimi, TUM, 1 Thørväld Circle, Hekla, Germany, larst@affiliation.org; Sammy Koujah, UCSD, San Diego, California, United States, 92092; Scott Webster, s1webste@ucsd.edu, UCSD, San Diego, California, United States, 92092.

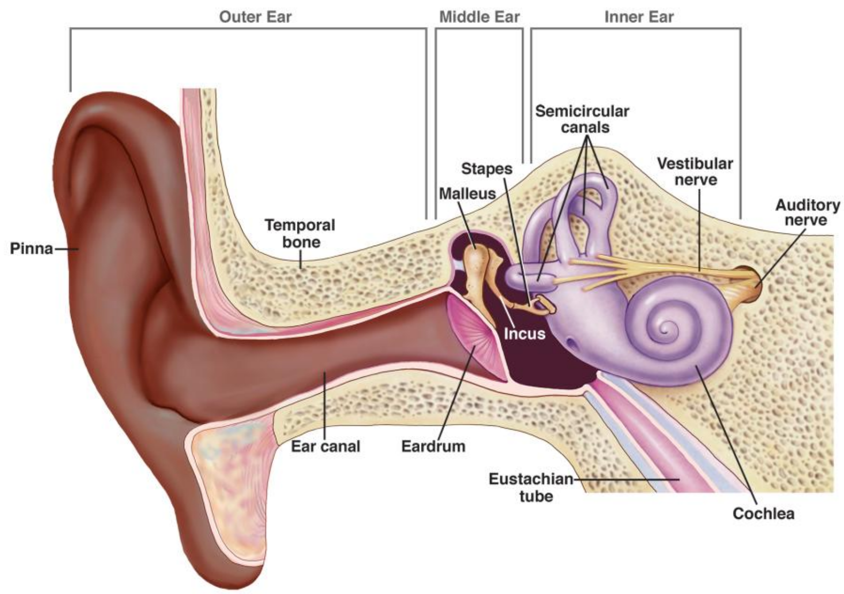


Fig. 1. Diagram of the outer, middle, and inner ear [1]. Sensorineural dysfunction pertains to the inner ear.

When hearing loss is localized to a single ear, it is termed Unilateral Hearing Loss (UHL). That is, one ear possesses normal function while the other does not. People with Profound Unilateral Hearing Loss are unable to utilize binaural hearing cues, which play a large role in spatial perception.[7] This results in a reduced quality of life on social, functional, and psychological levels.

2 MOTIVATION AND BACKGROUND

Children with profound UHL face difficulties with cognitive development and language acquisition. Overall, they are more likely to struggle in school than their counterparts with normal hearing. This is thought to originate from impaired binaural abilities to localize sound and parse conversations in complex auditory environments [5]. Those who develop profound UHL later in life still face the same difficulties of sound localization, the head shadow effect, and speech perception.

Sound localization is best accomplished with binaural hearing. While people restricted to monoaural hearing are capable of sound localization on the side of their dominant ear, this ability is severely limited in noisy situations. The head shadow effect, in which sounds coming from the direction of the contralateral ear are dampened or sometimes even imperceptible to the functional ear, assists binaural hearing cues by allowing two functional ears to measure the relative differences between audio sources (amplitude, frequency, etc). However, people with monoaural hearing are incapable of perceiving these relative differences [7]. This is analogous to the concept of stereoscopic vision: the brain processes the relative differences between two inputs in order to accomplish a new function. Both inputs are required for this processing to occur. All of these things combined affect a person's ability to recognize or interpret human speech, which is arguably the main issue for those with Profound UHL.

The inability for those with Profound UHL to identify the presence or location of human speech (due to the head shadow effect and poor localization skills) can be accommodated for depending on the mechanism in which the hearing loss has occurred. Conductive hearing loss is able to be treated with bone conduction hearing aids, which offer an artificial conductive interface to the cochlea. These can be used as wearable devices or surgically implanted as a Bone Anchored Hearing Aid (BAHA) or a Middle Ear Implant (MEI). Implants are more effective than wearables, but they require invasive surgical procedures. [6]. Conduction hearing aids are also not appropriate for those with sensorineural hearing loss, as they depend on a functioning cochlea and/or auditory nerve.

Our project is aimed to alleviate symptoms of those with Unilateral Sensorineural Hearing Loss as current solutions are incapable of solving the problem of sound localization while maintaining comparable fidelity to natural hearing. Contralateral Routing of Signal (CROS) devices use a microphone or bone conduction device to route audio (via radio, wire, etc.) from the contralateral ear to the functional ear. This allows the user to overcome the head shadow effect, but adversely affects monoaural sound localization capabilities [3]. Cochlear implants are useful for those with many forms of Bilateral Sensorineural Hearing Loss, but sound inferior to natural hearing, which causes discrepancies for those with UHL. [8] Cochlear implants are also not suitable for those with ANSD, as they must interface with a working auditory nerve. To circumvent all of these compatibility concerns, we are developing a sensory substitution device (SSD) utilizing haptic feedback as a means to provide sensory transposition for auditory localization.

A sensory transposition is a modal change in how a sensation is experienced. Research indicates that long-term usage of SSDs results in a synesthetic effect, wherein users actually come to unconsciously adopt these devices to perceive sensations that match their originally intended biological capabilities [4]. By using a sensory transposition device as a means to accommodate for auditory localization, we allow those with Profound Unilateral Hearing Loss to more effectively participate in human to human interaction, alleviating a number of social and developmental anxieties.

3 DESIGN

Our project design revolves around finding a way to supplement, or even replace, a person's hearing with another sense. One of our first ideas was to design a visual indicator that would appear on some form of wearable smart glasses. This indicator would indicate to the user the direction that sound was coming from, as well as its intensity. However, we believed that such an approach would be too intrusive for the user.

The idea we settled on was using some form of haptic feedback to indicate the direction of sound instead. As discussed above our goal was to build a system that gives people with unilateral hear loss an alternative way to localize the sound around them. We think the best way to achieve this goal, without being intrusive was to use two small vibrating devices strapped to either side of the user. Instead of hearing sounds we enable them to feel sounds from different locations around them through vibrations. Depending on the direction of where a sound comes from the user will feel the sound on the corresponding side.

The device we decided on was a Samsung smart watch as that was the technology we had available to us in our lab. We had a budget of 100 US dollars to purchase additional technologies, but we though it would be best to spend this money on purchasing a directional microphone array.

We used a microphone array to detect sounds and their directions, depending on the sound source we differentiated between humans and things. These detected sounds get saved as strings and the source, volume level and the direction are then captured on the Raspberry PI. The Raspberry Pi sends that data through a web socket to the two watches. Both watches receive the signals encoded as strings and based on the side of the watch the vibration gets executed. If the sound comes from the left, the left watch vibrates, if the sound comes from the right the right watch vibrates. If the sound comes from the back, both watches vibrate. In addition to that we implemented different vibration patterns based on whether it is detecting human voices with either a loud or low volume, as well as a different vibration pattern for sounds that originate from anything that is not a human voice.

In the beginning of our project we did not know that there are microphone arrays that are compatible with Raspberry Pis that can detect sounds from different directions, different sources, and different noise levels. This microphone array is the key element of our system, because it enables us to detect and allocate sounds through the usage of just one device. This eased the whole process and gave us a better chance to finish this project within the given time frame.

Since the beginning of the project we had the idea of differentiating sounds by translating them to specific vibrations. Our thought behind that is that the vibrations should just be as unique as sounds are. A user with unilateral hearing loss should be able to feel the same sensation or at least a sensation as close as possible to the original as anyone who is able to hear sounds. This different type of sensation, namely the different vibration patterns were one of our key focus areas. We tried to make sure early on in the project that we tested different vibrations, to make sure that we have that in place.

Throughout the time span of our project we worked on three different areas, but this will be explained in more detail later on. The first areas that we worked on was receiving the sounds on the Raspberry Pi from the microphone array, as well as sending the sounds decoded as strings. The second area was the whole process of configuring the web-socket on the smartwatches to receive the signals from the Raspberry Pi. The third area was the exploration of different vibration patterns.

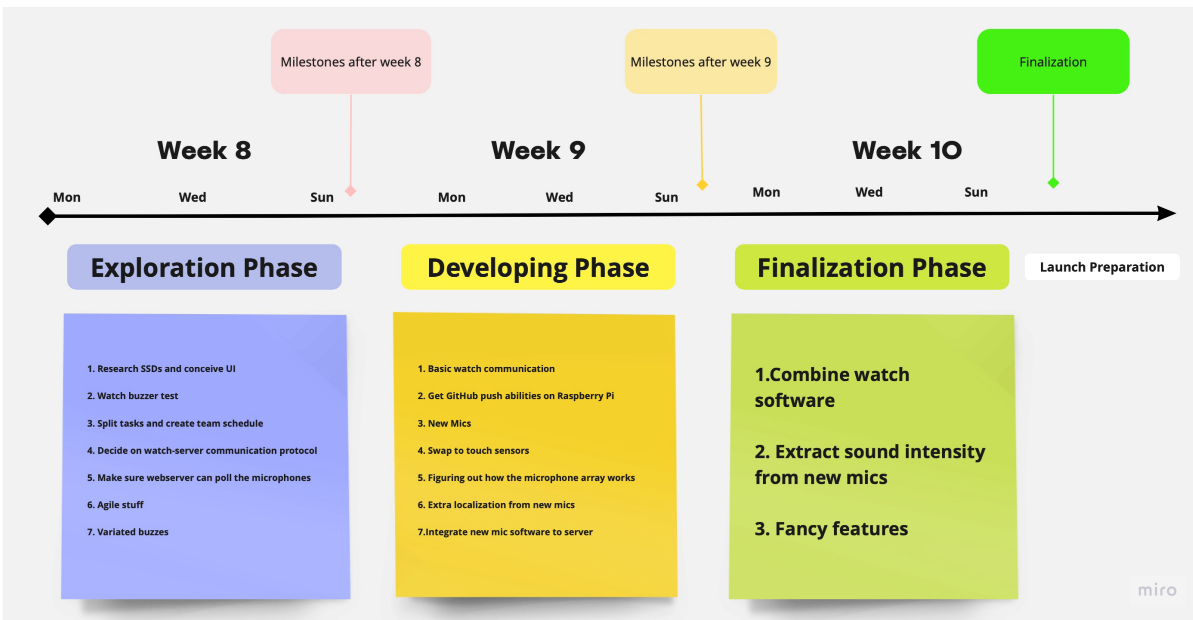


Fig. 2. Diagram of our project timeline based on our weekly zenhub reports

The next step was to merge all these areas step by step into one final system. The first task was to send the different signals as strings through a web-socket to the smartwatches. This task was about establishing a web-socket connection between the Raspberry Pi and the smartwatches. It took us some time to figure this out and to make it work as expected. The next task after properly testing that and setting that up was to link the different sounds, directions and sources of sounds to the right vibration. The key task here was to combine the two separate Android watch applications that we built and tested and two separate branches. One of the branches contained the application that was built to test different vibration patterns. The other application contained the whole functionality needed to establish a web socket connection to the Raspberry PI. Now the task at hand was bring the pieces together to make one android app out of these two separate sub-tasks. We did this initially with very basic vibrations that were the same only the origins of the sounds were different, meaning that the sides of the vibrating watch would be the only sign of differentiation.

Once we successfully tested that we further refined the overall user experience by differentiating between the source of a sound, the direction of a sound and the volume of a sound. This decisively enhanced the overall haptic feedback experience, since it enabled us to use different vibrations in different scenarios.

In 2 you can see a summary of our weekly zenhub reports all in one diagram. There were clearly three separate phases throughout our project development process. In the first week (week 8) which could be summarized as an exploration phase we did our research and we prepared for the development process. At first we had to understand and work out the steps that we needed and wanted to take. We built two sub-teams at that time, Arturo and Sammy worked on the Raspberry Pi and the sound detection with microphone array. Scott and Julian focused on the Android smartwatch application.

Through an iterative development approach with weekly sprints we were able to continuously progress towards a solution that got more precise step by step. The regular meetings and regular tests helped us to understand where we stand and where we would like to get next. We kept following this approach until we eventually merged together different separate tasks that each of us worked on into one final prototype.

4 SYSTEM DEVELOPMENT

4.1 Architecture

The system follows a client-server model, displayed in figure 3. We use two Samsung Galaxy watches as our vibration generators. They act as clients to our server. More specifically, they are websocket clients, since we want them to be in constant communication with the server, and real-time response is very important. They receive commands from the server and buzz accordingly.

The Raspberry Pi acts as our websocket server. The heavy lifting is done with FastAPI, a web framework for API-building in Python. It is in charge of talking to the two websocket watch clients and pinging them with relevant information about the sounds the microphone has picked up. It is connected to the internet using Ngrok so the watches can access it without extra steps.

Originally, the same script that controlled the websocket server also controlled the microphone. Once we tried to add logic to differentiate between loud and quiet sounds we were forced to use a separate script for the microphone. To communicate the sound information to the Pi we were then forced to either write to a file locally or add another client-server interaction. We chose to add another client for speed considerations. This new client connects to the server through Ngrok as well, except it doesn't use websockets. It simply submits a POST request with the newest sounds it has detected. They are formatted in the same way as the message that the Pi sends to the watches, which is a comma-separated string saying "HUMAN" or "THING", "LOUD" or "QUIET", and the direction ("LEFT", "BACK", or "RIGHT").

The original project proposal wanted to use Bluetooth. Since we then pivoted to using WiFi and internet connections, we didn't have enough time to implement robust security features. The clients and server exchange plain strings, although they are sent over secure connections - we use HTTPS and WSS connections as opposed to their unsecured counterparts. A lot of background security concerns are handled by Ngrok and FastAPI as well. This is a big area to address in future work, especially if this were to become an actual product.

4.2 Technology Used

As briefly mentioned above, we used three kinds of devices - a Raspberry Pi, a microphone array, and a pair of Galaxy Watches.

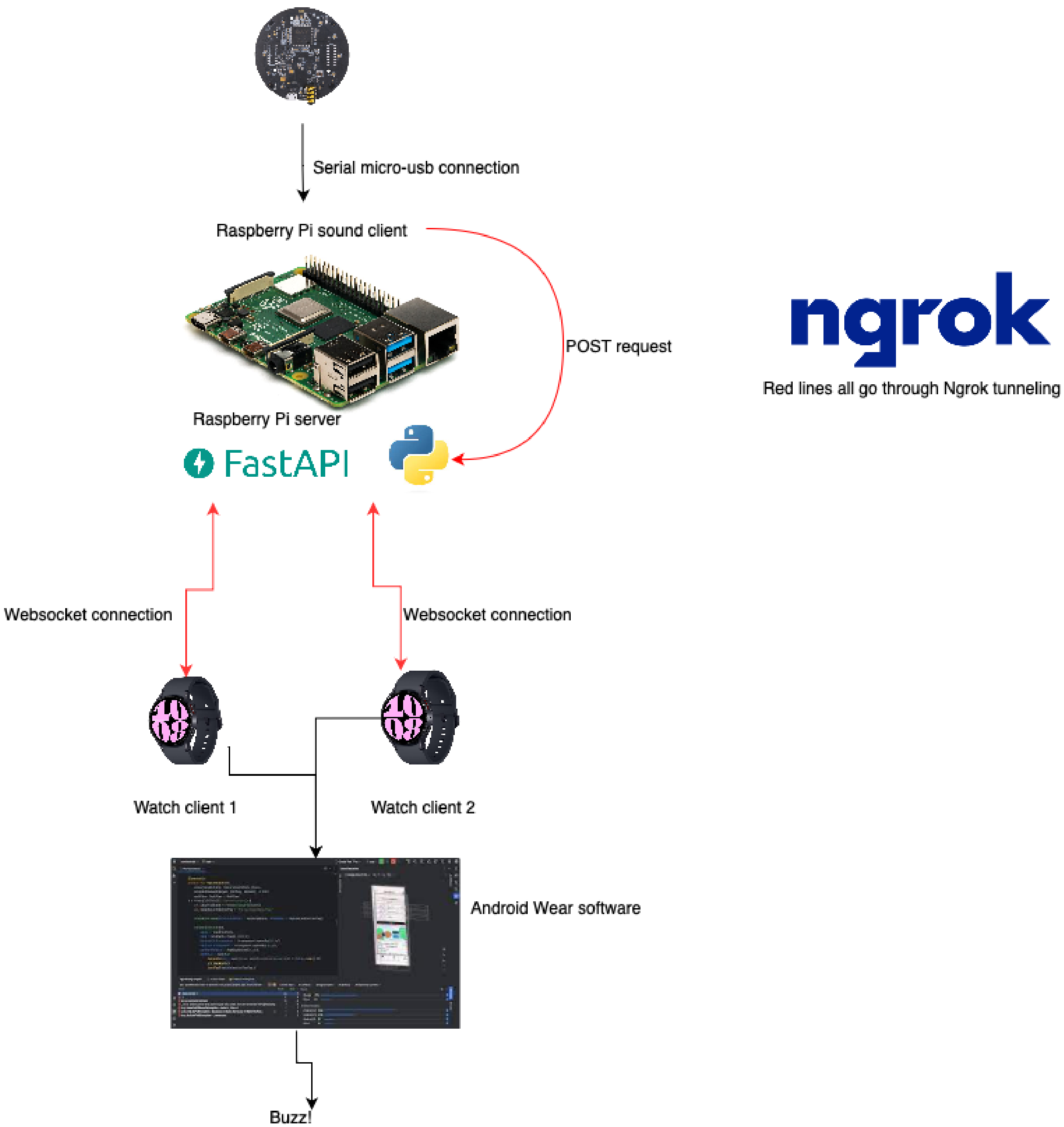
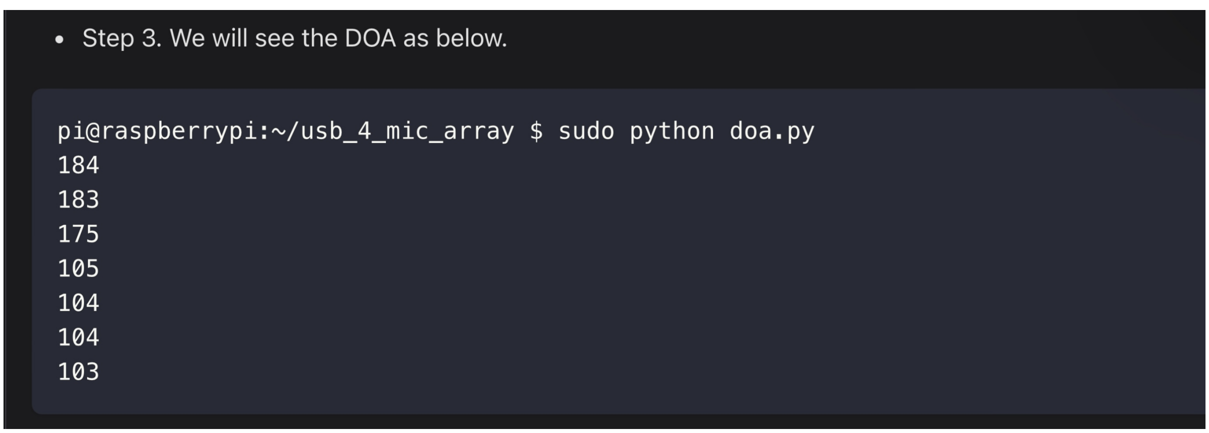


Fig. 3. Diagram of our system architecture. Microphone talks to Raspberry Pi client, which POSTs data to Raspberry Pi server. That server communicates with the watches, which buzz accordingly

4.2.1 Hardware.

4.2.1.1 ReSpeaker microphone array. The ReSepaker v2 microphone array was our microphone of choice. It is manufactured by Seed and is intended to be used for DIY projects that need smart speaker functionality. It has 4 microphones and advanced circuitry to calculate the direction of arrival (DoA) of sounds, as well as whether or not detected noise is a human voice. It is seen in figure 4 This really accelerated our development - otherwise things would have been more difficult. The manufacturers also provided a script to easily extract these two data points from the watch in Python, which was very convenient.

Fig. 4. The ReSpeaker v2 attached to backpack carrying case. The LED ring lights up to indicate the direction of detected sounds.



- Step 3. We will see the DOA as below.

```
pi@raspberrypi:~/usb_4_mic_array $ sudo python doa.py
184
183
175
105
104
104
103
```

Fig. 5. Screen capture of the output of the sample script from Seed on DoA. The numbers have no context.

That being said, there are certain drawbacks to using this microphone. While the manufacturers provided very advanced functionality on the chip itself, the documentation was not particularly detailed or helpful. The script can extract whether or not a sound is a voice, but provides no explanation as to what happens when several sounds occur at the same time, from different directions and from a mix of human and non-human sounds. Since those processes are all handled on-chip, all we see on the script-side is a number representing the DoA or a boolean value representing if a sound is a voice. This is shown in figure 5. For our current use this hasn't been a major problem, but it could easily become one. We did however have repeated instances where the microphone would misidentify the source of a particular noise - for example identifying a voice as a "thing". Without any context or explanation of how the microphone arrived at that conclusion, we couldn't really attempt to correct it. It wasn't frequent enough to be a massive problem, but it did affect the nuance of using the device.

More importantly for our uses, the microphone has a bug that has to do with volume control. It dynamically adjusts the gain it uses to process signals even when it's been set not to do so. That means that telling whether or not sounds are loud or not is somewhat difficult. Raw comparison is hard, but comparing to the last couple seconds is fine. We worked around that and simply generated an average volume level for the last couple seconds and compare new sounds to that average. What wasn't as straightforward was actually accessing the signal volume in real time. The ReSpeaker does not support it natively. We ended up using another Python package called Sound Device to access that.

4.2.1.2 Raspberry Pi. The Raspberry Pi is an excellent device for prototyping and small projects. As an established platform, there is lots of documentation and a fully functioning and full-fledged version of Python available for it. This sped up our development as well. The Pi also has a variety of I/O pins, which aided our early development with buzzers instead of the watches. It worked about as well as we could expect from a lower power device expected to read data from a microphone, make POST requests to a server, run a server, and run the ngrok tunnel.

4.2.1.3 Galaxy Watches. We used the Galaxy Watches to produce vibrations on the user's wrist. They can provide a variety of different vibration patterns and durations. We used these different vibrations to communicate different information to the user, like whether or not the source was a voice or not. The watches basically receive a string from the server containing the information about a detected sound. They check if the string contains certain keywords and decide how to buzz, as seen in figure 6. The Galaxy Watches were an interesting platform for us since we didn't have too much prior experience with Android development. Newer watches support different intensities and rumble qualities that we could explore in future iterations.

Fig. 6. The two Galaxy Watches set up to receive messages over a websocket connection. Notice that both display the message they received on the screen. In this case they both got "HUMAN, QUIET, FRONT" so they shouldn't buzz at all.

4.2.2 Software. We used two different special tools or frameworks for this project - Ngrok and FastAPI. We also used the Sound Device package in Python to access volume.

4.2.2.1 Ngrok. Ngrok was used to provide a public address that would send traffic to the Pi server. That way we avoided having to register an address of our own. It worked perfectly, aside from getting assigned a new public address everytime we turned the Pi back on.

4.2.2.2 FastAPI. We used FastAPI to simplify setting up a server on the Pi. It works with websockets as well as with regular HTTP requests, so we didn't have to do too much extra work to have both up and running. We chose it in part because the rest of the Pi software was going to be using Python (like the microphone script) and partly because of prior experience with the framework. Using FastAPI was really simple and it ran fairly quickly on the Pi.

4.2.2.3 Sound Device. The manufacturer software did not allow us to see the actual sound data at any given moment, so we used a third party Python package called Sound Device. It lets us see the information of the sampled waves at a particular instant. All we do with it right now is take the magnitude of the captured sound, i.e. the volume, to keep a running average the last couple seconds. In the future we could explore using more of that data, maybe in the context of signal processing or machine learning voice detection algorithms.

4.3 Features

The system provides basic features. The main feature is letting users know which direction sounds are coming from when they cannot see the source.

4.3.1 Direction of Arrival. The main feature is direction of arrival of sound. The microphone picks up on the direction that a particular sound is coming from. If it is not in front of the user, the sound client tells the server where it is coming from. We've separated the 270° field not directly in front of a user into three simple categories - right, left and behind, seen in figure 7. We also track sounds coming from the front of the user, but they produce no vibrations. For each case the sound client tells the server and the server tells the watches. The watch app lets the user set each watch to be either left or right on startup. Then, when they receive the message, only the correct watch will buzz. If sounds are coming from the right, the right watch will buzz. If they are coming from the left, the left watch will buzz. If they are coming from behind, then both watches will buzz simultaneously, or one after the other in a short time span. The possible latency is due to the internet connection. It is not significant enough to cause confusion, though.

4.3.2 Voice detection. One of our secondary features in voice detection. If the microphone picks up on a human voice, the noise client will tell the server and the server will tell the watches. The watches have a special sequence for voice sounds as opposed to non-voice, or "thing", sounds. They still buzz according to the DoA rules as well.

4.3.3 Sound Volume. Even when the microphone isn't picking up anything of note, we're taking that data into account. We keep a running average of the volume of the last 2 seconds of sounds. With that we can define what a loud noise is, and buzz accordingly. We defined two levels for a human sounds - quiet and loud. Relatively quiet sounds were those that were above the average. Loud human sounds were defined as more than one and a half standard deviations above the average. We only defined one level of "thing" sounds - loud sounds that were higher than the average by at least two standard deviations.

4.3.4 Buzz variation. Putting it all together, we had three different messages (each combined with the three possible DoAs) - quiet human, loud human and loud thing. Quiet human was assigned a short buzz (200ms), loud human got a long buzz (500ms) and loud non-human sound got a double buzz (200ms, 100ms of nothing and 200ms of buzz again). The watches get messages roughly every second so the double buzz isn't confused with two short buzzes in quick succession.

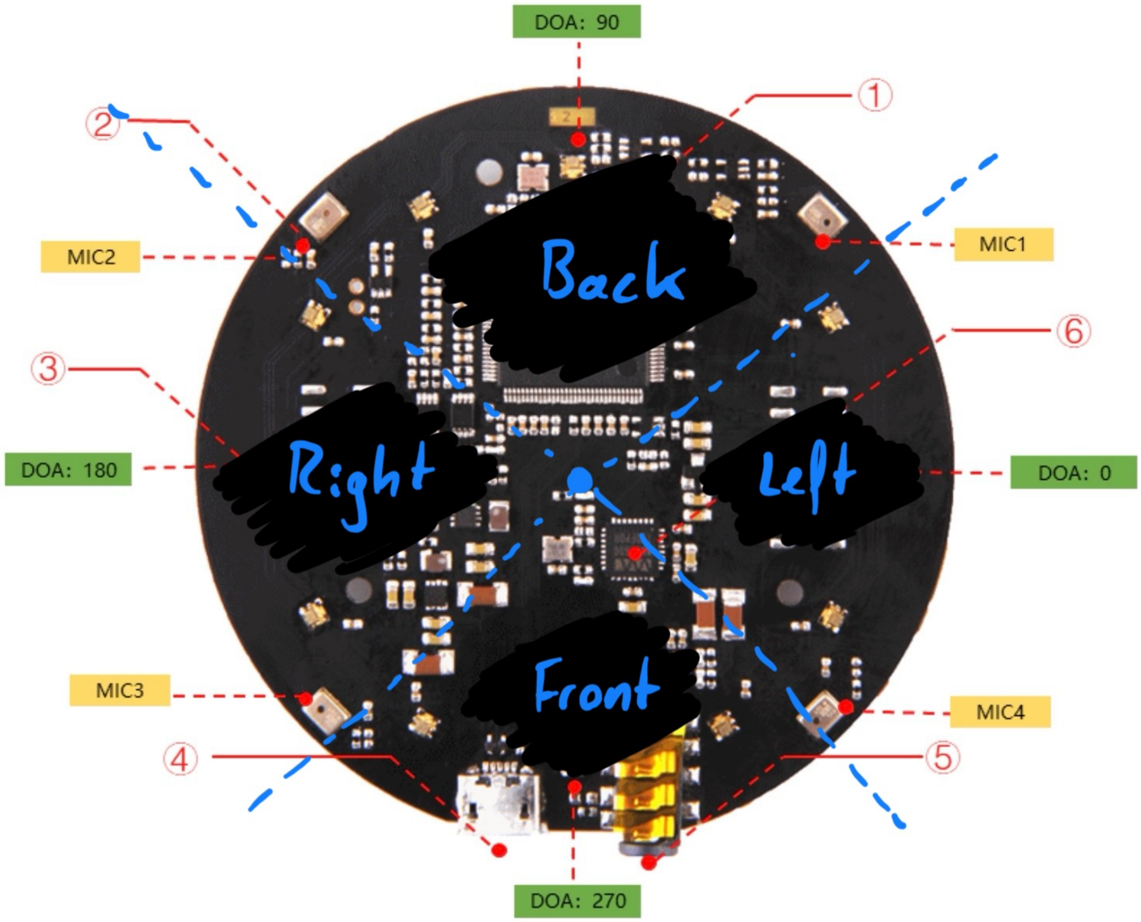


Fig. 7. A picture of the microphone with the directions that we deemed left, right, and back. The front was chosen so that the micro-usb cable is plugged in from the front.

5 TESTING AND EVALUATION

The testing of our device was fairly simple. One of our members suffers from UHL himself. So we were able to test the device directly on him to see what it would be like for an intended user to wear the device. Our testing process could also be done by others by disconnecting the microphone from the user. In figure 8, you can see one of our member holding the microphone while another wears the smart watches. By holding the microphone in one hand, we can rotate the microphone speak to it at different angles behind the wearer's back. This allows us to test the wearer's ability to tell what direction we are trying to tell him without our actual voices providing a hint.

Our tests were very informative throughout the entire development process. They revealed key information about the design of our product and how effectively it communicated information to the user. The two most

Fig. 8. A demo testing DoA capabilities. The left watch correctly vibrated.

important factors were the latency of the watches, and the clarity of the information being communicated to the user.

Because of our design we found that our product had a high latency that could range anyway between 0.5-2 seconds. We believe this was due to our system architecture. We set up an HTTP server on a raspberry pi that get to handle get requests from two different devices simultaneously. Because of the pi's limited bandwidth, there would be a significant latency when the pi needed to transmit data to both the watches, handling to open web sockets at the same time. Another flaw is that we used web sockets to connect the watches to the pi which meant that the internet was an unnecessary middle man as requests were being sent and received between devices.

We also tested to see if the clarity of information presented to the user was informative and consistent. There are a number of factors that could affect whether the user could understand the information that we are trying to convey to them. The biggest ones were: is the information being presented in a meaningful way, is the information reaching the user in a timely manner, is the information accurate. The latency of our device does affect the user's experience as the haptic feedback will be disconnected from the sound that caused it. this may confuse the wearer as to what sound the watch is alerting them of as the target user for our device still has some hearing.

However, our tests revealed a larger concern. The directional microphone we purchased had a built in speech recognition feature. But through testing we found that it was not accurate. It is important that the user find the haptic signals we send them to be meaningful. Our members found that this device takes some time to get used to. If the signals we send are inconsistent, it could prevent the user from understanding what each signal is supposed to represent. For example, we differentiate non-human speech by giving the user a double buzz instead of the typical single buzz they get when human speech is detected. If this feature is inconsistent, it makes it difficult for the user to learn what cues the different haptic signals mean.

The conclusion of our testing was that users can quite consistently discern the direction of sound. This was our initial and primary goal. Overall, the project was a success and shows promise as a real world solution. But further development is required to address the mentioned issues of latency and inconsistent speech recognition.

6 COLLABORATION

6.1 Structure of Team

Our group split into two main teams. Team 1 worked on development of the raspberry pi. This included setting up the HTTP server, setting up the directional microphone, and writing a python script to send data from the microphone through the server. Team 2 focused on development of the smart watches. Team 2 experimented with different vibrational patterns and how they could be used to convey information. They also developed an android app in java that would establish a connection with the HTTP server in the pi and programmed the logic of how the watches would vibrate given different responses from the server.

6.2 Project Timeline

Week 1:

Team 1:

- plugged in two sound sensors and simple buzzer to GPIO pins on the pi
- learned how to write python script to trigger buzzers from sound sensors detect sound
- began research into ngrok service for developing web servers

Team 2:

- began research into android development
- created simple UI for our smart watch app
- experimented with getting the watches to vibrate in different patterns

Week 2:

Team 1:

- wrote a python script to handle HTTP requests on the pi with FAST API
- set up an HTTP web server using ngrok client
- tested sending and receiving data continuously through a web socket

Team 2:

- research into how to set up web socket on an android smart watch
- began development of functionality for smart watches to connect to web socket on pi

Week 3:

Team 1:

- after arrival of Seeed ReSpeaker Directional Microphone in the mail, learned to receive directional data from the microphone on the pi
- implemented and tested speech detection functionality on the mic
- updated server script to send microphone data instead of previous sound sensor prototype from week 1

Team 2:

- finished debugging problems with establishing web sockets on the watches

  - finalized UI design for android app and added the ability to select whether the watch would be worn on the left or right

Week 4:

Team 1:
  - researched client that could read audio intensity level from microphone
  - implemented feature on server that would record average audio levels of a room to tune microphone sensitivity
  - finalized what categories of audio we wanted to communicate to the user through HTTP server: loud non-human audio, quiet human speech, loud human speech

Team 2:
  - finalized design of how to convey what vibrational patterns would convey different categories of sounds: non-human audio → double buzz, quite human speech → short buzz, loud human speech → long buzz
  - experimented with frequency of how often feedback should be given to the user through user testing

6.3 Problems Faced

One of the problems we faced was the issue with the very low battery life of our Samsung smart watches. We often found ourselves in situations where our watches had run out of charge in short amount of times and then when we tried to reload the watches that got often hindered by the watches overheating. The best way to cope with that was to constantly leave the watches on the charger, especially during the development process.

Another issue we faced was during the implementation of a web socket connection between the Raspberry Pi and the smartwatches. This task was an integral part of our overall development process. We overcame this issue through continuous testing and debugging of our implementation.

In our current solution we rely on the microphone array to detect sound from different directions, as well as the sources of sounds. One of the shortcomings of the microphone array is that the microphones do not have audio intensity. Such a feature would be really helpful when it comes to differentiating different noise levels. It could be really significant for people with unilateral hearing loss to make sure that they are fully aware of their surroundings.

7 CONCLUSION AND FUTURE WORK

UHL Aid is an assistive Sensory Substitution Device that enables users with Unilateral Hearing Loss to localize sound in their day to day life. As it currently stands, UHL Aid is functional, but not ready for daily use. Our current design has high latency, inconsistent speech recognition, and is not very portable. From the limited time we had to test our final product on a user with UHL, we believe that there is potential for a haptic based SSD. However, we would need to allow users more time with the device to see if they would get used to it over time.

The most significant improvement that would need to be made for the device to be practical in day to day life is the implementation of machine learning. Through machine learning, a more consistent and accurate speech recognition system could be implemented. One of the major pitfalls of our design is it's inconsistency in differentiating the type of sound detected, especially in noisy environments. Machine Learning could be used to make the system more robust, identify speech better, and allow the system to adapt to different auditory landscapes. This is very important for differentiating which audio is worthy of the user's attention, and what is just noise. The algorithm could also be trained to detect certain words, such as the user's name, or to learn and ignore the user's voice so that the vibrational actuator only goes off in response to relevant auditory stimuli.

On the topic of more complex computational system, portability needs to be addressed. If the device is to utilize more advanced speech recognition software, more computational bandwidth will need to be offloaded to some form of cloud server. A portable device like a raspberry pi will no longer be sufficient. With this change, we now introduce additional latency concerns as well as security risks. The connection between the wearable the device and the server will need to be secure. Alternatively, a purely local device could be implemented using wired communication with actuators and micro-controllers sown into clothing, like a LilyPad. This would solve the issue of latency and internet security but prevents making a more robust and complex system.

Whatever future developments are made, the design of a UHL Aid should prioritize user experience. The ultimate goal of the system is to assist people living with disabling hearing loss in navigating the complex soundscapes found in everyday life.

REFERENCES

[1] [n. d.]. How Do We Hear? https://www.nidcd.nih.gov/health/how-do-we-hear. Accessed: 2023-12-14.

[2] 2020. Auditory Neuropathy Spectrum Disorder: When the Audiogram Does Not Tell the Full Story. *Word of Mouth* 32, 1 (2020), 8–11. https://doi.org/10.1177/1048395020949087b arXiv:https://doi.org/10.1177/1048395020949087b

[3] Sebastian A Ausili and Hillary A Snapp. 2023. Contralateral Routing of Signal Disrupts Monaural Sound Localization. *Audiology Research* 13, 4 (2023), 586–599.

[4] David M Eagleman and Michael V Perrotta. 2023. The future of sensory substitution, addition, and expansion via haptic devices. *Frontiers in Human Neuroscience* 16 (2023), 1055546.

[5] JEC Lieu. 2013. Unilateral hearing loss in children: speech-language and school performance. *B-ENT* (2013), 107.

[6] Sabine Reinfeldt, Bo Håkansson, Hamidreza Taghavi, and Måns Eeg-Olofsson. 2015. New developments in bone-conduction hearing implants: a review. *Medical Devices: Evidence and Research* (2015), 79–93.

[7] Hillary A Snapp and Sebastian A Ausili. 2020. Hearing with one ear: consequences and treatments for profound unilateral hearing loss. *Journal of clinical medicine* 9, 4 (2020), 1010.

[8] Daniel M Zeitler and Michael F Dorman. 2019. Cochlear implantation for single-sided deafness: a new treatment paradigm. *Journal of Neurological Surgery Part B: Skull Base* 80, 02 (2019), 178–186.