Cherry: Sound Localization for the Deaf or Hard-of-hearing

Aishwarya Singh
Rochester Institute of Technology
Rochester, NY 14623, USA
axs1739@rit.edu

Alan Lambie
Rochester Institute of Technology
Rochester, NY 14623, USA
ajl5088@rit.edu

Hrishikesh Karale
Rochester Institute of Technology
Rochester, NY 14623, USA
hhk9433@rit.edu

Piyush Chauhan
Rochester Institute of Technology
Rochester, NY 14623, USA
pc8504@rit.edu

Tanmay Mahesh Songade
Rochester Institute of Technology
Rochester, NY 14623, USA
tms6649@rit.edu

Abstract
Deaf people by definition are people who do not hear well and cannot rely on their hearing abilities to interact with the environment. Unlike hearing people, they generally need to be made aware of the events occurring in their environment, such as a fire, someone trying to call them by phone or in person, a child crying, an approaching car, and so on, in their homes or workplaces. A significant amount of research efforts in assistive technology have explored these contexts for deaf people, looking for specific solutions. This project intends to build a system that is able to provide deaf people with subtle alarms of the events taking place in their environments, while they are in a mobile context.

Author Keywords
Sound capturing; sound processing; sound localization; sound visualization.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Introduction
The human brain is highly perspicacious when it comes to hearing – the cocktail party effect being the perfect

example of this, wherein a hearing person is able to focus their attention on a particular auditory stimulus while straining out the rest. This project seeks to contribute to the deaf and hard-of-hearing people, by creating a system which is able to replicate a virtual cocktail party effect of sorts.

Our prototype attempts to develop a system, which while relying on sound, would detect different events occurring in the environment. It intends to augment the awareness of deaf people with visualization techniques, providing them with enough information to be able to act and react to the events. Using a set of 3 microphones, arranged in order, to cover the left, right and back surrounding areas with respect to a user's current position, the system would be able to capture significant changes in the sound composition of the environment, providing relative location information about an event. The user would have the freedom to attend to the event or not, depending on their own holistic knowledge of the current environment.

Background and Related Work

Sound localization is a key aspect for this project. It is a natural part of our living process and is usually taken for granted. The importance of sound localization has evolutionary roots. An important aspect of survival for animals has always been to identify the location of potential predators before it is too late, and most of the times, this involves using their auditory system before their visual system is able to perceive the threat [9].

Some studies have explored sound localization, but most of them in static conditions, like the work of Liou et al. [7], where they built the Cross-power Spectrum Phase (CSP) method. This method localizes a sound as an intersection of expected sound directions, using different microphone arrangements. In particular, this study used 3 microphones, each 30 cm apart. The sound was amplified using amplifiers, signals were captured and processed on a computer, and LabVIEW software was used to display the signals for analysis.

Ho-Ching et al. [3] developed two visual display prototypes for providing contextual awareness of non-speech ambient sounds, such as a phone ringing and knocking on a door, to deaf individuals, in a working environment. The first prototype, the Spectrograph display, provided information about amplitude and pitch, whereas the second prototype, the Positional Ripples display, provided information regarding the amplitude and position of sound. Study results showed that the Positional Ripples display provided better visualization of sound with nearly 90% precision in the laboratory setting, whereas the Spectrograph detected ambient sounds with over 70% precision.

In another paper, KIM et al. [5] proposed an assistive device for hearing-impaired individuals, to help notify them of the direction of sudden loud sound occurrences, out of sight, and in their surrounding environment. The study consisted of three main procedures – detecting the time frame along with the onset of sound occurrences, reducing reflections in the detected time frame, and estimating the direction. The direction of the detected sudden loud sound was visually displayed in terms of four angular regions – front, back, left and right, with the help of four directional microphones, directing towards the corresponding angular regions, and using a modified generalized linear constrained distortionless (LCMV) beamformer. Output levels of the beamformer were

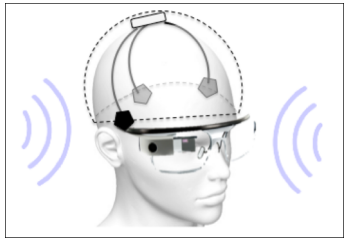

Figure 1: Capturing of a sound event. The winter hat, fitted with a set of 3 microphones, captures ambient sound.

compared and the resulting corresponding region was indicated using light-emitting diodes (LEDs).

Similarly, Gorman [2] designed a prototype – VisAural, for hearing-impaired individuals, to detect the direction of a sound event in their environment, and to notify them with the help of light-emitting diodes (LEDs), thus converting audio signals into visual cues. The prototype consisted of a pair of eyeglasses, with a mounted array of microphones for providing input, and LEDs fitted on both, left and right sides of the eyeglasses for output. Sound sampled every 0.2 seconds was checked to be beyond the array's noise threshold, and using a delay-and-sum beamforming algorithm, the angle of potential sound events was computed, from the set of selected angles (15°, 30°, 45°, 60° and 90°), both, to the left and right sides of the prototype. The resulting 10 signals were compared and the direction of the signal with the largest gain was indicated using the LEDs.

Methods

Using a set of 3 microphones, arranged in order, to cover the blind spots – left, right and back, with respect to a user's current position, the system is able to capture significant changes in the sound composition of the environment, visually providing relative location information about a 'sound event', to the user.

Sound Capturing

Our prototype consists of a set of 3 USB microphones, attached to a 4-port USB hub, and installed in a winter hat. These microphones capture the sound in a user's current environment (Figure 1). Each microphone acts as an independent input point, registering the magnitude of the sound, as well as the time.

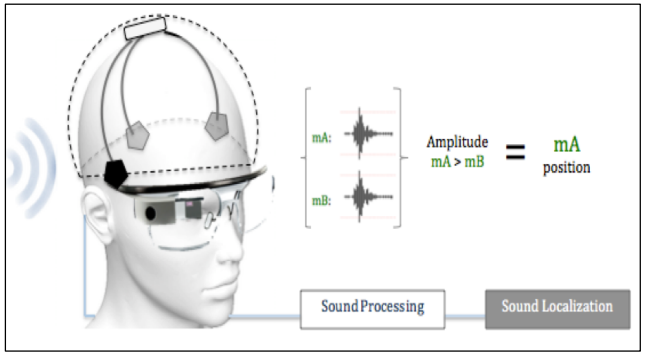

Figure 2: Processing and localization of a sound event. For example, here, the sound is captured by the microphones mA and mB; the system identifies that the sound was first captured by mA (t1 < t2), so the position of mA is the closest related to the source of the sound event.

Sound Processing and Localization

Once the magnitude of a sound is received through a microphone, the system processes the information to determine the range of noise – the threshold, using which, any sound identified over the range is isolated as a particular data point or sound event, and evaluated. The evaluation of these isolated data points is the process of interpreting them, in order to communicate them to the user. A buffer is applied to the threshold, in order to avoid sound magnitude which is very close to the threshold, and which would be considered as noise. Multiple microphones capture sound, while the system determines the sound with the highest magnitude, and registers the time of that event. Here, time is fundamental to obtain the relative location of the sound event. The first microphone to receive the sound is the one closest to the event (Figure 2).

Figure 4: Our prototype with 3 USB mini-microphones, attached to a 4-port USB hub.

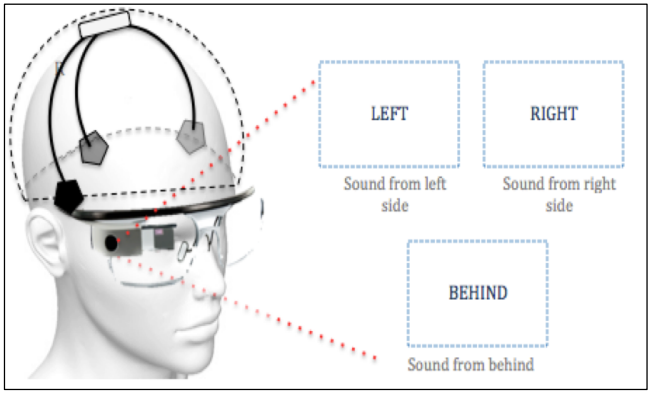

Figure 3: Visualization of a sound event. Using Google Glass as the output device, the system displays textual directional cues.

Sound Visualization

After the processing phase is complete, the 'sound event' is displayed on the output device – the Google Glass. The representation of the sound event, as captured by the microphone closest to it, is achieved using textual directional cues to cover the left, right and back surrounding areas, with respect to a user's current position (Figure 3).

Implementation

Hardware

The setup consists of 3 microphones connected to a USB hub, mounted inside a winter hat (Figure 4). The system gives a general direction of the source of sound – left, right or back, with respect to a user's current position (Figure 5).

Initially, surrounding sound is captured using the microphones for two seconds, which is then initialized to calculate a threshold that can be used to differentiate between the sound captured. A buffer of 10% is applied to the threshold, for testing purposes, in order to avoid sound magnitude which is very close to the threshold, and which would be considered as noise.

The system captures the surrounding noise in bursts of 30 s, for the next 10 s, wherein the data collected is checked for possible sound events. Keeping the threshold in check, the system then uses the next 2 s to adjust itself and re-calculate the threshold, which can be used for the next 10 s. This way, the threshold is calculated every 10 s, in order to consider changes in the user's surrounding. In case of an event, values from all three microphones is compared, and the microphone with the highest magnitude is considered to be the one facing the source of sound.

THRESHOLD CALCULATION AND SOUND EVENT

For each microphone, the surrounding sound recorded for 2 s is broken down into 10 groups, where peaks from every group are considered for calculating the threshold. The mean of the peak values, thus obtained, is considered to be the threshold for a particular microphone. The process is repeated for the remaining two microphones, and the final threshold value is calculated, as the mean of the thresholds for all three microphones. An event is considered, if and only if, values from all 3 microphones break threshold for the given time (Figures 6-7).

Software

Output of the MATLAB source code is stored in a JSON file, which is updated every second. The JSON file is a common resource between the MATLAB and JavaScript source codes.

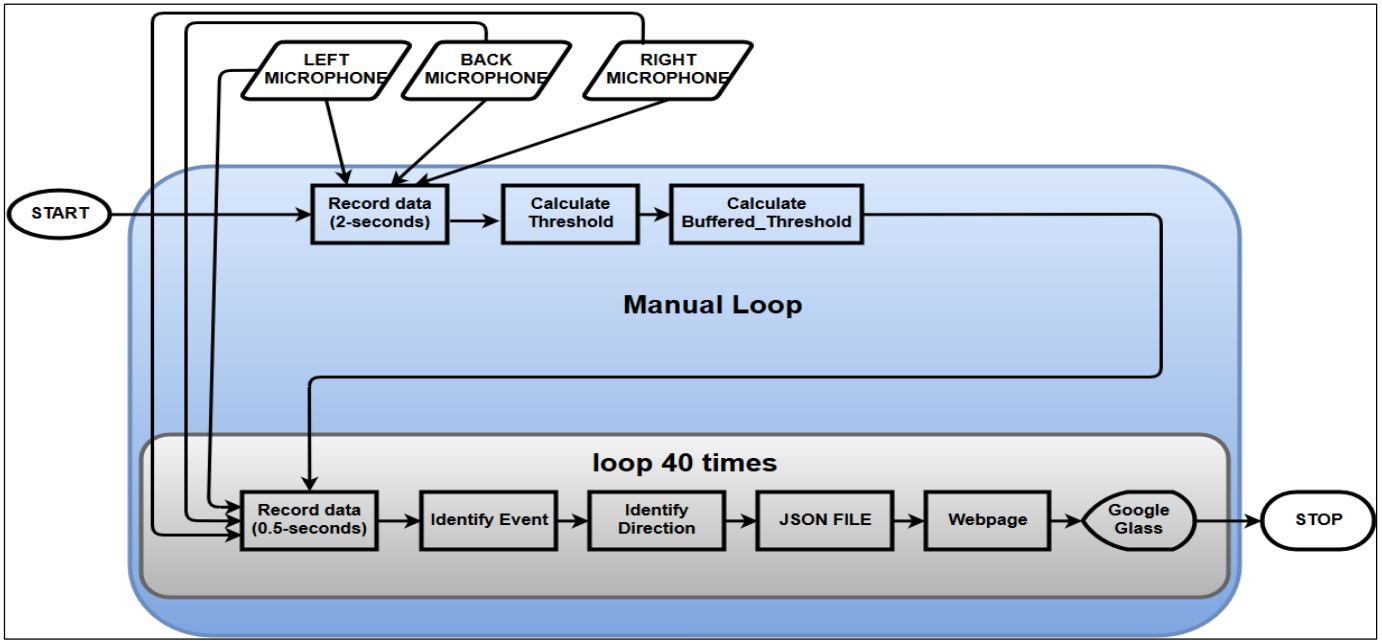

Figure 5: Flowchart representing the implementation of the proposed system 'Cherry', integrated with Google Glass.

JavaScript reads data from a file and stores it in a database. We are using the MongoDB database, as it supports the JSON format, in contrast to the structural SQL database. Once data is stored in the database, it is sent to the browser through socket.io, and gets uploaded on the webpage using the index.html file.

We are creating a web application using the MEAN stack, which facilitates real-time transfer of data from MATLAB to the browser. We are also using JSONlab, which is a free and open-source toolbox, for encoding/decoding JSON files in MATLAB. It can also be used to convert a MATLAB data structure, such as array, struct, cell, struct array and cell array, into a JSON formatted string. Once the data has been converted from MATLAB to JSON format, and is stored in a file, we can read this file to obtain real-time changes, independent of MATLAB.

When new data is received, a new instance of that schema with the latest data is created and saved in the database using the save() method.

Resources

Hardware

We are using a set of 3 USB mini-microphones attached to a 4-port USB hub. Initially, we tested the system with two microphones, and later added the remaining one microphone. The hardware specifications are as follows:

MICROPHONES

- Bluetooth v2.0 and v1.2 compliant,
- Supporting profiles: Networking, Dial-up, Fax, LAN Access and Headset,
- USB Interface,
- Size: 2.2 cm x 2 cm,
- Symbol rate: 3.0 Mbps,
- Range: 20 m,
- Supported Operating System: Windows 98, 98SE, ME, 2000 and XP.

HUB

- USB 2.0,
- 2′ USB A cable,
- 4 USB A jacks.

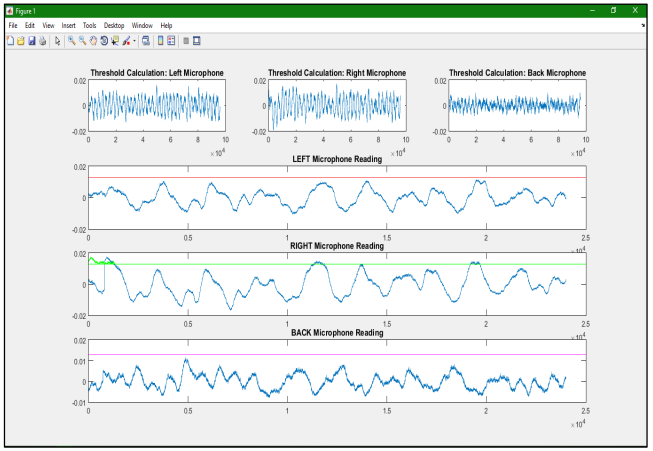

Figure 6: Values below threshold, in case of an environment where there is no external noise (here, it is all noise). The red, green and pink signals are used to indicate the magnitude of audio signals above threshold, plotted for visual purpose.

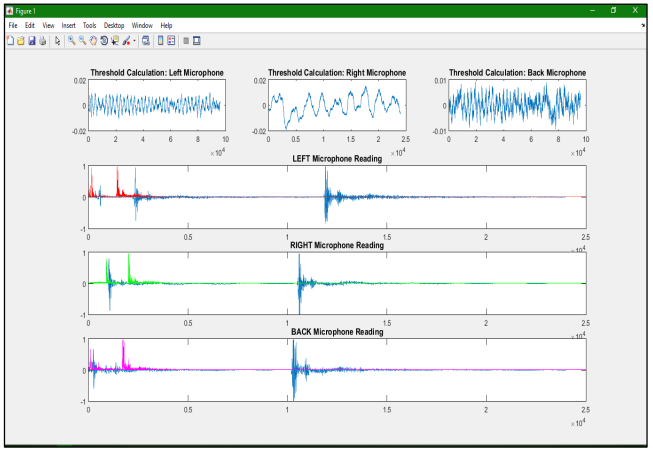

Figure 7: Detection of sound event, in case of multiple claps.

Algorithm

```
Identify microphones
Initialize microphones for recording
Loop 5 times (55 s worth total audio recording in this loop)
{
    Threshold Calculation

{
        Record data from all three microphones for 2 s
        Plot recorded data
        Calculate threshold to be used for the next 10 s of audio recorded data
    }
    Loop 20 times (10 s worth total audio recording in this loop)
{
        Process microphone data
{
            Record data from all three microphones for 0.5 s
            Plot recorded data
            Identify events
            Classify them (high-medium-low)
            Estimate direction of sound
            {
                Compare magnitudes (data values) of audio signal during an event
                Microphone with highest value is assigned as the one facing the general direction of sound
            }
        }
    }
}
```

Resources

Software

MATLAB FUNCTIONS

- audiodevinfo(IO, ID, Fs, nBits, nChannels) – to get information about an audio device,
- audiorecorder(Fs, nBits, nChannels, ID) – to create an object for recording audio from an audio source,
- pause(n) – to stop MATLAB execution temporarily,
- getaudiodata(recorder, dataType) – to store recorded audio signal in a numeric array,
- figure(h) – to create a figure window,
- subplot(m, n, p) – to create axes in tiled positions,
- plot(Y) – to create a 2-D line plot, and
- hold – to retain current plot when adding new plots.

JSONLAB FUNCTION

- loadjson() – to convert a JSON string into the related MATLAB object.

Overcoming the Challenges

Coming up with a way to calculate threshold had been an issue from the start. Initially, we calculated the mean value of the sample signal, in order to calculate the threshold. We found that the calculated threshold value was very low, due to the presence of more values on the lower side of the signal. We needed a threshold value that would do justice to the environmental noise, thereby eliminating most of the noise, for which the threshold had to be defined on the higher side and closer to the peaks.

We then came up with a method used in data mining, wherein data is divided into groups and a value from every group is selected, in order to understand the sample. Thus, for our final setup, we selected peak values from all the groups and calculated their mean, in order to get the required threshold.

The primary challenges that we faced were to get MATLAB to write data into the JSON file located on the server, and to integrate the system with Google Glass. Due to the limited Google Glass API documentation available, and unsuccessful attempts of screen sharing a laptop with Google Glass, we decided to load a dynamic website on Google Glass.

For providing directional cues via Google Glass, we created a public URL which could be accessed within the same network. We used the localtunnel npm module, in order to generate a unique URL, that could be shared with anyone, as long as the local instance remained active.

On the other hand, for integration with Google Glass, we installed a QR Code scanning application on the Glass, and scanned the QR Code, in order to navigate to the particular URL.

Result

Successful implementation of the project includes detecting an anomaly in surrounding sound, and identifying its source. The system understands the mobile nature of its user, that is, it is able to detect changes in a user's environment, and adapt evaluation parameters to new context, with 70% accuracy. The system dynamically re-evaluates the range of noise and subsequently determines the new threshold. This process of re-evaluation remains persistent, in order to detect the variations in sound composition arising from current environment changes (addition or subtraction of sound events), or changes in the context (the user moving from one place to another).

Future Work

For our future work, for sound visualization, we will try to achieve the representation of sound events using color-coding and frequency of blinks, wherein a 'low' sound event, barely above the threshold, will be represented by green color and a low blinking frequency; a 'high' sound event, significantly above the threshold, will be represented by red color and a high blinking frequency; and a 'moderate' sound event, in the middle range of the threshold, will be represented by yellow color and a moderate blinking frequency. The relative localization of sound events on the Google Glass will be represented using a set of physical orientations – left area of the display for a sound coming from the left, right area of the display for a sound coming from the right, and bottom area of the display for a sound coming from behind.

References

1. Byrne D. and Noble W. 1998. Optimizing sound localization with hearing aids. *Trends in Amplification*, 3(2), 51-73.
2. Benjamin M. Gorman. 2014. VisAural: A Wearable Sound-Localisation Device for People with Impaired Hearing. In *Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '14),* 337-338. http://doi.acm.org/10.1145/2661334.2661410
3. F. Wai-ling Ho-Ching, Jennifer Mankoff and James A. Landay. 2003. Can you see what I hear? The Design and Evaluation of a Peripheral Sound Display for the Deaf. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*, 161-168. http://doi.acm.org/10.1145/642611.642641
4. Larry E. Humes, Sylvia K. Allen and Fred H. Bess. 1980. Horizontal Sound Localization Skills of Unilaterally Hearing-Impaired Children. *Audiology* 19: 508-518.
5. Ki-Won KIM, Jung-Woo CHOI and Yang-Hann KIM. 2014. Detection and direction estimation of a sudden loud sound for the hearing assistive eyeglasses. *Inter-noise*, 1-10.
6. Xiaofei Li, Miao Shen, Wenmin Wang and Hong Liu. 2012. Real-time Sound Source Localization for a Mobile Robot Based on the Guided Spectral-Temporal Position Method. *International Journal of Advanced Robotic Systems*, Vol. 9, 1-8.
7. Jin Fu Liou, Kunal Joshi and Gaurang Vador. Real-Time Sound Source Localization. Retrieved February 5, 2016 from http://s.eeweb.com/members/kunal_joshi/projects/2011/03/13/EE586_Report-1300060274.pdf
8. Kazuhiro Nakadai, Hiroshi G. Okuno and Hiroaki Kitano. 2002. Real-Time Sound Source Localization and Separation for Robot Audition. In *Proceedings of the IEEE International Conference on Spoken Language Processing (ICSLP '02)*, 193–196.
9. William Noble, Shaune Sinclair and Denis Byrne. 1998. Improvement in Aided Sound Localization with Open Earmolds: Observations in People with High-Frequency Hearing Loss. *Journal of the American Academy of Audiology*, Vol. 9, 25-34.
10. Jeffrey Powers. 2013. Use Multiple USB Microphones to Record [How to]. Retrieved February 27, 2016 from http://howtorecordpodcasts.com/multiple-usb-microphones-record-how-to/
11. Glass, Google Developers. 2015. Reference. Retrieved March 11, 2016 from https://developers.google.com/glass/develop/gdk/reference/
12. MathWorks. 2016. Live Direction of Arrival Estimation with a Linear Microphone Array. Retrieved February 7, 2016 from http://www.mathworks.com/help/audio/examples/live-direction-of-arrival-estimation-with-a-linear-microphone-array.html;jsessionid=0dbebf59bbc267cdcfc7c7cfab4e?s_tid=gn_loc_drop
13. MathWorks. 2016. Connect to FTP server. MATLAB. Retrieved April 14, 2016 from http://www.mathworks.com/help/matlab/ref/ftp-class.html
14. Aishwarya Singh, Alan Lambie, Hrishikesh Karale, Piyush Chauhan and Tanmay Mahesh Songade. 2016. Cherry: Sound Localization for the Deaf or Hard-of-hearing. HCIN-722 Human-Computer Interaction with Mobile Devices, Rochester Institute of Technology, Rochester, NY, USA. https://github.com/SongTanmay/sound-localization-visualization