Physio-Acoustic Simulator

Scott Alspach
Project Member

1+360-620-7129
salspach@hotmail.com

Trevor Head
Project Member
4744 12th Ave NE #209
Seattle, WA 98105
1+509-220-3300
titohead@gmail.com

Rob McClure
Project Member
5000 25th Ave NE
Seattle, WA 98105
1+703-740-7251
rmcclur@uw.edu

ABSTRACT
Our project, the physio-acoustic simulator, is a piece of software designed to replicate what people with hearing disabilities hear to those with normal hearing. Many people don't truly understand what it means to not be able to hear certain things, and our software enables that. People who want to understand what their loved ones hear, or hearing impaired wishing to show others what the world sounds like to them, would both find it useful.

1. INTRODUCTION
The purpose of our project is to increase the awareness of how people with hearing disabilities experience the world. There are three target groups for our project. Firstly, for hearing impaired who wish to show others how they hear. Secondly, for parents who need to decide whether to get their children cochlear implants, our cochlear implant simulator can make sure they know as much as possible about how their child will hear with the devices. Thirdly, for general use by people who just want to know what life would sound like if they were hard of hearing.

2. RELATED WORK
Qian-Jie Fu, Ph.D, has developed a program similar to ours, called TigerCIS, which can be downloaded for free online. [2] Fu's simulator features a more robust CI simulator than ours, but his hearing loss simulator is comparable to ours. Aside from TigerCIS, however, we have been unable to find any other comparable hearing loss/ CI simulators. There are websites that give example soundclips of CI/hearing loss representations, like those found at Hear-it.org, [3] but the user is unable to customize the experience, having to use pre-chosen example sounds, and limited, general examples of hearing loss.

3. DESCRIPTION OF THE SOLUTION
Our project is made up of a variety of components. These include a simple audiogram, filter controls, a cochlear implant simulator, and a filter which can be used with any of these filter control components. All of these components are combined into one graphical user interface (GUI). One view of this GUI is given in Figure 1.

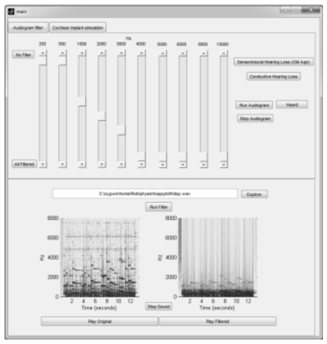

Figure 1. The application GUI

3.1 Audiogram
The audiogram is probably the first feature that many users will experiment with. It is displayed in a prominent place on the GUI and is easy to understand and use. When a user clicks on the audiogram button a message is displayed giving them instructions for taking the audiogram. The audiogram is written so that it will play a certain frequency of sound at an increasing volume until the user acknowledges that they can hear it. The audiogram then sets the filter control for that frequency to the amplitude at which the user first said they could hear the sound. After taking the audiogram the user is able to immediately load a sound file and run the filter.

3.2 Filter Controls
At the top of our GUI are a number of sliders which are used to control the filter which the user can use to edit audio files. When the sliders are moved to the top, then nothing is filtered out of the sound, when they are moved to the bottom the entire sound is filtered out. There are also sample audiograms which the user can select to automatically set the sliders to values which are characteristic of various types of hearing loss.

3.3 Cochlear Implant Simulator
The other main feature of our project is the Cochlear Implant Simulator. If a user selects the tab for the cochlear implant simulator they will be presented with inputs for the number of channels they want and the number of maxima they want, along with how they want to partition the channels. The number of channels indicates how many different frequency groups the audio signal should be broken up into. The type of partition the user selects will determine if the audio file is broken up into channels linearly or exponentially. The number of maxima the user selects determines how many of the channels will be selected, based on the maximum power of their signal, to recompose the filtered audio file.

3.4 Audio Filter

The main function of our project allows users to load an audio file and filter it using a filter created by one of the previously listed mechanisms. Users can explore their file system for a .wav file to load into the system. They can then click the "filter" button, and the audio will be put through the filter we have developed based on their inputs. The Spectrogram (showing the frequency amplitude by time) of the original audio and the modified audio will be shown to the user. They will also be able to play both the original and filtered sound files in order to hear the difference between the two.

Figures 2 and 3 show an application of this audio filter. Figure 1 shows a spectrogram of the audio file before any filtering is performed.

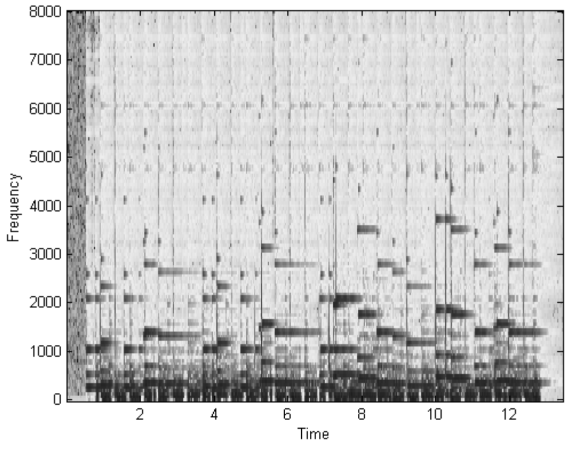

Figure 2. Spectrogram of the unfiltered audio

Figure 3 shows a spectrogram of the audio after it has been run through the filter. In this example, the filter has removed all frequencies that are higher than 3750 Hz.

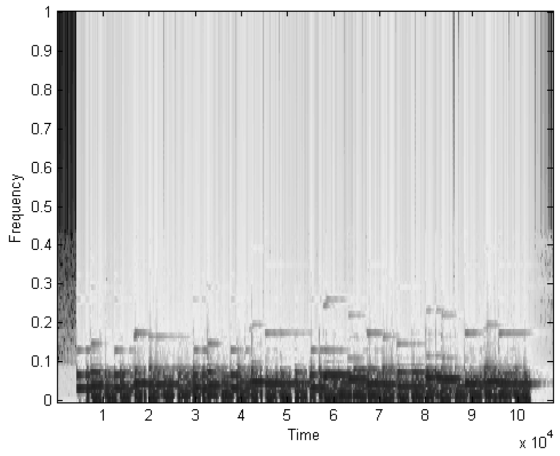

Figure 3. Spectrogram of the filtered audio

Note that in Figure 3, the frequencies shown on the y-axis have been normalized to the maximum frequency, which is 8000 Hz. Thus, the figures show that frequencies above 3750 Hz (~0.47) have been removed from the waveform.

3.5 Filter Method

3.5.1 Filter Description

Both types of audio filters are implemented in generally the same way. The audio file being used is loaded and run through a phase vocoder, which has been implemented in Matlab can be found at [1]. The implementation performs a series of short-time Fourier transforms (stft) on the audio, resulting in the power of a number of frequency bands over the time course of the audio file. These powers are then manipulated based upon the given filter.

3.5.2 Audiogram Filter

For the filter resulting from the audiogram, the frequency bands from the stft are scaled depending on the results of the audiogram. For example, if the sounds played at 4000 Hz was never heard during the audiogram, then the powers of the frequencies around 4000 Hz would be set to zero for all time steps. Scaling factors between zero and one are also possible.

3.5.3 Cochlear Implant Filter

For the filter resulting from the cochlear implant simulator, each time step is treated independently. Based on the partitioning scheme given for the simulation (as described in 3.3), the frequency bands from the stft are grouped into channels. The peak power of each channel is calculated, which is then used to sort the channels by maximum power. Based on the settings, a certain number of these maxima are selected, and the frequency corresponding to the center of the frequency channel is assigned that peak power. For example, if four maxima were selected, then the final frequency distribution for that time step would have four non-zero entries.

3.5.4 Filter Result

Finally, the results of the filtered stft information is run through an inverse short-time Fourier transform (istft), which recreates the audio data. This data can then be played to explore how it differs from the original audio.

4. CONCLUSIONS

Success of the application was evaluated subjectively by the users. One of the authors has profound hearing loss and was able to identify when he could or could not hear something or tell the difference between two sounds. Also, all users with normal hearing were able to recognize a difference when the filters were run. The cochlear implant simulation compared favorably to similar demonstrations found on the Internet in terms of the general tones and accuracy of the resulting audio.

Unfortunately, many aspects of the application were difficult to rigorously test for success. Ideally, the filters would accurately produce sound so that the filtered audio would sound the same to people with normal hearing as the original audio sounded to the person with hearing loss. It was beyond the scope of our project to quantify what the two groups of people heard and then compare them for similarity.

5. FUTURE WORK

There are a few aspects of the application that could be improved upon in future work. Currently, the way that the cochlear implant simulation is implemented results in a noticeable amount of buzz in the recreated audio, especially in the lower frequency ranges.

Also, modern cochlear implants now have methods of filtering out background noise, which could be added to the application.

The application is an open-source project released under an MIT license. The code and more information about the project can be found at http://code.google.com/p/physio-acoustic-simulator/.

6. REFERENCES

[1] D. P. W. Ellis. 2002. A Phase Vocoder in Matlab. http://www.ee.columbia.edu/~dpwe/resources/matlab/pvoc/.

[2] Qian-Jie Fu, Ph.D. 2006. Cochlear Implant Simulation and Hearing Loss Simulator. http://www.tigerspeech.com/tst_tigercis.html.

[3] "Soundfiles - Impressions of Hearing Loss and Tinnitus: Hear-it."http://www.hear-it.org/page.dsp?area=244.