RIBBIT: A Cost-Effective iOS Hearing Aid App

TCU College of Science & Engineering
reSEaRCh Science and Engineering Research Center

Authors: Duy Dang, Robert Kern, Esteban Kleckner

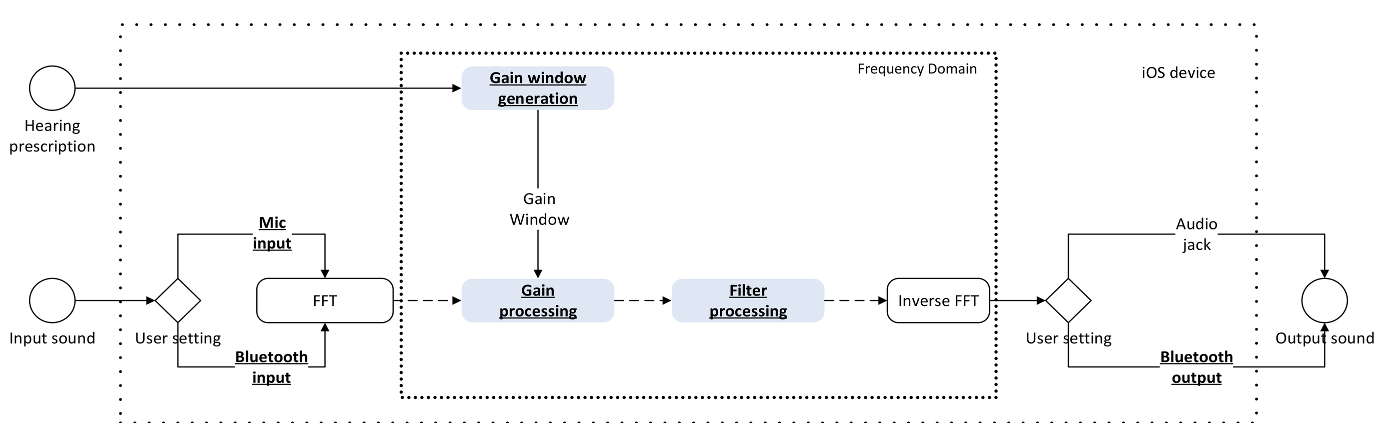
Faculty Advisors: Liran Ma Ph.D. (CS) and Donnell Payne Ph.D. (CS)

Background

Motivation:
- To provide a solution for hearing-impaired individuals who cannot afford a traditional hearing aid

Goals:
- Design and implement an affordable, and user-friendly hearing aid on Smart Devices, such as an iPhone

Ribbit Architecture



- Two inputs to the iOS app: sound and hearing prescription
- Sound transformed from the time domain to the frequency domain with Fast Fourier Transform (FFT)
- Based on the prescription, appropriate gain and filtering algorithms are applied in the frequency domain
- Sound transformed back to the time domain with the Inverse FFT and output

QR-Based Prescription Generation

Online hearing prescription to QR-code converter

Prescription Data

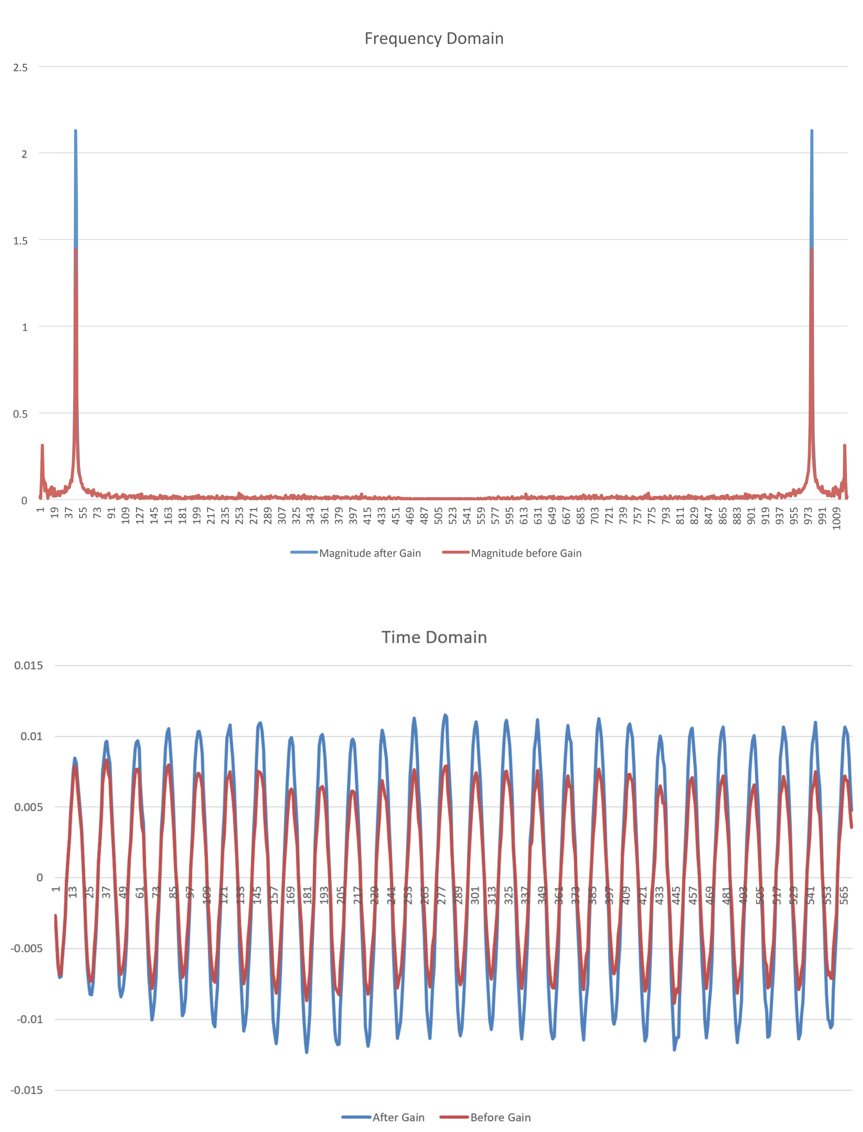
Mode	Indoors
Patient	Bob Marley
Prescriber	Frank Sinatra
Date	3/30/16
Debug	True

	Left Ear	Right Ear
[125]	5	5
[250]	20	15
[500]	20	20
[1000]	20	25
[2000]	36	40
[4000]	70	65
[8000]	80	75

Generate QR

- A convenient, secure method to load prescription
- QR code generation is done in a browser
- Read in by the phone, using the camera

Gain Processing

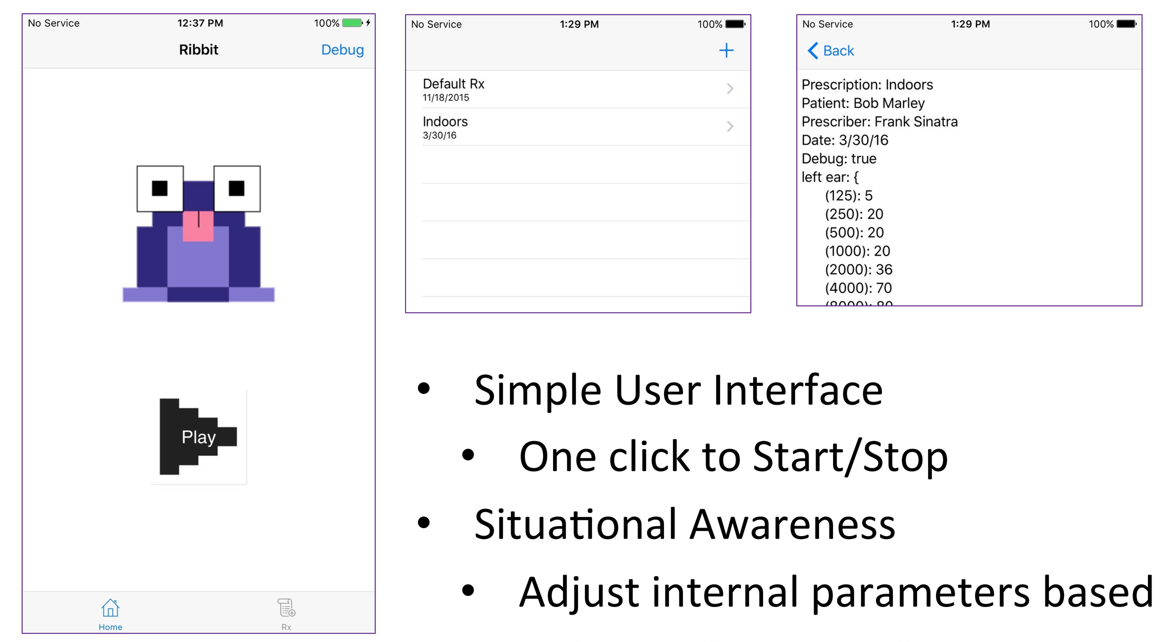


- Hearing-impairment- loss of the ability to hear/detect some (usually higher) frequencies at the normal loudness
- Gain requirements:
  - Boost a certain frequency by a specific loudness level (dB)
  - Calculate the dB values to increase each individual frequency
- Gain processing steps:
  - Compare the prescription of a normal-hearing person to that of the user
  - Generate a gain window based on the above comparison
  - Multiply the signal buffer by the gain window

Filter Processing

- Filtering the input sound removes noise and sound artifacts that otherwise impact the user's ability to fully understand/comprehend speech
- The filtering process utilizes the vDSP framework; Apple's Digital Signal Processing library. Specifically, the Fast Fourier Transform (FFT)
- The FFT is used to break the signal into its constituent parts, then apply a filter window and recombine these parts into a coherent signal
- Left top: An example filter window designed to reduce amplitudes between 4 and 24 kHz—a range that doesn't provide information necessary for understanding speech
- Left middle: An example of frequencies that would be removed from an input signal
- Left bottom: The differences between the input and output signal

The Ribbit Interface



- Simple User Interface
  - One click to Start/Stop
- Situational Awareness
  - Adjust internal parameters based on different "scenarios"

Conclusions and Future Work

- Using a Smart Device as a hearing aid shows promise
  - The app is able to finish processing sound samples within 22ms
  - Sound quality achieves acceptable levels
  - By tailoring the application to the user, we offer a more personal experience
- Future enhancements would include:
  - The elimination of sound artifacts and the inclusion of smart technologies, such as location sensing
  - The opportunity to test our application in a setting where human trials are possible

Acknowledgements

The team would like to thank Dr. Payne for pushing us to reach our end goal, Dr. Ma for giving us a challenging project to work on, Dr. Richardson for helping us through the quagmire that is math, and Katie Grebel for helping with all of the work that was needed to get this project off the ground and heading in the right direction. We would also like to thank Dr. Watts and Tracy Burger from the Miller Speech and Hearing Clinic for helping us understand what goes into audiograms.

References

Accelerate vDSP Reference - https://developer.apple.com/library/prerelease/tvos/documentation/Accelerate/Reference/vDSPRef/index.html#//apple_ref/doc/uid/TP40009464
De Gotzen, A., Bernardini, N., & Arfib, D. (2000). TRADITIONAL (?) IMPLEMENTATIONS OF A PHASEVOCODER: THE TRICKS OF THE TRADE. Proceedings of the COST G-6 Conference on Digital Audio Effects Verona.
iOS platform audio mechanism reference - https://github.com/ooper-shlab/aurioTouch2.0-Swift
Laroche, J., & Dolson, M. (1999). IMPROVED PHASE VOCODER TIME-SCALE MODIFICATION OF AUDIO. IEEE Transactions on Speech and Audio Processing (pp. 323-332). IEEE.
McLeod, P., & Wyvill, G. (2005). A SMARTER WAY TO FIND PITCH. Proceedings of International Computer Music Conference.
Swift 2.1 Documentation - https://developer.apple.com/swift/