HRTF Individualisation Based on Localization Errors in 3D Space

Specification

Robin Yonge

Febuary 2017

Virtual realityComputer-mediated reality [**?**] has come to represent the dominant prediction for the near future of human-computer interaction. If this prediction is to come to pass, then it is necessary that sufficient consideration is given toward the fidelity of both the visual and the auditory aspects of VR and AR technology. Currently, most implementations of spatial audio in VR and AR applications are based around the use of HRTFs. Mostof these implementations use one of a few publically available datasets, derived from measurements performed on either human subjects or a model such a the KEMAR[**?**][**?**]. For the use of VR/AR to become commonplace the experience must be universal. Because spatial audio relies on complex measurements of real-world subjects (be they flesh and blood or not) applications tend to use average or neutral HRTFs, ones that should work for the largest number of people. Of course, by dint of being average there are many people for whom these do not work. In order to achieve a universal experience, then, it is necessary to find a process for simple individualization of spatial audio that is sufficiently simple, and can be performed by the end user as a part of the standardsetup of any VR/AR device or application.

p1

The goal of this project is to produce a working prototype of an application that takes as input a neutral set of HRTF measurements, and returns an individualised set. For a process like this to achieve widespread use it must be simple enough that it can be performed by any given user with nothing more than a head-mounted display, and intuitive enough that it requires no in-depth knowledge of spatial audio. The output HRTF set should also provide a noticeable improvement in the ability of the user to locate sound sources in a virtual 3D environment.

notes:

hrtf individualisation and synthesis methods through the ages? structural model, other weirder ones? Why are they bad?? Most recently PCA/PCW, this is what I will use, why? why adjust not synthesise?

on what basis will we make our adjustments? It will be done on the basis of localisation errors from the user. Depending on the nature of the error, we will adjust PCWs and re-test.

how do we record the nature of their localisation errors? We do it in-VR of course. Give the user a reticle and ask them to point it at the position in space that they think a given sound source eminated from. Use that data to adjust HRTFs and feed it back in! Play another sound, note error, adjust again, rinse and repeat. Informed guessing possibly on the basis of the structural model? Not totally blind. Float the idea of incorporating ML in future build.

How do we make this fun/enjoyable? Build it in an interesting 3D space, room of shelves all covered in alarm clocks? locate the clock that's beeping?

End product will be: A summary. 3 pieces, the hrtf databases, the processing module, and the VR frontend!

p2

what technologies will I use and why?

Building the core hrtf-modifying module in Python and SciPy, that should be all that needs. Would like this module to have a nice sort-of API, to demonstrate the ease with which it could be incorporated into another product and to make it easier for me to build stuff on top of it.

The database used looks like it will likely be the CIPIC KEMAR set, because it's more well-documented than the other newer databases, and of a higher resolution? Also looks like the AIR and IRCAM ones don't have a mannequin head version and using a human dataset seems like a bad starting point?

The UI/frontend I'm still not 100percent on, I know Unity and it would be a weekend's build time, but I'm not sure I can necessarily get the data I need from it out to my python module and I don't know if building it as a Unity plugin is an option? Either that or I learn me some simple OpenGL.

When it comes to processing the audio in the scene itself I think I'll just use something simple like fmod because simpler with a spatial audio/convolution plugin because I don't *think* I want to fuck with real DSP. ITD/ILD/filtering is all contained in the hrtf so the audio approach doesn't need to be complex.

For rudimentary testing I'll record a few sample angles/elevations with the binaural microphones on myself and then test them against the individualised hrtfs for the same directions. After that I can test it purely based on the error rate of the users in the final product.

Work plan, timeframes, etc

Minimum Viable Product: Python hrtf-editing module, with a blank void VR frontend that's full of sound sources.

Ideal: Python hrtf-editing module with visualisations, nice fancy VR frontend that turns it into a game of locate-the-object.