HEAD-RELATED TRANSFER FUNCTION RENDERING AND ADAPTION

SEAN BOYD
University of Victoria
Electrical Engineering
July 2007

ABSTRACT

This document will discuss the implementation and results of the ELEC 484 Final Project. It will discuss the goals achieved, the work left to be done, and the operation of the written software. As well a listing of all code will be located at the end of this document.

M files, sound files, and a PDF of this report can be found at: http://web.uvic.ca/~seanboyd/

1. INTRODUCTION / MOTIVATION

Most recorded sound and music is recorded with the intent that it will be played back on loudspeakers. It is for this reason that listening to music on headphones or earphones does not yield the effect desired by the audio engineer who preformed the final mix on the recording. The audio engineer performing the mixing most likely was using speakers to monitor the sound while making changes. When we listen to this music with headphones or earphones, the paths of the sounds are isolated, and crosstalk is eliminated. As well, the effects of the listener's body are eliminated. By processing the recorded sound, we can artificially induce these effects so that the music will sound as though it is coming from loudspeakers, and not headphones or earphones.

2. PROGRAM STRUCTURE

The program contains a model of the hearing system as outlined in the Pinna-Head-Torso system as illustrated in Figure 1.

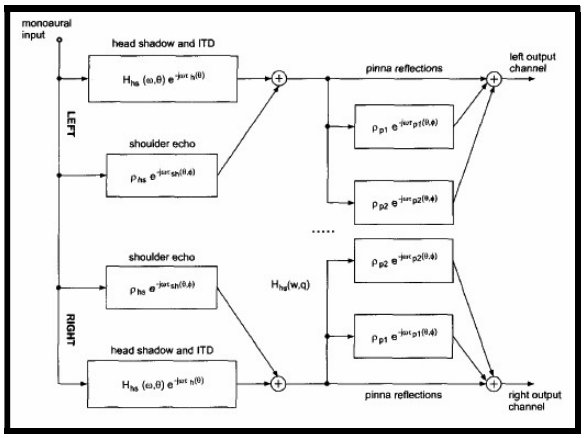

Figure 1. Structural Model of the Pinna-Head-Torso System [1]

The program first processes both channels with the *room* function. It simply adds a reflection to the sound improve the illusion of room acoustics.

Each channel is then processed by the *head* and *shoulder* functions, in parallel. Then the outputs of these functions are combined and sent through the *pinna* function. Each channel is then processed in the same way, but using a *theta* argument which is the inverse of the original.

The *head* function provides the illusion of the "acoustic shadow" on the side of the head opposite to the sound source. The filter approximates the head as a sphere using the Equations 1 and 2.

$$\alpha(\theta) = \left(1 + \frac{\alpha_{\min}}{2}\right) + \left(1 - \frac{\alpha_{\min}}{2}\right)\cos\left(\frac{\theta}{\theta_{\min}} 180°\right) \qquad (1)$$

where $\alpha_{\min} = 0.1$ and $\theta_{\min} = 150°$

$$H = \frac{(\omega_0 + \alpha F_S) + (\omega_0 + \alpha F_S) z^{-1}}{(\omega_0 + F_S) + (\omega_0 + F_S) z^{-1}} \qquad (2)$$

where $\omega_0 = \frac{c}{a}$

where c is the speed of sound and

a is the radius of the head

The interaural time delay can be approximated by using a first order all pass filter using Equation 3.

$$\tau = \begin{cases} -\frac{a}{c}\cos\theta & for \quad 0 \le |\theta| < \frac{\pi}{2} \\ \frac{a}{c}\left(|\theta| - \frac{\pi}{2}\right) & for \quad \frac{\pi}{2} \le |\theta| < \pi \end{cases} \qquad (3)$$

The *shoulder* function approximates the torso effects by calculating a single reflection with Equation 4.

$$\tau_{sh} = 1.2\frac{180° - \theta}{180°}\left(1 - 0.0004\left((\phi - 80°)\frac{180°}{180° + \theta}\right)^2\right) \qquad (4)$$

Lastly, the sound is processed by the *pinna* function. The pinna effects are approximated as a number of short echoes which are calculated using Equation 5 and Table 1.

$$\tau_{pn} = A_n \cos\left(\frac{\theta}{2}\right)\sin\left(D_n\left(90° - \phi\right)\right) + B_n \qquad (5)$$

n	ρ_{pn}	A_n[samples]	B_n[samples]	D_n
2	0.5	1	2	$\cong 1$
3	-1	5	4	$\cong 0.5$
4	0.5	5	7	$\cong 0.5$
5	-0.25	5	11	$\cong 0.5$
6	0.25	5	13	$\cong 0.5$

Table 1. Parameters for Calculating Amplitude and Time Delay of the Reflections Produced by the Pinna Model [1]

After both right and left channels have been processed for each ear, there are four signals produced. These signals are combined into two signals, for stereo sound. The signals are then normalized to prevent clipping during the *wavwrite* process. As well, if the option was selected, the sound will be played back in MATLAB.

3. PROGRAM OPERATION

The sound files are best suited for in-ear canal phones or IEMs. All M files must be unpacked into the MATLAB *work* folder. The main file is "hrtfmodel.m". Lines 18 and 19 allow the user to set the azimuth of the right and left speaker. The zero degree position is located directly in front of the user. 90 degrees is to the left of the user and -90 degrees is to the right of the user. Line 21 sets the elevation angle of the speakers. All angles are specified in degrees.

Line *24* allows the user to alter the size of the sphere for which the head is modeled. The size is represents the diameter and is specified in centimeters. Line *25* allows the user to pick from one of two pinna models, by entering "1" or "2". Option 1 is the most common pinna model, while option 2 represents a slightly altered model. In practice, it is difficult to discern between the two.

Lines *28* and *29* specify parameters of the room model. The time delay and amplitude of the room reflection can be set to the operator's preference.

Line *32* allows the user to specify the file they wish to process. In this example, a file named "davematthewsband.wav" has been selected. Note that the user should not enter the ".wav" part of the file name. The output file will automatically be saved with a suffix of "_out.wav". In this example, the output file would be saved as "davematthewsband_out.wav". Note that the program processes the entire file at once, and the whole file at once, so attempting to process a wav file which is larger than 60MB or 6 minutes, can cause the system to "run out of memory". A 6 minute WAV file can be processed in just over 30 seconds with my laptop. The amount of memory used by the MATLAB process peaks at about 950MB during the *wavwrite* process.

Line *33* allows the user to specify whether or not they wish to have MATLAB play the output. Note that the output file will be saved, regardless of this setting.

During the execution of the program, a status message is displayed in the MATLAB command window to provide the user with information regarding the status of the program.

4. MEMORY OPTIMIZATION

An initial shortcoming of the MATLAB code was that the program could only process a WAV file of about 20MB or less than 2 minutes. If a WAV file of over 20MB was processed by the program, the program would halt upon the first call of the *pinna* function. If observed in Windows Task Manager, memory consumption increased to well over 1400MB as the program came to a halt.

Initially, the apparent solution was to develop the system into a "block-processing" system. After considering the difficulty in using the *wavread* and *wavwrite* functions, a different approach was considered. By placing the *whos* function in numerous places in the code, memory usage was tracked throughout the program's execution. It was determined that several intermediate signals were consuming memory throughout the program. By using the *clear variable* function on each variable after it was no longer needed, significant memory was released.

As well, the structure of the *pinna* function was rewritten to require much less memory. After the function was rewritten, it consumed 62.5% less memory than the original function.

After all these improvements were made, it became possible to process files of up to 60MB or 6 minutes in length, which is longer than the average contemporary song. By observing Windows Task Manager, it was determined that memory consumption never increased beyond 950MB when processing a 60MB WAV file.

An interesting discovery was that during the processing of a WAV file, the memory usage peaks during the *wavwrite* functions execution. As well, if a file of over 60MB is processed, the program fully executes except for the *wavwrite* process. At this time in the program,

the only memory still allocated is two vectors containing the left and right output channels.

The block-processing implementation was not pursued further because it seems that the *wavwrite* function is the limiting factor, and the program can now process a sound file of considerable length.

5. MATLAB GUI INTERFACE

To increase the operator's ease-of-use, a simple MATLAB GUI was constructed. As seen in Figure 2, the interface allowed the user to specify all of the features of the program without editing the M file. The user could specify the positions of each speaker, the room characteristics, the HRTF parameters, and which file they desired to be processed. As well, the operator can choose to have MATLAB instantly play back the sound file. Lastly, there is a status indicator which displays the current status and progress of the program. All of the parameters were assigned defaults so that the operator could easily process any file without difficulty.

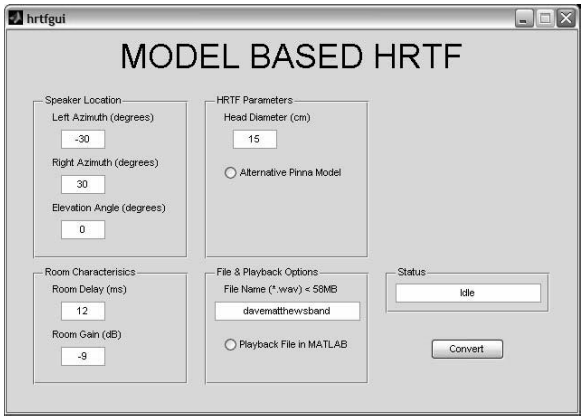

Figure 2. MATLAB GUI Interface.

The GUI was developed using the *guide* function in MATLAB. Unfortunately, the automatically generated code did not function properly. The *callbacks* were not functioning properly. A considerable amount of time was spent researching tutorials, as well as Mathworks own GUI creation guide, but the code never fully functioned properly. This GUI implementation is intended to be simple and similar to creating a GUI in Visual Studio, but it has a number of quirks which hinder the implementation. After spending several hours unsuccessfully attempting to implement the GUI, it was pursued no further.

6. REFERENCES

[1] Zolzer, U. *DAFX: Digital Audio Effects*. Wiley, 2002

[2] Brown, C.P. and Duda, R.O. *A Structural Model of Binaural Sound Synthesis*. IEEE Tran. Speech and Audio Processing, 6(5):476-488, Sept. 1998.

[3] Gardner, W.G. and Martin, K. *HRTF Measurements of a KEMAR Dummy-Head Microphone*. Technical report #280, MIT Media Lab, 1994.

[4] National Institute of Standards and Technology (NIST) Manufacturing Engineering Laboratory (MEL) Website http://www.mel.nist.gov/what.htm

[5] Huopanimei, J. and Zacharov, N. *Objective and Subjective Evaluation of Head-Related Transfer Function Filter Design*. Journal of the Audio Engineering Society (JAES), 47(4):218-239, April 1999.

[6] Kisteler, D.J. and Wightman, F.L. *A Model of Head-Related Transfer Functions Based on Principal Components Analysis and Minimum-Phase Reconstruction*, 90:97-126, 2001.

[7] Durant, E.A. and Wakefield, G.H. *Efficient Model Fitting Using a Genetic Algorithm: Pole-Zero Approximations of HRTFs*. IEEE Transactions on Speech and Audio Processing, 2002.

[8] Susnik, R., Sodnik, J., Umek, A. and Tomazic, S. *Spatial Sound Generation Using HRTF Created by the Use of Recursive Filters*. EUROCON 2003, Ljubljana, Slovenia.

[9] Rao, K.R. and Ben-Arie, J. *Optimal Head Related Transfer Functions for Hearing and Monaural Localization in Elevation: A Signal Processing Design Perspective*. IEEE Transactions on Biomedical Engineering, November 1996.

[10] Chanda, P.S. and Park, S. *A Binaural Synthesis with Multiple Sound Sources Based on Spatial Features of Head-Related Transfer Functions*. 2006 International Joint Conference on Neural Networks, July 2003.

[11] Massachusetts Institute of Technology (MIT) Media Lab http://sound.media.mit.edu/KEMAR.html