Project Report:

Eng 006 Winter Quarter 2016
Final Project
Dr. Yankelevich
Group: TRAPLAB
Members: Truc Le, Sucharita Roy, Aditya Nirgun

In the Audio Sampler Project, we used MATLAB to create a multi-function audio sampler that allows the user to upload a .wav or .mp3 file to a visualized workspace. This uploaded audio can then be edited by chopping it into smaller clips using the audio visualizer at the top. Various audio effects can also be applied using the popup menus and sliders on the left side of the GUI. Finished clips can then be saved or assigned to MIDI style buttons in the center with playback functionality. The interface also includes a metronome in the upper left corner, and a synthesizer and volume control in the bottom right.

The program begins with the user loading an audio file to the deck. We used UIGETFILE to allow the user to retrieve a locally saved .wav or .mp3 or .au file. The file is then analyzed for its data (stored as an *m x 2* signal array) and sample rate. These values are stored as global variables and then modified and called by any function editing this particular file. For example, the chopping mechanism is executed by using the zoom and pan controls on the frequency plot displayed in the deck. The portion of the track the user wishes to edit is centered within the limits within the graph. With the use of a POSTCALLBACK function, the plot returns the values of the ends of the axis which proportionately index the signal array and return the array as a specifically smaller piece than when it started. This indexing and manipulation of the signal array is used very similarly in the implementation of the audio effects Lowpass and Highpass, which filter the values in the signal array by a certain coefficient. The DJ effects Speed Up and Slow Down use the sample rate global variable to increase and decrease the rate at which the signal array is processed by the audioplayer object. These are only a few examples of the effects we've included in our code but hopefully this presents a better idea of the thought processes behind our methods. Over the course of this project, we were confronted by several bugs and complications but having three minds working together resulted in fresh perspectives and clever workarounds. Using the workspace, we were able to use unsuppressed outputs to monitor and target key problem areas in the code. The metronome was created using wait function coupled with a wave function creating a tone. The metronome counts off a default 4/4 measure with a higher tone for the downbeat. This code was inspired by a MATLAB code for finely timed tones at precise intervals that we found through our research (See Appendix B). The synthesizer uses a wave function as well and in a similar fashion although instead of the time interval set to be adjusted, it is the frequency that can be controlled by the use of a slider. The output of the wave function is passed to the SOUND( ) which is modified by some functions supported by the Signal

Processing Toolbox in MATLAB Stimulink such as SQUARE( ) and SAWTOOTH ( ). These functions change the waveform of the tone to produce a different synth sound with every selection.

Appendix A

All team members have read the task summaries contained in this report and have been given an opportunity to comment.

Since the submission of our tasklist, our original plans for dividing up the work on this project changed over the course of the few weeks we worked on it. Initially we worked on the speaker control and the playback, stop, resume, and load buttons together. Together we learned about how MATLAB processes audio signals and we figured out a basic plan for how the variables and data were to be manipulated by each function and a tentative layout for the GUI. After we were competent in these skills we each worked on our assigned portions separately and then consulted and collaborated on all the functions and features before they went into the final GUI.

Aditya worked on the synthesizer, visualizer, zoom and pan, the metronome and the GUI predraw. Sucharita worked on the lowpass/highpass filters, delay, speed up / slow down, sample reversal, and voice removal. Truc worked primarily on audio player controls, speaker controls, MIDI buttons, chopping, load and save file buttons, debugging and GUI implementation. Since he was intimately involved in creating the GUI, he had to organize and edit all of the variables and functions Sucharita and Aditya worked with in order to have them function properly in the GUI. For the video presentation, Truc did the screen recording demo and Aditya did the voice over narration and Sucharita also did the voice over narration along with the video editing. In general, we have agreed that everyone involved has contributed fairly and significantly to the project.

Appendix B

User: Pursuit; Beep Series; Stack Overflow; Jul 26, 2013 at 20:15
http://stackoverflow.com/questions/17885730/creating-fast-successive-beeps-in-matlab-psychtoolbox
Mark R. Petersen; Musical Analysis and Synthesis in Matlab; MAA's College Mathematics Journal; Vol. 35, No. 5, p.396-401, November 2004

http://amath.colorado.edu/pub/matlab/music/