Experimental Music 2 Composition

Alex Baker 1057758

Concept

My submission for this assignment comes in the form of a small application that generates music, based on some source audio and some chance procedures.
The primary inspiration for this application was John Cage's chance-based music, specifically his tape based pieces including *Williams Mix*, *Fontana Mix*, *Rozart Mix*, and *Newport Mix*. My original proposal stated my intention to construct what was essentially an automated application that took several user-defined groups of samples, arranging them in a manner similar to *Williams Mix*, and modifying certain parameters that Cage also modified such as duration, panning and amplitude, using chance procedures. A piece of software called *Williams [re]Mix[er]* has been developed by Larry Austin, an academic at the university of North Texas, that fulfils a similar purpose. I intended to expand on these ideas purely by selecting new parameters to modify, such as filter cutoffs, envelopes and parameters of arbitrary audio units.
However, as I started to work on it, I became more interested in exploring ideas less directly connected to *Williams Mix*. The first major difference is the source of the sounds to be used. Instead of having separate samples provided by the user, a single audio file (the intent is for this to be some form of musical recording but I left it open ended intentionally) is provided and events are selected from it at random. The events are grouped using an unsupervised machine learning technique, and these groups are supposed to be analogous to the categories of sound in *Williams Mix* and *Williams [re]Mix[er]*. By doing this, I achieved more abstractly defined categories than the original “county, electronic, manually produced” (and so on). The inspiration for the change in source of audio came when I was reading about Cage's *Imaginary Landscape No. 4*, a piece that features radios being turned on and off at certain times. I liked the idea of taking small segments of a larger piece of audio and using them as material to construct a composition.
After generating some music with my first draft of the program, I then looked to overall structure as my next point of expansion. Many of Cage's pieces feature a strictly defined structure known as 'Micro-macrocosmic structure'. After investigating this, I became interested in algorthimically-defined structure, and came up with an simple way of defining structure that feels elegant to me. I use a superposition of two sine waves, the frequency and phase of which are selected at random. A function like this is used for the density of each group of events, and one that is used for the density and amplitude of every group which I added because I felt that the densities of each group weren't connected enough and a “global” structure function was required.

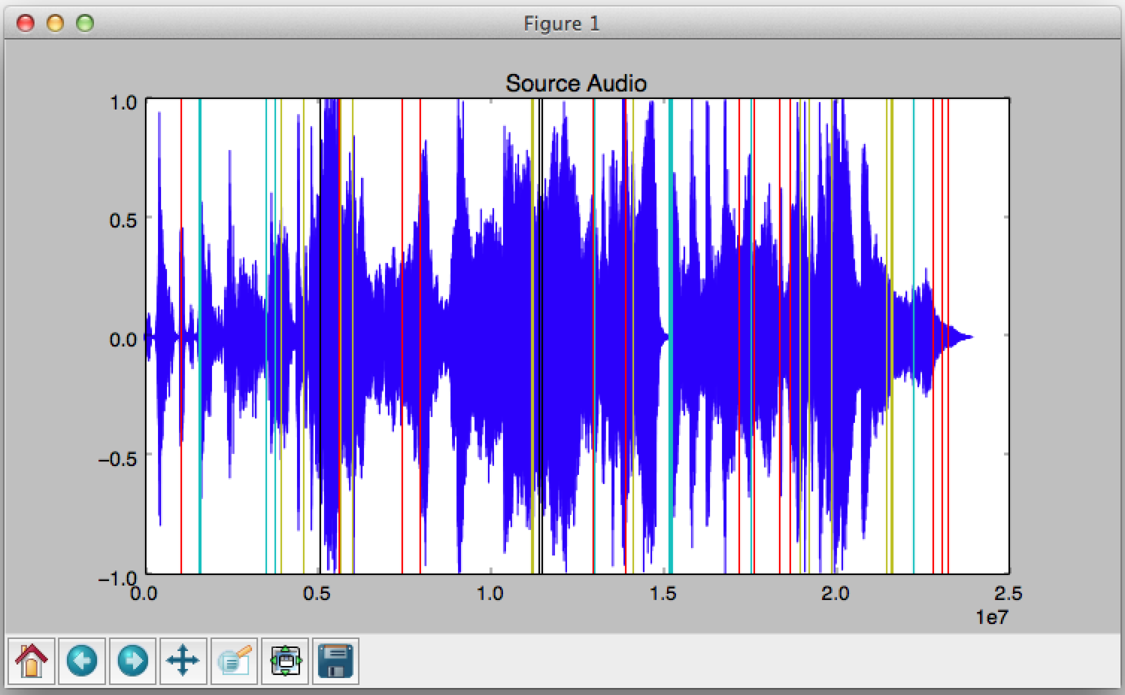
Instructions for running the application

Requirements: Python 2.7.x, NumPy, SciPy, matplotlib
First, make sure your source audio file is a 44.1khz mono uncompressed wav file, and is placed in the same directory as the script.
Open the main.py file. There is a section called “User Variables”. You should change the 'sourceaudio' variable to the name of your source file. There are a number of other parameters underneath that which are explained in comments next to them.
When you are happy with your parameters, navigate to the folder containing the script in Terminal, and run './main.py'.

If you wish to explore the functionality of the script further and understand the algorithm to a greater extent, there are a few sections towards the end that can be enabled to allow graphs to be plotted that may shed some light on how it works.

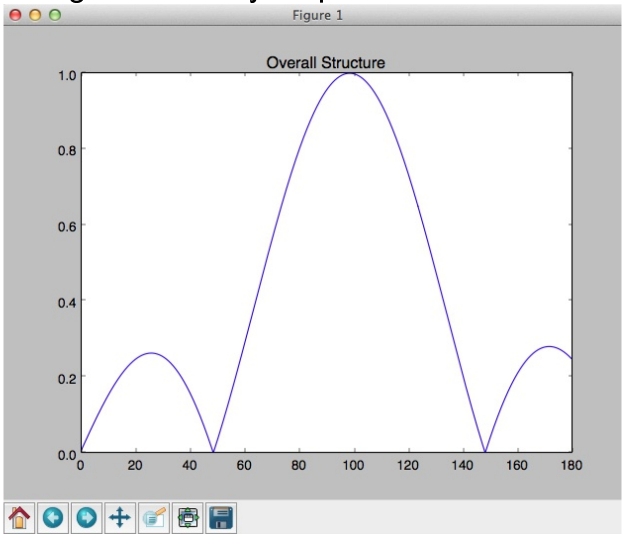
Algorithm

First, an audio file is taken as input. It is converted to floating point format so I can manipulate it in a more intuitive way.
A user-specified number of events is selected at random from the source audio. These are grouped using the K-Means clustering algorithm (MacQueen, J., 1967), based on a user-defined number of randomly selected spectral features. This usually results in audio events being grouped with events that have similar spectral properties, though the extent to which they are related differs between runs of the script.



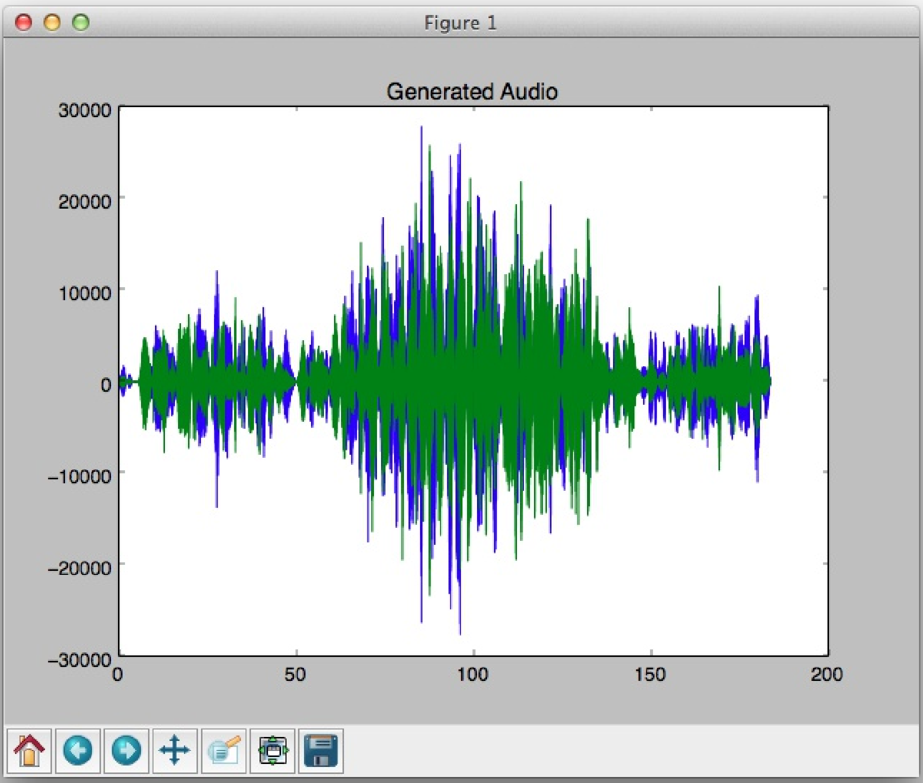
The above figure shows the source audio for the first provided audio example (Coalesce_mono.wav, the front left channel of my CC3 composition) with vertical lines showing the start positions of selected events. The colours of the lines correspond to event groups.

Next, the structure functions specified at the end of the Concept section are generated. The figure below shows the global density/amplitude function for the first example.



Next, the application loops over each event group and creates tracks for them consisting of a left and right buffer. Each track is filled with events from the corresponding group, by moving through the desired length of audio specified by the user and running a chance procedure every specified interval. The chance is equal to both the track and global density structure added together, all multiplied by the specified track density.

After each track has been filled, the audio is mixed down to stereo and exported as a wav file.



From this figure, it is clear to see that the global structure function strongly impacts the form of the piece.

Bibliography
Cage, J. (1951). *Music of Changes*.
Cage, J. (1953). *Williams Mix*.
Cage, J. (1951). *Imaginary Landscape No. 4.*
Austin, L., Thompson, M. (2001). Williams [re]Mix[er]: An Interactive I Ching Composing Program. *International Computer Music Conference*.
John Cage Work Detail: Music of Changes. Retrieved from http://johncage.org/pp/John-Cage-Work-Detail.cfm?work_ID=134
John Cage Work Detail: Williams Mix. Retrieved from http://johncage.org/pp/John-Cage-Work-Detail.cfm?work_ID=246
MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. Retrieved from
http://projecteuclid.org/download/pdf_1/euclid.bsmsp/1200512992