Anonymous MIT student 2009-12-04

Prof. Chirstopher Ariza 21M.380

Sonic System Project

Approximating Acoustics with Convolution

The purpose of this project is to design and implement a software Sonic System that could effectively model the acoustics of a room. This is carried out by using several special properties of sound and acoustic systems, which facilitate the intense computation associated with such tasks. The project was implemented in Java and uses the Java Sound back-end in order to access audio streams such as speakers, microphones, and software mixers.

The motivations behind this project were numerous. The major aesthetic motivator was the experimental piece “I'm Sitting in a Room,” by Alvin Lucier. This piece was generated as follows: Lucier spoke into a microphone and recorded his voice. He then positioned the microphone in a certain way and recorded the playback of his voice, which also captured his acoustics. Over time, these acoustics would cause several aspects of his voice to fade and would amplify others. Eventually, the recording converges to a smooth, rich tone, which I find very aesthetically appealing. Since then, I have been thinking of ways to simulate that process computationally, which could incredibly speed up the process by which Lucier arrived at his new sound.

The principles by which this is carried out through software are all founded on the breathtaking linearity of sound systems such as rooms. Because sound travels at a near-constant velocity, independent of frequency, to good approximation, any acoustic system without much

noise can be approximated as a Linear, Time-Invariant System (LTI).

A remarkable property of these systems is that in emitting a unit sample (a sample whose amplitude-time product is 1) and observing the response to such an input, one can extend these observations and predict a system's outputs given other inputs. The method by which this is calculated is called convolution, which is defined as follows: the sum from negative infinity to infinity, of one function sampled at the iterative index, times another function at the absolute index minus the iterative index.

The fact that one can simulate the acoustics of a room and why convolution holds is due to a property of linear systems called superposition. Superposition can be defined as follows: Given one input and its response, and another input and its response, the response to both these inputs must be the same as the simple addition of the single, observed responses. Similarly, given an input and its response, the response to that scaled input must be the observed response scaled by that same factor. Thus, any discrete signal can be modeled as a superposition of unit pulses.

The framework that was chosen for this project is Java. Java is an object-oriented programming language that lends itself well to dealing with quantized, discrete entities. Aside from its robust object-orientation, Java was chosen for its minimalist, visual Sampled Sound framework. The framework operates as follows: Each “entity” that deals with sound implements the interface Line. Each Line represents a flow of data, either out or into the computer. There are mixer objects that serve as encapsulators and proprietors of these Lines. These mixer objects can be obtained from the AudioSystem object and are direct interfaces to the different mixers present on a machine. Due to the high level of abstraction, the framework simplifies the task of choosing a mixer; all that is required is to specify the characteristics of the Mixer or the Lines. All of these classes are encapsulated by the AudioSystem object, which serves as a hub from which Lines and

Mixers may be accessed. All of the remaining classes are simply instantiations of the mentioned superclasses and are a bit more specialized: some Lines are dedicated to sound capturing and others to sound playback.

When one accesses a Line, if it is available, one “opens” the line and starts streaming data to or from the program. The Java Sound framework only supports Pulse-Code Modulation encoding schemes, which are schemes that encode the wave as a succession of samples, each of which contains an amplitude of the sound wave. The stream is composed of an array of bytes which is renewed in intervals as the sound object is processed.

The convolver is implemented as follows: The Convolver class handles user interaction. Filenames and parameters are passed through the terminal and are interpreted by the program. Computation is handled by the ConvolverMixer, which encompasses two input InputAudioStreams, one for the unit impulse reponse, and another for the raw audio input, and one output stream. The ConvolverMixer handles multiple channels gracefully, but places one major restriction on the input streams: both input lines must share the same sampling frequency, bit depth, and number of channels. The program is built hierarchically, dividing major tasks into simpler subtasks. The convolved samples are output to the specified File that was included in the Object declaration. The output data file has the same properties of the input streams.

The output of the system merely convolves the audio (at this time). The files that I have used so far are freely available on the Internet. The output sounds convincing but could be tweaked to take into account differences in maximum amplitudes, among other factors. A way in which the program could be improved substantially would be by enabling the recording of voice in order for that stream to be convolved. Another way in which this Sonic System could be improved would be by creating a method through which a unit impulse could be generated and

another through which a room's impulse response could be captured. This project has a lot of potential for experimentation with convolution and reverberation. Eventually, I hope to be able to iteratively convolute a sound input and to arrive at a sound similar to Lucier's.

References:

- "Overview of the sampled package". Obtained at: http://java.sun.com/docs/books/tutorial/sound/sampled-overview.html
- 6.02 Lecture Notes. Obtained at: http://web.mit.edu/6.02/www/
- currentsemester/handouts.html#lectures
- French, A. P. (1971). "Vibrations & Waves: The M.I.T. Introductory Physics Series". Published by W. W. Norton and Company in New York/London.

Note:

The code for this project can be accessed online at http://18.244.6.67/Public/21M.380/ssp/
This code is a work in progress. The final iteration will most likely have more features.

21M.380 Music and Technology (Contemporary History and Aesthetics)

Fall 2009