Abstract

The user interface has been one of the greatest hindrances to computer music composers who wish to perform detailed editing and manipulation of digital sound. A sound editor and processor has been developed, written in C and using the X Window System graphics interface, for use in the creation of computer music and the teaching thereof. In its final form, it will incorporate all of the sound processing algorithms currently associated with the Cmix software system. The graphics interface is written in C, and the program will run on any UNIX system supporting X, and possibly other operating systems which support that interface. In its current state, the program features copying, mixing, splicing, looping, transposing, and enveloping commands, all controlled via a mouse plus a small amount of typing at a dialog box. The program will be demonstrated on a NeXT computer, and its capabilities will be explored. Mixview has been developed at Columbia University, New York City and at the Center for Experiential Music and Intermedia, University of North Texas, in Denton, Texas.

Introduction

As a composer using computer-generated sound, I found that the available software for the manipulation of such sound, such as Cmix, Csound, and Csound, were not designed for the kind of detailed sound-sculpting that I had found so attractive in my early "classical" training in an analog tape studio. For instance, I could not change a single particular cycle of a waveform and view it in another file involved a tedious search of the soundwave with a histogram routine at increasingly higher resolutions until the place was located, unrolling down to sinew, then creating a synclone to look at the data as to another routine in order to accomplish the transfer, plus a complicated operation to shift the original data over to offset the "splice". What was needed was an integrated software package in which all the essential editing and processing algorithms were available as a suite of tools, with a graphics interface allowing visual feedback of the results. Such software is commercially available for the Apple Macintosh series, running under its native operating system, but is not portable to any other architecture. The use of the sound files is also limited by the available memory of the Macintosh system. The program mixview C is an example.

Overview

Due to my familiarity with Cmix (developed at Princeton University by Paul Lansky) the underlining design of my program (beneath the graphics interface) closely parallels that system, with a couple of important differences. First, all of the processing routines function as regular C subroutine rather than being compiled in individually and "compiled" from a command line. Second, the majority of the activity takes place in virtual memory, thereby avoiding the slowdowns of disk use. Third, all data necessary for the completion of any processing command is either sound graphically (mouse click, etc) or via information typed into and in response to the many "diagonal boxes". The program runs in a fashion similar to other X applications: the command is typed at the keyboard any time after the X server software has been started. All other user input is done through the program's various data-input system.

The Graphics Interface

A brief overview of the X Window System: X consists of a large library of low-level graphics functions, plus routines for creating and manipulating windows and other graphics "objects", plus a number of higher-level "toolkits", developed under the creators of X or others working in cooperation with them, which govern the creation and manipulation of more complex "objects" (such as buttons and scrolling text windows). Mixview uses the Lamont X Toolkit (LXT), developed by Roger Davis, Sam Howard, and Bill Mosle at the Lamont-Doherty Geological Observatory in Palisades, NY. I have worked in collaboration with Mr. Davis on LXT, which is still in beta-test form.

ICMC GLASGOW 1990 PROCEEDINGS

262
LXT has allowed me to create a set of windows, one for each control of the visual display of the soundfile, and any number of windows for the actual plotting of the sound waveform. As with most look-alikes, LXT uses buttons, labels, sliders, text-entry boxes, and other objects as the interface between the user and the underlying routines. In some ways, mini-view resembles a word processor. The sound clip can be scrolled back and forth in the window, and accouters are selected by highlighting them, at which time they can be copied, cut, or erased, and then inserted elsewhere. The editing possibilities are numerous, though, due to the nature of sound processing.

The Program

In its current form, mini-view is written in C. Preparations are underway for a re-write in C++ or Objective C, and the code in its present form has a large number of object-oriented structural features. It is modular in nature, allowing for easy expansion as new routines are added. It is designed to take advantage of the speed and memory of a mainframe system, but should function adequately on any system with at least 2 Megabytes of RAM. It has been successfully compiled and run on a Sun 3/280, a Sun 4, a Vax 11/780, and the NEXT workstation.

Current Features

Editing commands: copy, remove (to cut buffer), splice ou. erase, nsp (splice out without save), mix, replace, splice in, consolidate.

Allowing commands: phrase (apply amplitude envelope), reverb, insert space, transpose, stretch (frequency-dependent duration change). (See figures 1A-E). Filtering commands: elliptical, resonant, low pass, comb. Global commands: change sampling rate, change file length, mcaic, add text comment.

The sound waveforms may be viewed at any horizontal or vertical resolution (see figures 2A-E). The start and end times of the currently displayed portion are displayed, as well as the times for the currently selected edit region. The current insert point time is displayed in the waveform window, with an optional display of the amplitude for the file at that point. Portions of a file may be copied into a buffer for transfer to another file. The version running on the Sun 3/280 at Columbia University has the capability of doing direct D/A conversion from memory through a driver designed by Brad Garden, using hardware designed by Micro Technologies Unlimited. Mini-view has been ported to the NEXT workstation, running on the X server which is available for the NEXT, and it is in this form that it will be demonstrated. The soundfile playing routines have been adapted for use with the D/A converters on the NEXT, so that one can have immediate audio feedback for all sound modifications. Both EOB and NEXT soundfile formats can be edited and played, and both short integer and floating point files can be edited.

The Future

Future additions to mini-view include additional processing algorithms, FFT and envelope analysis, mono-to-octave processors such as panning and room simulation, and a graphic interface for LPC analysis and Phase Vocoder analysis. Also, a major structural revision is underway to allow a theoretically unlimited number of windows and windows in windows to be open simultaneously for the purpose of user-file transfers and editing. As additional workstations manufacturer develop software and hardware for digital-to-analog conversion, code for these machines will be added.

Conclusions

Given the rise in popularity and availability of graphics workstations with main frame level memory and speed, it seems natural to develop systems that take advantage of the faster and more intuitive methods of control made possible by these workstations. This is especially important in education, where the complexities of an ASCII-oriented interface may prove daunting to novices just beginning to work with computers. I wish to stress that mini-view is not a commercial product. It is made up of components which are in the public domain. It has been designed for use in academic educational environments and is not for sale.

References

