Listen: A Data Sonification Toolkit

Catherine M. Wilson and Suresh K. Lodha
Computer Science, University of California—Santa Cruz

Abstract: Data sonification is the representation of data using sound. Although data sonification has been a subject of research for the last fifteen years, much remains to be learned about how data should be mapped to sound, what constitutes a good mapping, what kinds of data can be meaningfully sonified, what attributes of data can be meaningfully sonified, and how sonification should be combined with visualization. On the basis of the work done so far, we expect that sonification will soon be used routinely in scientific visualization. In order to encourage the incorporation of sonification into the research environment, a flexible, adaptable, extensible, portable, and interactive toolkit is needed.

A primary goal in the design of *Listen* was to provide such a toolkit for use in exploring data of any type. *Listen*, is an object-oriented, modular system. It provides incremental functionality; researchers can begin using sonification with a minimum investment of time and resources. Once sonification has proven its value, researchers can implement more sophisticated capabilities. *Listen* can be easily adapted by the user to a particular environment and extended when additional functionality is required. A key feature of *Listen* is that it can be easily incorporated into an existing visualization system. Some key words in this essay are sonification, visualization, interactive, portable, and MIDI.

Introduction

Sonification is the mapping of data values to properties of sound such as pitch, volume, duration, timbre and location. It is the aural equivalent of scientific visualization. It is analogous to mapping data values to color or other properties of light in visualization. Advantages of data sonification, particularly in conjunction with visualization, have been recognized in the last few years (Scaletti, 1994). In spite of great deal of work on sonification, the need for a general-purpose data sonification toolkit, that can conveniently allow experimentation with scientific data, remains. Such a system needs to be interactive, flexible and portable. The primary goal in the creation of *Listen* was to create such a tool.

Background

Sound Hardware

Many computers, including the SGI workstation, the Sun SparcStation, the Macintosh, and the PC, contain internal audio devices. Since the advent of multimedia, the ability to add a MIDI device has also been implemented on these platforms. The advantages of MIDI devices are many. They are abundant and inexpensive. Many sound parameters such as pitch, volume, timbre, and duration can be controlled by standard MIDI messages. Modern synthesizers allow many notes to be played simultaneously and most allow control of stereo balance. If more control of the sound envelope is needed, proprietary messages can be used if the device supports them.

The MIDI device used in development of *Listen* was the Yamaha PSR-320, an inexpensive synthesizer that supports General MIDI. It was connected to the computer by an Opcode MIDI translator.

Sound Libraries

The availability of sound libraries greatly simplifies the task of implementing a sonification system. SGI workstations have an audio library to control the internal audio device and a MIDI library to control an external MIDI device. *Listen*is implemented on the SGI platform and uses both the audio and MIDI libraries to control the sound hardware. The use of these libraries makes it easy to extend the functionality of the program by increasing the understandability of the interface to the sound hardware.

Sonification Systems

We refer the reader to (Scaletti, 1994) for a survey of applications of sonification to particular problem domains. The few general sonification systems that have been created have had limited success for a number of reasons. The system designed by P. Astheimer, as an extension of the apE visualization system, limits the user to implementing a visualization on that system (Astheimer, 1993). The EXVIS system, which is limited to a two-dimensional iconographic visual display, is not sufficiently general or flexible (Grinstein and Pickett, 1989). The sonification system by Minghim and Forrest for visualizing volumetric data and surface properties, implemented on Macintosh Quadra, is not portable (Minghim and Forrest, 1995). Carla Scaletti's Kyma system requires exotic sound hardware, the Capybara (Scaletti, 1994).

Although these devices provide great flexibility in sound synthesis through the use of digital signal processors (DSPs), they are not commonly available. Moreover, implementing a sonification system with this type of device involves extensive knowledge of sound synthesis algorithms and familiarity with synthesis hardware.

The Porsonify system designed by Madhyastha and Reed was implemented on Sun SparcStation without any audio or MIDI library, which made the task much more difficult and resulted in a system that was complicated to use and difficult to learn (Madhyastha and Reed, 1995). To communicate with the sound devices, daemons and device drivers were written to handle the interaction with the hardware. Porting the program to another platform without sound libraries would require new daemons and drivers to be written -- a daunting task. It is unclear whether the high level functionality of Porsonify is sufficiently divorced from the hardware interactions to allow easier porting to a platform such as SGI that has sound library support. In addition, possibly because of the constraints of the platform, interaction with Porsonify is difficult. The user must work with multiple files, such as widget control files, sound hardware configuration files and sound control files to create a sonification. Because of this complexity, Porsonify has a steep learning curve.

Listen

The primary objective in the creation of *Listen* was to provide an inexpensive, general purpose, flexible, interactive, adaptable and portable sonification tool. Here we present a brief summary of *Listen*. Further details are available in (Wilson, 1996).

Listen Toolkit

To provide incremental functionality, there are four basic programs that comprise the toolkit. *Listen 1* accepts command-line arguments and uses the internal audio chip. *Listen 2* uses the internal audio chip and has a graphical user interface that permits more complex mappings than *Listen 1*. *Listen 3* uses a MIDI device and also has a graphical user interface. *Listen 4* is a module version of *Listen 3* and is designed to be easily incorporated into an existing visualization program:

Listen 1 and Listen 2: *Listen 1* and *Listen 2* allow the user to get started using sonification minimum difficulty. No additional hardware is required, since both use the internal audio chip. *Listen 2* uses a graphical interface provided by XForms. The user can interactively specify several different sound parameters using menus, sliders and buttons. If XForms is not installed on the system, the command-line program can be used. These basic programs allow a potential user to get started quickly. If sonification is not found to be useful, the investment of time and resources has been minimal.
Listen 3: If sonification produces good results, the addition of a MIDI device can be justified. *Listen 3* uses a MIDI device and also provides more complex mapping possibilities. In *Listen 1* and *Listen 2*, all mappings are linear. In *Listen 3* a linear mapping is the default, but it is possible to select a transfer function that is non-linear. This feature allows the user to implement high-pass, low-pass or boolean filters. Using a utility package, the user can interactively define customized non-linear transfer functions. These can then be loaded into the *Listen* program. It is also possible to define a transfer function in which each parameter has its own unique function.
Listen 4: Once sonification has proven its value and users have gained experience with it, the addition of the module, *Listen 4*, to existing visualization programs can be done with the assurance that the investment of time will be rewarded. Integrating the module into an existing program is quite simple and straightforward and is described in detail in (Wilson, 1996).

As a unit, these four programs provide a structured starting point for experimenting with sonification. The expectation is that, using the basic ideas already implemented and with the experience already gained, the user will create more complex, more interesting, and more useful sonification tools as more is learned about sonification in a particular environment.

The Modules

Listen is an object-oriented system written in C++ for the SGI platform. *Listen* has five modules: an interface module, a control module, a data manager module, a sound mapping module, and a sound device module. The modularity of the design is the single most important feature in achieving the project objectives. Flexibility, adaptability, and extensibility would be impossible to achieve without this modular, object-oriented design.

The interface module handles user input and provides feedback to the user about the state of the program. The control module is the intermediary between the interface and the other three basic modules. It creates a clear separation between the basic modules and the interface. For *Listen 1*, *Listen 2* and *Listen 3*, the data manager module reads the data from a file to learn essential information about the data fields as well as minimum and maximum values of each data field, and feeds this information to the mapping module. When the module, *Listen 4*, is integrated into a visualization program, the parent program manages the data. The data manager can then be relieved of the responsibility for reading data files, but must still keep track of the minimum and maximum values of each data stream that is to be sonified. The sound mapping module keeps track of how the data is to be mapped to the sound parameters. It also keeps track of information about the sound parameters themselves, including their maximum, minimum, and default values. The sound parameters implemented are: pitch, duration, volume, and location. In the MIDI implementation, a timbre can be assigned to each data stream. The sound device module is responsible for initializing the sound equipment and causing it to create a particular sound.

Attributes of Listen

Listen is general. It allows the user to sonify any kind of data and supports both a dataflow model and a probe model.

Listen is flexible. Any data stream can be mapped to any sound parameter using any timbre. Subsets of each data stream can be easily selected for sonification. *Listen 3*, with MIDI support, can sonify any reasonable number of data streams in any combination.

Listen is intuitive. The graphical interface presents a display of all of *Listen's* sonification capabilities simultaneously and is laid out in such a way that setting up a sonification is self-explanatory.

Listen is interactive. Because all mapping controls are always present on the graphical interface, setting up a sonification involves little more than selecting a few menu items and manipulating some sliders. Many aspects of the sonification, such as sound parameter ranges and defaults, can be changed even while a sonification is playing. If a sonification is uninteresting, it can be stopped at any time.

Listen is adaptable. Because of the modular structure of the program, it is very understandable, which makes it robust and easy to modify.

Listen is extensible. New functionality can be added easily by extending *Listen's* classes as needed.

Listen is portable. Only the sound device module and possibly the user interface need to be changed in order to port *Listen* to another platform. *Listen* can be implemented on easily available hardware. MIDI devices are inexpensive and their use in multimedia ensures that many platforms will support them.

Applications

Listen was easily incorporated into a system for visualizing geometric uncertainty of surface interpolants (Lodha et al., 1995) and a system for visualizing algorithmic uncertainty in fluid flow (Lodha et al., 1996a). Sonification was found to be very useful and the results are presented in (Lodha et al., 1996b).

Conclusions

Although there are a number of visualization systems available, such as AVS, IBM Data Explorer, and SGI Explorer, many scientists prefer to create their own customized visualization systems tailored to the data they need to display. A sonification toolkit that can be easily adapted to each specialized visualization system can provide the ease of use and flexibility needed for sonification to be accepted and used. *Listen* provides this adaptive capability.

One way to determine how to create meaningful sonifications is to experiment with creating them using real data. Often, effective sonifications are data driven, with the nature of the data itself suggesting new and more effective ways of using different sound parameters and different ways of mapping the data to them. *Listen* provides an interactive, flexible and portable environment to encourage this essential experimentation.

References

Astheimer, P. (1993). Sonification tools to supplement dataflow visualization. In Patrizia Palamidese, editor, *Scientific Visualization: Advanced Software Techniques.* Ellis Horwood. pp. 15-36.

Grinstein, G.G. and Pickett, R.M. (1989). Exvis - an exploratory visualization environment. *Proceedings of Graphics Interface '89.*

Gregory Kramer, (ed), (1994). *Auditory Display: Sonification, Audification, and Auditory Interfaces.* Addison-Wesley.

Lodha, Suresh K., Pang, Alex T., Sheehan, Bob, and Wittenbrink, Craig M. (1996a). Uflow: Visualizing uncertainty in fluid flow. *Proceedings of IEEE Visualization '96.*

Lodha, Suresh K., Sheehan, Bob, Pang, Alex T., and Wittenbrink, Craig M.(1995). Visualizing geometric uncertainty of surface interpolants. *Proceedings of Graphics Interface '96*, pp. 238-245.

Lodha, Suresh K., Wilson, Catherine M., and Sheehan, Bob (1996b). LISTEN: Sounding uncertainty visualization. *Proceedings of IEEE Visualization '96*.

Madhyastha, Tara and Reed, Daniel (1995). Data sonification: Do you hear what I see? *IEEE Software*, 12(2):85-90.

Minghim, R. and Forrest, A.R. (1995). An illustrated analysis of sonification for scientific visualization. *Proceedings of IEEE Visualization '95*, pp. 110--117.

Scaletti, Carla (1994). Sound synthesis algorithms for auditory data representations. In Gregory Kramer, editor, *Auditory Display: Sonification, Audification, and Auditory Interfaces*, pages 223--251. Addison-Wesley.

Wilson, Catherine M. (1996). Listen: A data sonification toolkit. *M.S. Thesis*, Department of Computer Science, University of California, Santa Cruz.

Author Information

Catherine M. Wilson and Suresh K. Lodha
Computer Science
University of California, Santa Cruz
Santa Cruz, CA 95064 USA