A FLEXIBLE FRAMEWORK FOR REAL-TIME SONIFICATION WITH SONART

Woon Seung Yeo, Jonathan Berger, R. Scott Wilson

The Center for Computer Research in Music and Acoustics
Stanford University, Stanford, CA 94305
`brg@ccrma.stanford.edu`

ABSTRACT

We describe significant developments towards a real-time implementation of *SonArt*, the parameter mapping framework first presented in [1] [1]. Enhancements include the incorporation of *Open Sound Control* (OSC) [2] which facilitates network communications, direct access to a variety of real-time synthesis software platforms, and distributed synthesis. The original goals for open-source, platform independence, and modularity are further discussed with an example implementation using Java and OSC ([4]).

1. INTRODUCTION

In this paper we describe significant developments of the parameter mapping framework described in (Ben-Tal, Berger et al, 2002) with particular focus on the following issues:

- network transmission of data
- real time sound generation
- distributed synthesis
- cross platform, modular design

These issues are critical for our goal of implementing a flexible, real-time sonification framework. We have incorporated Open Sound Control (OSC) [2] (updated at [3]) protocol for networked communication among hardware and software. OSC allows for modularity between the data server, parameter mapper and synthesis platform. It also facilitates distributed synthesis adding considerable flexibility to the framework. The need for this flexibility is evident in our experiments with hyperspectral image data of colon cells. This data, collected incooperation with our collaborators at Yale University, consists of datacubes of normal and abnormal colonic tissue. Each data cube consists of 491 x 652 pixels, each pixel containing 128 dimensions. Our sonification experiments included sonifying all 128 dimensions as well as a variety of data-reduced sets attained through principal component analysis, local discriminant bases and nearest-neighbor classification. A visual RGB representation of the data is used referentially, allowing for sonification of specific pixels or areas of pixels by navigating around the visual image with a computer mouse. This navigation method necessitated interfacing with a robust and efficient real-time synthesis system. The high dimensionality of the data posed both conceptual and optimization challenges. We generalized the problems encountered and searched for design solutions to these. The problems include:

- logistics and locale. Data should be available to multiple clients simultaneously and immediately whether that data originates from a single source or from multiple sources, and whether the data is being continuously updated (for example, taking live data output from a sensor or imager, or addressing archived data.
- transmission. A networked distributed sound synthesis system is a major advantage to a sonification design providing potentially limitless computing power and flexibility. Minimizing latency is of critical importance to this end.
- communications. Our ideal sonification design framework calls for platform independent solutions and flexibility of synthesis design as well as the ability to combine multiple solutions for a single sonification task.

The logistical, transmission and communications issues cited prompted us to incorporate OSC as the communications protocol between the data server, the parameter engine and the synthesis systems.

2. OPEN SOUND CONTROL

OSC is a protocol for communication among various sound control and/or synthesis engines (i.e., computers, synthesizers, and other multimedia devices), offering flexibility and power. Using OSC, it is possible to send/receive data between applications, both locally and through network. Moreover, it's implemented in almost every widely-used sound synthesis softwares, including Csound, Max/MSP, Pd, and Supercollider, which allows them to sonify our dataset. At the core of OSC is a specification for the format of messages. Each OSC message contains a URL / path style target address and a variable number of parameter arguments. The message format is independent of transport layer, but the vast majority of existing implementations use UDP packets over IP networks for transport. A Java implementation of OSC is described in **??**.

The implementor of a particular sonification paradigm is thus able to design an optimal set of messages they will respond to, covering the full range of the parameter space within that paradigm. They would then publish that set of messages along with the acceptable ranges for each argument in a commonly agreed upon format. Stored with the message catalogue is implementation specific information needed by the central parameter mapper to instantiate and/or initiate communication with the paradigm in question. The message catalogue and implementation information are used by the parameter mapper to present the user with a graphical interface for patching together connections between data sources and synthesis targets in a consistent manner.

To date, we have developed a number of programs that incorporate OSC for sonification purposes. We desribe two of these efforts.

[1]This work is supported by DARPA award F41624-03-1-7000

3. THE DATA VIEWER: AN EXAMPLE APPLICATION

Based upon the needs to use a visual representation of data as a means of navigation for sonification exploration we designed a Mac OS X application with:

- a user interface that allows the user to click on a data point, get the corresponding (MATLAB) data of that point, and send them out as OSC locally or through network
- the ability to import .mat data files and associated image files.
- the ability to provide an IP address and port number for OSC communications.
- the ability to support multi-IP addressing.

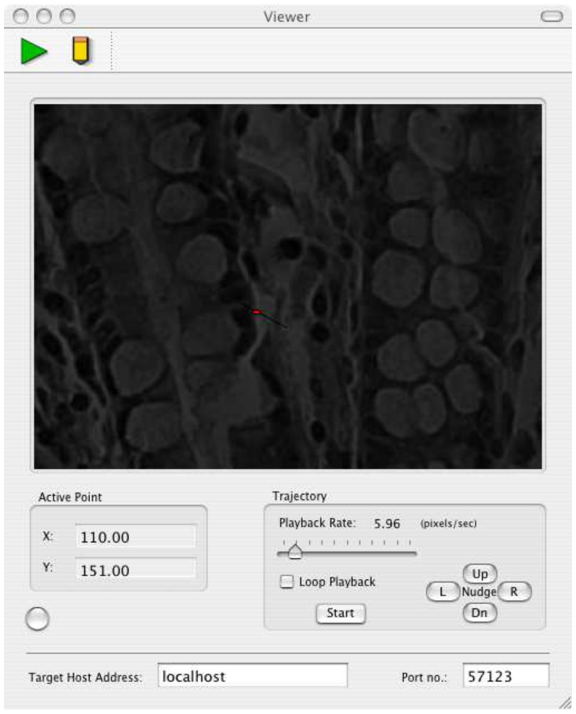

Figure 1: *OS-X Data Viewer.*

Figure 1 is a screenshot of this program.
Viewer supports two modes of data transfer:

- trajectory. the user can set up a linear path within the screen image by dragging the mouse. A playback panel allows for the transfer of the data vector along the points on the path through the OSC connection. The location of path can be linearly calibrated, and data playback rate can be adjusted.
- mouse. The data vector of individual pixels corresponding to the user selected mouse position are sent through the OSC connection.

Subsequently, a Java applet version of the data viewer was written and is currently in use for managing sonification using standard web browsers (figure 2). This provides the flexibility of platform independence. Although currently the viewer supports only local transmission, a future network version will provide a powerful framework for serving and processing sonification from a centralized data source.

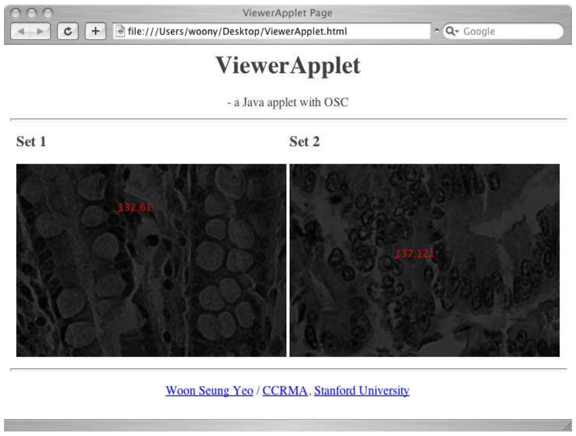
ViewerApplet Page
file:///Users/woony/Desktop/ViewerApplet.html
ViewerApplet
- a Java applet with OSC
Set 1
Set 2
132,61
137,121
Woon Seung Yeo / CCRMA, Stanford University

Figure 2: *Java Data Viewer.*

Basically, this has almost same function as the OS X program mentioned above. It shows an image when loaded by browsers, and as users click and/or drag on the image, the data vector corresponding to the (x,y) coordinate values of mouse pointer will be sent out through OSC connection. As with many other applets, parameters for the dataset, image display, and OSC connection parameters can be configured externally by using HTML tags. This is a useful feature for designing web pages as interfaces for data sonification. The following is an example of HTML source used to display the applet.

```
---------------------------
<APPLET archive="DemoMouse.jar"
    code="DemoMouse" width="512"
    height="512" vspace="10">
<PARAM NAME="IMAGEFILENAME"
    value="Coeff_B01.small.jpg">
<PARAM NAME="DATAFILENAME"
    value="Coeff_B01.mat">
<PARAM NAME="PORTNUMBER"
    value="57123">
<PARAM NAME="TYPETAG"
    value="/webSoni">
</APPLET>}
--------------------------- }
```

Since this applet works on every machine with Java-enabled web browsers, it is virtually platform independent. This is a critical factor for distributed sound synthesis systems over network where machines of various platform coexist.

Although currently this applet only supports local transmission, a future network version will provide a powerful framework for sonification server of centralized data.

4. CONCLUSION

Our programs produced satisfactory results in terms of efficient data transmission performance when tested on a local network. We could generate sounds with Max/MSP, and PD using 128 dimensional colon cell data transmitted from a different machine over the network. However, it was still one-to-one unidirectional communication within a well-controlled environment. Future research will focus on achieving robust performance for more general cases of distributed environment. Sonifying large-scale, high dimensional data imposes significant issues in terms of storing and managing mapping information. We continue to develop methods to meet these challenges.

5. REFERENCES

[1] O. Ben-Tal, J. Berger, B. Cook, M. Daniels, G. Scavone, and P. Cook *SONART : The Sonification Application Research Toolkit*, Proceedings of the 2002 International Conference on Auditory Display, Kyoto, Japan.

[2] M. Wright, A. Freed, A. Lee, T. Madden, and A. Momeni, *Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC*, Proceedings of the 2001 International Computer Music Conference, Havana, Cuba.

[3] *Open Sound Control*,http://cnmat.cnmat.berkeley.edu/OpenSoundControl/

[4] C. Ramakrishnan, *Open Sound Control for Java*, http://www.mat.ucsb.edu/ c.ramakr/illposed/javaosc.html