Sonification, electronic physical instruments, aesthetic experiments and their role in my work

Jonáš Gruska
Bachelor's Thesis
Institute of Sonology

October 5, 2012

Abstract

Acknowledgements

Contents

Chapter 1

Introduction

Chapter 2

Sonification and audification experiments

2.1 Thoughts on current state of sonification

To me, topic of sonification is intriguing for few reasons. First of them could be, that it is a topic benefiting from interest from various groups of scientists and creatives - from biologists, through mathematicians, to musicians and sound artists. Another is a belief, that this topic still has a lot of cloaked potential. I will try to discuss this clash between scientific and artistic approach, as well as fields of opportunity raising from this duality.

I would like to define two modes of sonification, although generally there is only one definition of the word. Main understanding of the term is usually analytical, *representative*, where it is the "use of non-speech audio to convey information" (Kramer et al., 1999), or as Nick Collins puts it, where one tries to provide "aural alternatives to pie charts, line graphs and spreadsheets" (Collins, 2006). My addition to the definition is *an artistic mode*, where the main point of the process is not to make certain data accessible for aural analysis, but to use those data as a essence of specific aesthetics. These modes can overlap, but usually the distinction is clear, and analytical mode prevails in quantity.

2.2 Wireless network sonification

As a part of my experimentation with untraditional sources for musical or generally sonic data I decided to extend my line of work on the layer of wireless networks. First I would like to define some basic terms of the technical side of

the issue and then continue by describing my point of view on even more important social and artistic side.

One of the core subjects related to this field are *packet analyzers* or *sniffers* – tools for silent interception of network traffic. In technical terms, it is software (or hardware) which passively receives all data link layer frames passing through the device's network adapter, not just those directed towards recipient. They are generally used by hackers, while also having legitimate use by system and network administrators or security experts. Sniffing is possible due to the physical characteristic of network. Either with LAN (Local Area Network) or unencrypted WLAN (Wireless LAN), all data which are passed through the system are accessible to everyone. It is than a job of WLAN or LAN adapters (network cards) on the site of client to separate its part of the data using filtering based on MAC (Media Access Control) address. MAC address is an unique identifier of every network adapter. Except simple network MAC filtering, it can be for example also used for assigning IP address by DHCP (Dynamic Host Configuration Protocol).

By enabling functionality of network device called *promiscuous mode* one is able to avoid this filtering and obtain all present network data. This is possible as well as for wired as for wireless adapters, thus for wired or wireless networks. Of course there are certain limits to this, for example when a *network switch* is placed in the network, it is not possible to receive everything. Network switch, different from *repeater* or *hub* (device only physically 'multiplying' the cable), is more advanced device and acts as a MAC filter by itself. (Pallavi & Hemlata, 2012)

A classic example of a sniffer is **tcpdump**. Tcpdump was developed at University of California in 1987 by Van Jacobson, Craig Leres and Steven McCanne and its main use is traffic analysis. Unix Manual entry describes it as "dump traffic on a network", but that is very reducing. Over the years since its first release it has become a rather complex and powerful tool. One of the features of tcpdump is that it can work easily with both LAN or WLAN packets and therefore provides us with various data sources for our artistic purposes. Because of its enormous functions on one hand and lightweight operation on other, it has become my favorite tool for wireless network sonification and will be discussed more later on in the text.

In the beginning of my research I was considering few options, ways, and processes. I was motivated to learn C programming related to audio, so that became my first choice. It has started by writing my own sniffer, based on library developed by same tcpdump developers. Library is called **libpcap** (PCAP stands for Packet CAPture) and serves for implementation of tcpdump functionality to one's software. Obviously, this wasn't very easy task to do, especially for unexperienced C programmer as myself. After I finally managed to sniff data using my software, the next step was to use **PortAudio** library to implement the audio part of the project. (WHAT MORE?)

Finally I partly succeeded, by writing “Sonodump”. Sonodump worked exactly as I wanted - it ‘sniffed’ the network data and reflected current traffic. The conversion was done through treating data harvested from the network as audio data - simple WAV file. User has only two options to select from - which interface should be used (wireless or wired) and what sample rate is expected. Use of the term sample rate is a bit inaccurate, because there is no correct, real sample rate of the stream - it is not audio stream. We can just guess or purposefully select some value to affect the way sonification behaves. Lower sample rates result in rather low-frequency based sonic content and higher contain more higher frequencies. This occurs simply because of the speed the network data are read, as it can be described as an obscure case of wavetable synthesis with tables generated in the network stream.

Sonodump already gave some interesting results, yet it wasn’t very efficient. My programming skills weren’t just on the right level to optimize the code properly. Therefore I need to look for other options. In autumn 2011 I was asked by organizer of AudioArt festival in Krakow, Marek Chołoniewski to present my sonification work on his festival. I happily agreed and immediately thought of wireless sonification project. But Sonodump wasn’t enough in this case - I needed something more adjustable, powerful and interesting.

One of the questions was whether to treat the sonification process in more traditional way, lacking interaction with the stream which is being sonified. I felt inner conflict between keeping certain level of purity and working on the most successful and pleasant aesthetic experience. In the end the idea of live interaction with the stream of data won, with argument planning for achieving non-traditional way of control over the resulting sound. With image of this wild, partly unpredictable instrument I started to research other technical solutions.

As mentioned before, Sonodump was not optimal and had few bugs, and in a meanwhile I discovered very useful set of commands present on Linux systems. Main motivation for work with Linux / UNIX was system shell - terminal, allowing scripting and importantly for this project, *pipelining* and *redirecting*, providing user with methods for interconnecting processes and files. For example, output of one process can be streamed in real time to another without user caring about any ‘middle-man’ text file or thinking about memory allocation. This comes particularly handy when you want to *sonify* something.

Good example is redirecting of an non-audio file to sound card. What this represents is treating the file as a stream of data consequently output to the sound card as it would be audio data - coming for example from a mp3 player. On some distributions it is enough to write this command to the terminal:

```
cat SONIFY_ME.txt > /dev/dsp
```

Command `cat` read the file `SONIFY_ME.txt` - which is an regular text file and

using the symbol ¨>¨ bash takes care of writing it to ¨/dev/dsp¨ which is a virtual port for sound card. Writing to this port opens a stream on the sound card and plays given data as audio. I have ran into few artists using this kind of sonification in their work already. **MENTION SOMETHING?**

Basic process of reading network data and converting them directly to sound is not much harder. The `cat` command is now replace with (tcpdump), which is a tool mentioned before - traffic analyzer, *sniffer*.

```
tcpdump -i wlan0 -A > /dev/dsp
```

Tcpdump opens a packet capture at device ẅlan0¨(wireless module). Option ¨-Am̈eans conversion of the characters to ASCII symbols (so we can always read them in terminal). This already gives us some amount of sonification of the network with just single line of code.

2.3 City attributes sonification

Chapter 3

Modularity and physical instruments

3.1 Reinvention of the physical musical environment

3.2 Musical use of microcontrollers and low-level computers

Chapter 4

Conclusion

Chapter 5

References

References

Collins, N. (2006). Noises Off: Sound Beyond music. *Leonardo Music Journal*, *16*, 7–8.

Kramer, G., Walker, B., Bonebright, T., Cook, P., Flowers, J., Miner, N., & Neuhoff, J. (1999). *The Sonification Report: Status of the Field and Research Agenda.* Retrieved 05/10/12, from `http://www.icad.org/websiteV2.0/References/nsf.html`

Pallavi, A., & Hemlata, P. (2012). Network Traffic Analysis Using Packet Sniffer. *International Journal of Engineering Research and Applications*, *2*(3), 854–856.