Wind and Wave Auditory Icons for Monitoring Continuous Processes

Stéphane Conversy
Laboratoire de Recherche en Informatique - URA 410 du CNRS
LRI - Bâtiment 490 - Université de Paris-Sud
91405 Orsay Cedex, France
conversy@lri.fr http://www-ihm.lri.fr/~conversy

ABSTRACT
This article presents the design and the use of two new auditory icons: the sounds of waves and wind. A synthesis algorithm is described to compute and control these sounds with high-level parameters in real-time. These auditory icons can be used effectively to monitor background activities, in particular when there is a need for continuous monitoring or when there is a need to prevent problems rather than to address them. They are a first step in the realization of controllable cohesive sound ecologies.

Keywords
Non-speech audio, auditory icons, background activities, continuous monitoring.

INTRODUCTION
Sound has a number of advantages over visual displays for presenting information: it does not take up any screen space; we can focus on a sound while hearing another one simultaneously; we can forget a sound and be aware of it again when it changes. These features allow us to monitor a background task with sound while in the midst of another activity.

Monitoring background activities with sound is effective only if it minimizes explicit queries for information by the user. This can be achieved by playing sounds either intermittently, such as ShareMon [2,3], or continuously, such as in the Arkola bottling plant simulation [4] or Gaver's machine sounds [5]. Intermittent sounds can act as a reminder that something is going on in the background. They are also useful because it can provide serendipitous information [7], i.e. information we find relevant later, or information we have not explicitly queried. With continuous notification, information is available immediately, by focusing on the sound providing it. Thus, continuous notification allows prevention as opposed to cure: when a sound changes, it may be a harbinger of an impending issue that we can address before it happens. For example, it is more interesting to know that the paper tray is about to run out of paper than to be notified that it is empty.

Since we often need to monitor a task continuously, audio interfaces have to use non-intrusive, varied and informative sounds. Gaver's theory of auditory perception [6] states that humans hear non-musical sounds as cues of what is going on around them: they analyze these sounds as auditory events with high-level attributes, such as the material of a door slamming. This has led Gaver to introduce *auditory icons*, i.e. everyday sounds for use in the interface that can be described with high-level parameters. This article introduces two new auditory icons imitating natural environments: wind and waves. These sounds and their behaviours are well known to humans, and they do not disturb them as long as they have a low level. We have developed two algorithms to synthesize and control these sounds in real-time and have used them in several applications.

SYNTHESIZING WIND AND WAVES
Although sampling natural sounds gives higher quality results than synthesizing them, we have chosen synthesis to get better control over the sounds. We do not need sounds that imitate a real sound perfectly. Sounds only need to be recognized as the result of actual events, from which high-level parameters can be extracted and evaluated.

Both algorithms are based on dynamic filtering of a white noise. The filter is an IIR (infinite impulse response) filter, whose main effect is to emphasize a set of frequencies. Its parameters are the center frequency it emphasizes and the bandwitdth of the main lobe. Both parameters evolve with time envelopes controlled by high-level attributes.

Wind sound
The wind sound is a noise gliding up and down continuously, according to a rate that depends on its strength. The higher the strength, the smaller the rate and the higher the mean pitch. Thus, a *strength* parameter controls the domain over which the frequencies of the filter are randomly chosen, and the domain over which the times of change of the gliding direction are randomly chosen. The next frequency to reach is $f=100+rand \times strength \times 10$ and the next time (in ms) is $t=rand \times (110\text{-}strength) \times 50$. *Rand* returns a random value in the range [0, 1[and *strength* is in the range [1, 100]. The bandwidth of the filter is 60 Hz.

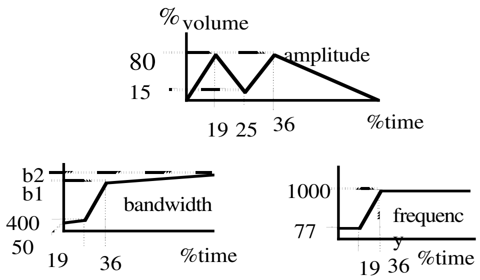

Figure 1: Envelopes for the wave sound

Wave sound

A breaking wave can be divided into three parts: the wave breaks (high volume), it rumbles (lower volume, low pitch) until it reaches the beach (more noise and higher pitch). High-level parameters for a wave sound are the *size* which controls the length of the sound, the *shape* which makes the wave spread widely on the beach or break onto itself, and the *beach*, which makes the wave sound as if it breaks on a different kind of beach, e.g. sand or rocks.

Figure 1 shows the envelopes for the overall amplitude of the sound and the center frequency and bandwidth of the filter. The control points have their abscissas expressed as a percentage of the total length of the sound. The *size* controls the total length of the sound (typically 2 to 5 seconds), and thus, the abscissas of the control points. The *beach* acts as a gate for the noise : instead of choosing a value for each sample, we pick a new value every one, two, three or four samples. This makes the wave sound more or less noisy. The *shape* (in the range [1, 100]) controls the end of the envelope of the bandwidth. The higher the *shape*, the higher the b_n values: $b_1 = 400 + rand \times 1000 \times shape/30$, and $b_2 = b_1 + rand \times 1000 \times shape/20$.

Implementation

With careful coding, the algorithms can be implemented efficiently. The wind sound requires three multiplies, one add and one look-up in a table per sample. The wave sound requires seven multiplies, four adds, and two look-ups in a table per sample. The sounds have been incorporated within the ENO audio server [1]. By using a filtered noise, slow attacks and random values, we avoid high pitched, high volume recurrent sounds, making them suitable for a long play without annoyance.

APPLICATION

These sounds correspond to natural phenomena that have no obvious mapping with computing concepts, unlike, e.g., folders in the desktop metaphor. In order to take advantage of the users' knowledge of these sounds, the processes being monitored with these sounds should have a behavior that matches as closely as possible that of the natural events. Since the natural phenomena evolve slowly, using these sounds to reflect a fast-changing activity would confuse the user. Furthermore, perceiving a perturbation in environment should lead the user to infer that something is going wrong. Thus, using the wind sound to monitor a network flow is inappropriate, since a strong wind would mean a high throughput, which means that the network is performing well. Instead, the strength of the wind should be bound for example to the number of lost packets.

Our current application for these sounds is a process monitor for a network of workstations. We are experimenting with various mappings for the sounds and their parameters. The main idea is that the sound should be noticed when something goes wrong, e.g. overloaded network or workstation, broken hardware. This requires the use of additional sounds for events (as opposed to continuous processes) and a spatial layout of the sounds to help monitor several of them at once.

CONCLUSION

We have introduced two new auditory icons: wind and waves. The synthesis algorithms producing them are cheap enough to synthesize them in real-time. They follow the auditory icon paradigm and they can be controlled in real time with high-level parameters. Informal tests showed that users recognize them easily, and can track changes in the parameters.

The wind and wave sounds are a new step in providing a sound toolkit for interface designers. Future works should produce more sounds, as well as tools to design them.

REFERENCES

1. Beaudouin-Lafon, M., and Gaver, W. W. ENO: Synthesizing Structured Sound Spaces. In *Proc. Symposium on User Interface Software Technology, UIST'94*, ACM, 49-57.
2. Cohen, J. Monitoring Background Activities. In *Proc.International Conference on Auditory Display, ICAD'92*, 499-531.
3. Cohen, J. "Kirk Here:" Using Genre Sounds To Monitor Background Activity. *In Adjunct Proc. Human Factors in Computing Systems, INTERCHI'93, ACM*, 63-64.
4. Gaver, W. W., Smith, R. B. Auditory Icons in Large-Scale Collaborative Environments. In *Proc. Human-Computer Interaction, Interact'90*, ACM, 735-740, 1990.
5. Gaver, W. W. Synthesizing Auditory Icons. In *Proc. Human Factors In Computing Systems, INTERCHI'93*, ACM.
6. Gaver, W. W. What In The World Do We Hear ? An Ecological Approach To Auditory Event Perception. In *Ecological Psychology* (5) 1 (1993).
7. Mynatt, E. D., Back, M., Want, R., Baer, M., Ellis, J. B. Designing Audio Aura. In *Proc. Human Factors in Computing Systems, CHI'98*, ACM.