SimScene : a web-based acoustic scenes simulator

Mathias Rossignol, Gregoire Lafay, Mathieu Lagrange, Nicolas Misdariis

SimScene: a web-based acoustic scenes simulator

Mathias Rossignol
IRCAM
1 place IgorStravinsky
Paris, France

Gregoire Lafay
IRCCyN, École Centrale de Nantes
1 rue de la Noé
Nantes, France

Mathieu Lagrange
IRCCyN, École Centrale de Nantes
1 rue de la Noé
Nantes, France

Nicolas Misdarris
IRCAM
1 place IgorStravinsky
Paris, France

ABSTRACT

We introduce in this paper a soundscape simulator called SimScene, designed to be used as an experimental tool to characterize the mental representation of sound environments. The soundscape simulator allows a subject to generate a full sonic environment by sequencing and mixing sound elements, and manipulating their sound level and time positioning. To make the simulation process effective, SimScene has not be designed to manipulate individual parameters of individual sounds, but to specify high-level parameters for whole classes of sounds, organized into a hierarchical semantically structured dataset. To avoid any linguistic bias, a listening oriented interface allows subjects to explore the dataset without any text written help. The entire software is developed in Javascript using the standard Web Audio technology, and is thus fully supported by most modern web browsers. This fact should allow experimenters to adopt a crowdsourcing approach to experimentation in order to assess hypotheses on large populations, and facilitate the development of experimental protocols to investigate the influence of socio-cultural background on soundscape perception.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: Signal analysis, synthesis, and processing

General Terms

Audio, Simulation, Human Factors

Keywords

sound perception, cognitive psychology, soundscape, web-based soundscape simulator, web-based acoustic scenes simulator, web-based sound environment generator

Web Audio Conference 2014 IRCAM, Paris France

1. INTRODUCTION

The notion of soundscape has been introduced by the well known book *The tuning of the world* of R.M. Schafer [16], as a concept which tends to describe the sonic environment by putting focus on the listener's appreciation. Applying this concept to the acoustical research community led to a growing amount of interdisciplinary projects [16][5][6][17], investigating the way humans perceive and decompose sound environments. As those studies come from different research traditions, new tools are needed to investigate soundscape perception to allow experimenters to integrate both qualitative and quantitative methodologies. A relevant way to tackle this issue is to ask a subject to recreate a particular sound environment, and characterize it. In that sense, a soundscape simulator may offer new ways to investigate soundscape composition and perception. We propose that analysing the simulation process, in other words the chosen sound sources and the acoustical parameters applied to them, may provide useful data to better understand how human mental representations of sound environments are structured.

Some generative systems have already been proposed to generate sound environments [12][13][18][7]. Those methods are mostly designed to be used in the sonification of virtual sound environments[18] [7]. They may be seen as semi-autonomous generative system as they do not take user interaction as unique input, but also visual descriptions or soundscape recordings. Other methods that have been proposed and take user interaction as input are designed to aid composition, and thus not suited for perceptual studies [12][13].

To the best of our knowledge, only the work of Bruce *et al* [4] [3] tackles the issue of proposing a tool that 1) allows humans to manipulate or create a sound environment, and 2) is suited for perceptive experimental research. The tool proposed by Bruce *et al* allows subjects to add or remove sound sources, change sound level and other acoustical parameters, and set the source positions in the space. Using this framework, they asked subjects to manipulate sound source recordings of a urban square and found that sources inclusion or exclusion depends more on social expectation than acoustic features. Bruce *et al* point out the fact that the lack of available sources limits the analysis, and suggest to group sound sources into "semantic groups" to circumvent this issue.

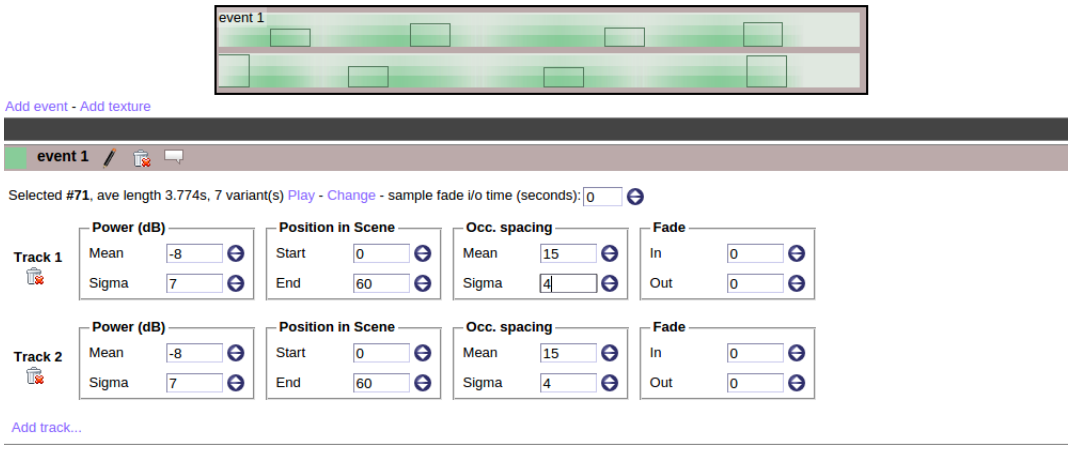

Figure 1: Two distinct instantiations (realizations) of two identical sound tracks having the same parameters.

We introduce in this paper a web-based simulator that can be used to create acoustic scenes. In order to ease the design of those acoustic scenes, the occurrences of each source in the scene are not individually micro-managed, but controlled globally using a reduced set of high level parameters. Exact sequencing information, *i.e.* which audio sample is played at what time and at which level, is determined at runtime. This paradigm allows the user to focus on the global aspect of the acoustic scene. A specifically tailored display is proposed to conveniently show the influence of the stochastic controlling parameters and their exact actualization. The simulator works with a semantically structured sound dataset that users may explore without any written textual help. SimScene provides the subjects with a set of acoustical parameters to shape the sound tracks in terms of intensity and time positioning. In order to easily reach a wide population of potential users and test subjects, and facilitate socio-cultural studies through crowdsourcing experiment, all the software is developed using Javascript libraries, relying on the widening spread of the Web Audio standard, and is thus supported by most current web browsers (currently Mozilla Firefox, Google Chrome, Safari and Opera).

We shall in this paper focus on the architecture and operation of SimScene, inviting the reader to refer to [8] for examples of experimental results obtained thanks to this system. After introducing the general concept of the simulator and the sound dataset used in our work, we present the SimScene mechanics and software architecture.

2. SIMULATOR MODEL

To allow users to create sound environment, a generative model is designed. The proposed model takes into account morphological and perceptive considerations. In this model, the basic elements are classes of sounds and not individual sounds. Classes of sounds are collections of semantically identical sounds as for example the classes "passing-car" or "passing-scooter".

To ease the creation of soundscape, the simulator assumes a perceptually inspired distinction made between the sounds populating the sonic world which may be summed up in the sentence of Nelken and Cheveigné [14] concerning the soundscapes: "a skeleton of events on a bed of texture". The distinction between the so called sound events and sound textures is perceptively motivated as several studies point out the fact that this two types of sounds provoke two distinct cognitive processes [9], [11] [10]. Roughly speaking, short and salient sounds are considered as events ("car passing", "male yelling", "bird singing"), and long and amorphous sounds are considered as textures ("wind", "rain", "street hubbub", "urban square hubbub").

Considering that $s(n)$ is a given acoustic scene composed of C sound classes c_i, the proposed model is such that:

$$s(n) = \sum_{i=1}^{C} t_i(n) \tag{1}$$

Where each t_i is a semantic sound track. For the sake of simplicity, we only detail here the model of an events track, then explain the adaptation of the model to texture tracks.

t_i is defined as a sequence of n_i sound events $e_i^k(n)$ randomly chosen among the $|c_i|$ samples in class c_i: for each k in $[1..n_i]$, $e_i^k = c_{i\,\mathcal{U}(1,|c_i|)}$, where $\mathcal{U}(a,b)$ represents an uniformly distributed integer random value between a and b included. Each event is scaled by an amplitude factor sampled from a real normal distribution with average μ_i^a and variance σ_i^a. The interval separating the offset times of consecutive samples for track i is, similarly, randomly chosen following a normal distribution with average μ_i^t and variance σ_i^t. Formally, each sequence t_i is thus expressed as:

$$t_i(n) = \sum_{j=1}^{n_i} \mathcal{N}(\mu_i^a, \sigma_i^a) c_{i\,\mathcal{U}(1,|c_i|)}(n - n_i^j) \tag{2}$$

$$n_i^j = n_i^{j-1} + \mathcal{N}(\mu_i^t, \sigma_i^t) \tag{3}$$

where n_i^0 is set to 0 by convention. The signal of an event is defined in such a way that $e(n) = 0$ if $n < 0$ or beyond the signal's duration.

In the case of a texture track, two implementation differences must be observed to maintain a perceptually acceptable output: first, signal amplitude is only drawn at random once, and that value is applied to all samples; second, sample start times are not randomized but chosen so that the texture recordings chosen from class c_i will be played back-to-

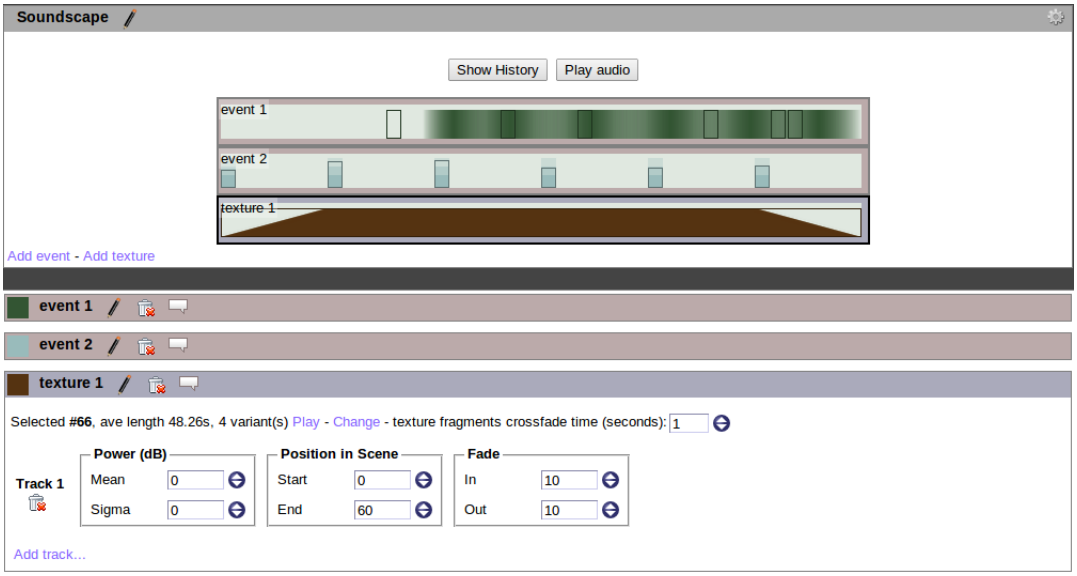

Figure 2: The interface of SimScene using with three tracks: the two first tracks correspond to event classes, and the last one to a texture class. Tracks are shown on top, and audio parameters at the bottom. In this example, parameters of the event tracks are folded.

back with sufficient overlap to create an equal-power crossfade between them, thus generating a continuous, seamless track.

For both event or texture classes, users may interact with μ^a and σ^a to set sound intensities. For events classes, users may interact with μ^t and σ^t to set the time between sound events. In that sense, the manipulation adopt a stochastic approach, as the simulated scene is only an instance of a multitude of other potential scenes. Figure 1 illustrates the fact that a same set of parameters applied to two identical sound classes can lead to two distinct instantiations.

3. USE OF THE SIMULATOR

We outline here the general concepts and workflow of the SimScene interface, and will delve more precisely on design and implementation details in the following sections.

3.1 Interface

To some extent, the simulator interface is close to that of an audio sequencer, with the display divided into two areas: the top part shows the audio tracks, whereas the bottom part shows the audio controllers related to each track. Each track has one distinct set of audio parameters. Figure 2 shows a example of the SimScene interface.

To add a sound track, the user clicks on the *add event* or *add texture* button, thus opening the corresponding sound selection dialog (please refer to Section 4.2 for an in-depth description of the sound selection process). Once a sound class is selected, the selection interface is removed, and an audio track bound to the selected sound class is created and appears at the top of the interface.

In contrast with traditional audio sequencers where tracks are time dependent acoustical representations of an audio signal (most of the time the sound pressure level) the tracks of the simulator are symbolic representations of temporal sequences of fragments. The way fragment representations are handled depends on the type of sound class, event or texture, bound to the track. When a user selects a sound event class, all the fragments of the sound collection of the class are chosen, and depicted with color blocks. The number and time positions of the fragments depend on the audio parameters. When a user selects a texture class, one color block is displayed along the entire track length.

The color block appearances respond to the control parameter values. Block height is related to the *sound intensity* whereas block position depends on the *start*, *stop* and *mean between events parameters*. A blur effect is used to illustrate amount of variance (in effect, randomization) applied to the parameter values: the higher the variance parameters, and stronger the blurring effect. Black outlined blocks indicate the position of instantiated items—as precised above, those positions are only particular instantiations resulting from the stochastic parameters. Global fade in and fade out parameters are illustrated by a rising, resp. falling, ramp.

3.2 Sound performance

Once a user has added all the required tracks, and adjusted the parameters to his liking, he may of course listen to the resulting scene: the system downloads, adjusts and schedules the performance of sounds fragments on the fly, and if the user modifies the parameters during the performance those modifications will be taken into account. This can be a useful tool in the typical "listen-adjust" loop of sound production, however it should be noted that if parameters include a random factor the changes may not always be "smooth".

The random assignment of sound fragments, within collections, to each instance of a sound on a track, is drawn again each time the scene is to be played, thus making each execution unique. Concerning textures, if diverse fragments are available, they are randomly chosen and concatenated (with a seamless crossfade) in an order chosen to avoid any

immediate repetition of the same fragments. The user is therefore forced to think in terms of global sound classes, rather than individual sounds.

Once a user is satisfied with their production, they may download and save a full rendered mixdown of their composition.

3.3 Usage for experimentation

Should it be needed, the system provides mechanisms to observe and save to a server-side database the users' actions at various levels of granularity: from a simple snapshot of the final state of the composition, all the way down to a full recording of every action taken on the interface during the composition session, with millisecond-precise time stamps. Care should be taken to adapt the choice of what activity to record to the intended analysis task, since over-abundance of data may make the collected information nearly impossible to process.

4. SOUND FRAGMENT SELECTION

In other psychological methods such as interview or questionnaires, subjects are asked to describe a sound environment and thus are only limited by their memory and by the extent of their vocabulary. In our case the sound data set plays a key role as subject expressiveness is conditional upon its diversity, organization and accessibility.

4.1 Dataset structure

Simscene currently makes use of a dataset of urban environmental sounds which we have developed; we shall pick our examples from it, but other sound sets are of course possible, and planned.

In this dataset, short sounds such as "women-footstep", "male-yelling" or "passing-car" are considered as sound events, whereas long sounds such as "rain", "wind", "crowd" or "urban-traffic-hubbub" are considered as sound textures.

In order to make the dataset usable and accessible, segmenting the sonic world to provide the subject with a reasonable and representative number of sound events and textures is a crucial step. Several studies point to the lack of standardized taxonomy of sounds due to the high variability of terms used to described a same sonic environment [15] [2]. Considering that, the dataset is organized into two hierarchical structures of semantic classes, one for events, the other for textures. The top classes of those structures represent high-level domains, such as "urban transport", regrouping a variety of general categories, like "boat" or "car", which are themselves organized into sound classes ("starting-up-car", "passing-car"), *etc.* The deeper the level of a class, the lower the variability between exemplars gathered into that class.

Figure 5 depicts the hierarchical structure used for both the event and texture datasets. Subjects interact only with the leaf classes of the two structures (event and texture), whose hierarchical organisation is designed to help subjects explore the datasets, as explained in Section 4.2

4.2 Selection Process

Sound selection is considered carefully. The aim is to propose a selection interface which enables users :

- to quickly understand the size, variance and the organization of our sound dataset;
- to quickly find a desired sound.

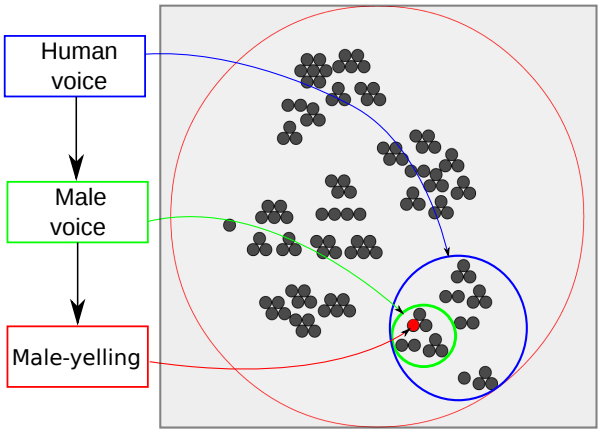

Figure 3: A example of display for the selection interface

These two goals are dictated by experimental constraints, as it is a priority that subjects be able to achieve the experiment (re-create a soundscape) in a given amount of time. Another implicit constraint, inherent to the experimental protocol, is that the selection interface should not introduce any bias in the selection process: subjects must not be influenced in their sound choices to re create their mental representation of a particular soundscape. In order to avoid the need of any verbal index, a listening oriented interface is designed to explore the sound data set without any written textual help.

Sound classes are represented by circles distributed on a 2D space and packed together according to the hierarchical structure of the dataset (see Figure 3). Close classes in the hierarchy are thus represented by close circles in the selection interface display. If the hierarchy is modified, the selection interface display is automatically modified. Circle packing functions of the *D3.js* [1] library are used to display the hierarchy. Figure 3 shows the selection interface display of the sound event classes of the urban environmental sound dataset, and illustrates the way in which the spatial organisation of circles is linked to the semantic hierarchy of the dataset. As there are two dataset structures, one for the events and one for the textures, there are two distinct selection interface displays.

To explore the sound dataset, users simply click on the circles, which triggers the "sound prototype" for that class, that is, the item considered to be the most representative of the class.

4.3 Dataset Management

Normal users only ever experience the sound dataset for browsing and selection; we quickly present here its development process.

To create the dataset, developers upload to the server full unsegmented recordings called "source sounds". Once a source sound is uploaded, the developer selects on a web interface segments of it which are called items, and can be either an audio event or and audio texture, then tags them according to their physical source. Figure 4 gives an example of this process. In this case, the source sound is a recording of street where scooters and cars pass by. Each occurrence

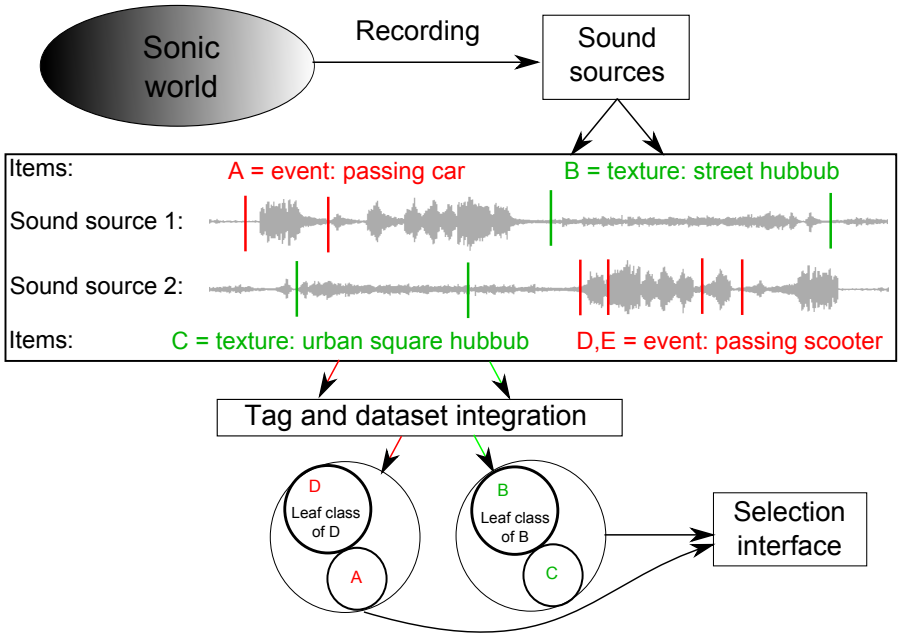

Figure 4: Dataset Management: the leaf class of D groups all items of the "passing scooter" category, including E (similarly for A, B and C)

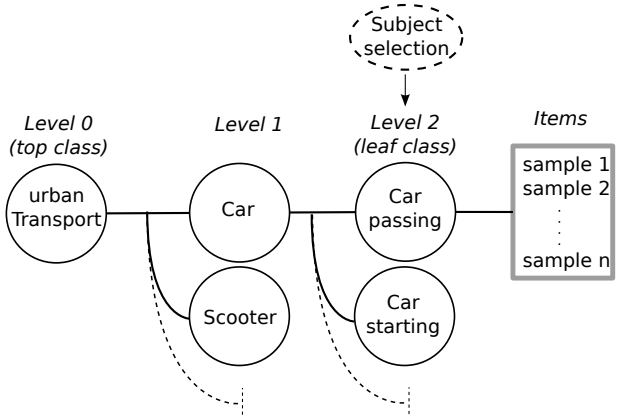

Figure 5: Scheme of the dataset hierarchical structure used both for the event and texture datasets. Depending on the considered top class, there could be more than 3 levels.

of a scooter or car passing by is selected as a sound event, and appropriately tagged. Lastly, a sequence of the source sound where no distinguishable event occurs is selected as a sound texture, and tagged "street hubbub".

5. AUDIO CONTROL PARAMETERS

A central concept of SimScene it that users have no precise control on the scene: first, they do not select or interact with individual sounds, but with a group of sounds belonging to the same semantic class; second, concerning the scheduling of those sound collections, they only control high-level indications. The available audio controllers allow users to scale the sequence of items along two dimensions, one being the sound intensity, and the other the time positioning. Table 1 lists the available audio parameters.

Audio Parameters	Description
Sound intensity (dB)	Average and variance
Spacing time (sec) (only for events)	Average and variance
Position in Scene (sec)	Start and Stop
Fade In/out (sec)	Global
Fade In/out (sec) (only for events)	For each event item

Table 1: Available parameters to control audio tracks.

Three audio controllers are used to temporally scale the items sequences. The *start* and *stop* parameters set the beginning and the end of the sequence in the scene, whereas the *spacing time* parameters, for events only, allow user to set the average time interval at which items shall be repeated. As users control sequences and not item, spacing time between items is defined in terms of average and variance.

For sound intensity, three audio controllers are considered. The *sound intensity* parameter, which is also defined in terms of average and variance, the *global fade In/out* parameter, which applies a fade effect on all the items sequence, and the *fade In/out* parameter, relevant for events only, which applies a fade effect separately on each item.

6. SOFTWARE ARCHITECTURE

A major feature for any system aimed at experiment crowdsourcing is that it should be hassle-free for the end user: any request for volunteers to use a specific OS, install software,

configure their hardware, *etc.* will reduce the size of the volunteers pool. We therefore chose to develop Simscene to run in a web browser, taking advantage of the Web Audio API[1].

The Web Audio API appeared in 2011, as a product of the W3C Web Audio working group, and was quickly implemented, first in the Chrome web browser, then in Firefox. As of October 2014, the estimated support figures are that about 65 % of users worldwide can run a webpage based upon that API, which makes it ideal for our purpose. Development was longer and most convoluted than expected, due to the fact that we started work in 2012, while the standard was still in evolution and its implementation still imperfect: it was for example necessary for a while to use custom implementations of gain control nodes due to sound quality issues with the Chrome implementation. Two years later, however, it seems that the standard has reached maturity, and while some work still needs to be done to ensure that Web Audio applications can run on all browsers, it clearly appears as a reliable solution.

In order to maintain a consistent interface preserving a constant synchronization between the complex description of an audio scene and its visual depiction in the interface, we make use, in addition to the D3.js library already mentioned, of the data-biding abilities of the Angular framework[2]. This allows us to fully focus on program logic and sound production.

7. CONCLUSION

We have presented in this paper an acoustic scene simulator designed to run in a web browser, for example to be used as a tool for crowd-sourced perceptual experiments. The simulator is used to generate realistic-sounding soundscapes from elementary samples of single events or textures. By only letting users control the positioning and intensity of those samples through high-level parameters, and giving no control on individual sounds, the interface we propose leads users to focus on semantic sonic constructions and global scene perception, thus making their behavior more easily analyzable from a cognitive or perceptual point of view.

8. ACKNOWLEDGEMENTS

Research project partly funded by ANR-11-JS03-005-01.

9. REFERENCES

[1] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011.

[2] A. Brown, J. Kang, and T. Gjestland. Towards standardization in soundscape preference assessment. *Applied Acoustics*, 72(6):387–392, May 2011.

[3] N. S. Bruce and W. J. Davies. The effects of expectation on the perception of soundscapes. *Applied Acoustics*, 85:1–11, 2014.

[4] N. S. Bruce, W. J. Davies, and M. D. Adams. Development of a soundscape simulator tool. *proceedings of Internoise 2009*, 2009.

[5] W. J. Davies, M. D. Adams, N. Bruce, M. Marselle, R. Cain, P. Jennings, J. Poxon, A. Carlyle, P. Cusack, D. A. Hall, et al. The positive soundscape project: A synthesis of results from many disciplines. *proceedings of Internoise 2009*, 2009.

[6] W. J. Davies, M. D. Adams, N. S. Bruce, R. Cain, A. Carlyle, P. Cusack, D. A. Hall, K. I. Hume, A. Irwin, P. Jennings, M. Marselle, C. J. Plack, and J. Poxon. Perception of soundscapes: An interdisciplinary approach. *Applied Acoustics*, 74(2):224–231, Feb. 2013.

[7] N. Finney and J. Janer. Soundscape generation for virtual environments using community-provided audio databases. In *W3C Workshop: Augmented Reality on the Web*, 2010.

[8] G. Lafay, M. Rossignol, N. Misdariis, M. Lagrange, and J.-F. Petiot. A new experimental approach for urban soundscape characterization based on sound manipulation : A pilot study. In *Proceedings of the 26th International Symposium on Musical Acoustics*, 2014.

[9] V. Maffiolo. *De la caractérisation sémantique et acoustique de la qualité sonore de lâĂŹenvironnement urbain, ("Semantic and acoustical characterisation of the sound quality of urban environment")*. PhD thesis, Université du Maine, Le Mans, France, 1999.

[10] J. H. McDermott, M. Schemitsch, and E. P. Simoncelli. Summary statistics in auditory perception. *Nature neuroscience*, 16(4):493–498, 2013.

[11] J. H. McDermott and E. P. Simoncelli. Sound texture perception via statistics of the auditory periphery: Evidence from sound synthesis. *Neuron*, 71(5):926–940, Sept. 2011.

[12] A. Misra, P. R. Cook, and G. Wang. A new paradigm for sound design. In *Proceedings of the International Conference on Digital Audio Effects (DAFx-06)*, pages 319–324. Citeseer, 2006.

[13] A. Misra, G. Wang, and P. Cook. Musical tapestry: Re-composing natural soundsâĂă. *Journal of New Music Research*, 36(4):241–250, 2007.

[14] I. Nelken and A. de Cheveigé. An ear for statistics. *Nature neuroscience*, 16(4):381âĂŞ–382, 2013.

[15] M. Niessen, C. Cance, and D. Dubois. Categories for soundscape: toward a hybrid classification. In *Inter-Noise and Noise-Con Congress and Conference Proceedings*, volume 2010, pages 5816–5829, 2010.

[16] R. Schafer. *The Tuning of the World*, pages 1–301. Borzoi book. Knopf, New York, 1977.

[17] B. Schulte-Fortkamp, B. M. Brooks, and W. R. Bray. Soundscape: An approach to rely on human perception and expertise in the post-modern community noise era. *Acoustics Today*, 3(1):7âĂŞ15, 2007.

[18] A. Valle, M. Schirosa, and V. Lombardo. A framework for soundscape analysis and re-synthesis. *Proceedings of the SMC*, pages 13–18, 2009.

[1] Web Audio specification: http://www.w3.org/TR/webaudio/

[2] Angular library: https://angularjs.org/