A Web Audio Node for the Fast Creation of Natural Language Interfaces for Audio Production

Michael Donovan
Northwestern University
mbdono56@gmail.com

Prem Seetharaman
Northwestern University
prem@u.northwestern.edu

Bryan Pardo
Northwestern University
pardo@cs.northwestern.edu

ABSTRACT

Audio production involves the use of tools such as reverberators, compressors, and equalizers to transform raw audio into a state ready for public consumption. These tools are in wide use by both musicians and expert audio engineers for this purpose. The typical interfaces for these tools use low-level signal parameters as controls for the audio effect. These signal parameters often have unintuitive names such as “feedback” or “low-high” that have little meaning to many people. This makes them difficult to use and learn for many people. Such low-level interfaces are also common throughout audio production interfaces using the Web Audio API. Recent work in bridging the semantic gap between verbal descriptions of audio effects (e.g. “underwater”, “warm”, “bright”) and low-level signal parameters has resulted in provably better interfaces for a population of laypeople. In that work, a vocabulary of hundreds of descriptive terms was crowdsourced, along with their mappings to audio effects settings for reverberation and equalization. In this paper, we present a Web Audio node that lets web developers leverage this vocabulary to easily create web-based audio effects tools that use natural language interfaces. Our Web Audio node and additional documentation can be accessed at https://interactiveaudiolab.github.io/audealize_api.

1. INTRODUCTION

We wish to help builders of audio production tools, such as reverberators, compressors, and equalizers, make easy-to-use interfaces so that a broader range of people will be empowered to use these tools. In this work we describe the Audealize API, which provides a natural language interface for controlling two audio effects: equalization and reverberation. The API provides an AudioNode for Web Audio that uses crowdsourced mappings between descriptions and low-level signal parameters to allow a user to control the settings of each effect by specifying a word (e.g. “bright”, “warm”) that describes their desired sound.

Audio production tools are used by musicians, engineers, producers, v-loggers, podcasters and others to transform raw audio into a state ready for public consumption. Two of the most popular production tools are reverberation and equalization. Reverberation is used to add echoes to a recording to make it sound as if it is in some space (e.g. a cathedral), to make the sound better (e.g. “warmer”) or make it more interesting (e.g. “chaotic”). Equalization is used to cut or boost frequencies in the sound to make it sound, for example, “muffled” or “bright.” End-user interfaces for typical reverberators, compressors and equalizers have steep learning curves, due to their reliance on using knobs and dials with labels relating to low-level signal parameters to control the audio effect, such as the equalizer shown in Figure 1. This forces the user to navigate a space of low-level signal parameters to find the desired effect.

Experienced engineers can use this signal-parameter space effectively and tool developers are typically experienced engineers. Therefore, interfaces to these tools typically reflect this mindset. There is, however, a conceptual gap between these tool builders and many potential users of these tools. Example users who often do not know the technical language of audio production include acoustic musicians, podcasters, v-loggers, and music hobbyists. Such people typically formulate their thoughts about acoustic concepts using colloquial terms and communicate audio concepts with those terms. A well known engineer described this phenomenon as follows: “It’s a situation all engineers have been in, where a musician is frustrated trying to explain to you the sound he or she is after, but lacking your ability to describe it in terms that relate to technology, can only abstract. I have been asked to make things more ‘pinky blue’, ‘Castrol GTXy’ and...‘buttery’” [2].

This interface paradigm (low-level controls) for traditional audio production tools has been transferred to Web Audio [9] implementations of audio production tools. However, many users of audio applications on the web are likely to be laypeople, due to the low barrier of entry (no installation beyond your browser required) and ease of sharing (just linking others to a page would have them using the tool). Because of this, it is important to facilitate the creation of audio production tools on the web that are accessible to laypeople.

Designers of traditional interfaces have tried to address this accessibility problem by introducing “presets”, which are predefined settings for the audio production tool with a natural language label, defined by the tool designer. However, the vocabulary used by tool builders to name presets does not have a large overlap with the vocabulary laypeople use to describe audio effects. One prior study found...
2. RELATED WORK

The most closely related work to our own is by Stables et al. [14], in which they also gather data relating user effect descriptions and low-level signal parameter from interactions a user has with a VST (Virtual Studio Technology) or AudioUnit audio effect plugin, such as the one in Figure 1. Our work differs from theirs in two ways. First, it is meant for audio effects via the Web Audio API rather than VSTs. Second, our vocabulary is derived from crowdsourcing tasks on the web (via Amazon Mechanical Turk [8]), resulting in a vocabulary drawn from laypeople with little to no experience with audio production.

Mycroft et al. [6] [7] looks at the effect that traditional audio production interfaces have on creativity and cognition. They find that complex signal-parameter interfaces for audio effects, such as the one in Figure 1, can affect working memory and cognitive load, having an adverse effect on critical listening. Further, the visual paradigm adapted by equalizers can create perverse incentives (e.g. making a good-looking equalization curve, rather than making an equalization curve that sounds good). An interface that uses natural language is one way to circumvent these issues.

Schmidt [11] presented an early natural language interface for audio production. In the interface, rather than control audio effects, users could give commands like ‘Play/pause the bass’ and perform other basic audio production tasks. This had a limited vocabulary, determined by the system developer. Sabin et al. [10], made an interface for equalization where users alter the effect by navigating a two dimensional space, where the axes of the space are equalization settings along different perceptual parameters. It was parameterized with a vocabulary of four words. Mecklenburg et al. [4] built an equalizer interface where users can communicate to the system in subjective terms, enabling interactions such as making sounds “warmer” or “brighter”.

In Seetharaman et al. [13], users control an audio effect using a two dimensional word-map, as seen in Figure 4. Words that are nearby on the map are similar in terms of human perception of the audio effect.

In Seetharaman et al. [12] and Zheng et al. [15], users are presented with a lightweight survey in which users describe the effect an audio effect has on a sound using natural language. The datasets collected by Seetharaman et al. [12], Cartwright et al. [3], and Zheng et al. [15] form the basis for our proposed Web Audio node. For an overview of the crowdsourcing mechanisms and the mapping between words and low-level signal parameters, the reader is referred to [13].

The API we present here makes these mappings between crowdsourced vocabulary and actionable settings of audio effects available as an API so that anyone can build interfaces for audio effects based on natural language vocabulary. To our knowledge, this is the only such API available.

3. PROPOSED WEB AUDIO NODE

Our work builds on the Web Audio API, a JavaScript API that enables real-time audio synthesis and processing for web applications [9]. Our proposed Web Audio node provides a natural language interface for controlling two audio effects:
equalization and reverberation. The node allows a user to control the settings of each effect by specifying a word (e.g. “bright”, “warm”) that describes their desired sound. The node then processes any incoming audio first through the equalizer and then through the reverberator.

The underlying equalizer is the same one used in Audealize [13]. It is a graphic equalizer composed of forty peaking filters with center frequencies logarithmically spaced between 20Hz and 20kHz. Since the center frequencies are fixed, each filter is controlled only by a gain parameter. An equalization curve may then be described by forty gain values, given in dB, where positive values correspond to a boost and negative values correspond to a cut around a given center frequency.

For the underlying reverberator, we use the reverberator used in SocialReverb [12]. The reverberator is built around a network of six parallel comb filters and is controlled by five main parameters: the gain and delay of the first comb filter in the network (which dictate the gain and delay values of the rest of the filters), the delay between the two stereo channels (which controls the perceived width of the stereo effect), the center frequency of a low-pass filter applied to the output of the reverberator, and the ratio between the wet and dry signals.

3.1 API Reference

Our node implements the Web Audio AudioNode interface and contains two main child nodes, one for each effect. Our node takes as input an audio signal and processes it first through the equalizer node and then through the reverberation node. Each effect is controlled by three high-level parameters: the natural language descriptor, a parameter specifying the strength of the effect, and an on/off parameter. These parameters are exposed to the developer as member variables of our Web Audio node.

The “amount” parameter controls the intensity of the effect. Member variable eq_amount is a single scalar that multiplies the gain values of all 40 filters in the equalizer. The default value for eq_amount is 1.0, with higher values corresponding to a more intense application of the descriptor. A value of 0 yields a flat EQ curve, and negative values can be used to achieve the inverse effect. For example, setting eq_descriptor to “bright” and setting eq_amount to a negative value will apply an equalization effect that makes the sound less “bright.”

The intensity of the reverberation effect is controlled via reverb_amount. This value can be in the range [0,1] representing the ratio between wet and dry signals, with a value of 1.0 meaning that the node will output only the reverberation and none of the dry signal. The final two parameters, eq_on and reverb_on, are simply Boolean values indicating whether or not each effect is engaged. Setting either variable will cause the corresponding effect to immediately turn on or off. Both effects are bypassed by default and are enabled automatically when a descriptor is set.

The settings of each effect are controlled primarily by descriptors, one-word strings that describe the desired sound (e.g “tinny”, “underwater”). Each effect is controlled by a separate descriptor, represented as member variables eq_descriptor and reverb_descriptor. When set with a string, our node will search for a matching entry in the SocialFX dataset of 4297 words for the relevant effect and setting. If a match is found, the low-level signal parameters of the effect will be immediately set to the values corresponding to the descriptor. If the descriptor is not found, the node can optionally attempt to find synonyms using WordNet and return the closest synonym in SocialFX. When accessed, the descriptor member variables will return an object with the following properties:

- **word**: The natural language descriptor
- **effect**: The effect being described ("eq" or "reverb")
- **settings**: An array containing the settings of the low-level signal parameters as they are defined for the descriptor. For equalization descriptors, this contains gain values for each of the 40 filters.
- **num**: The number of unique user interaction sessions upon which the definition of the descriptor is based.
- **agreement**: A measure of the agreement between the definitions provided for the descriptor by the num user sessions. This indicates how confident we are in the assumption that the natural language descriptor and its corresponding audio effect settings match perceptually. Higher agreement values indicate higher confidence.
- **x, y**: The coordinates of the descriptor when a corresponding feature vector relating to its audio effect setting are mapped onto a 2-dimensional space using multidimensional scaling [1]. Descriptors that are near each other in this 2D space have similar perceptual characteristics when their corresponding audio effects are applied. This can be used to build interfaces such as the one in Figure 4.

4. EXAMPLE USES

Audealize, described in [13], offers one example of a natural language interface enabled by our Web Audio node. In this interface, equalization and reverberation descriptors are presented in separate 2-dimensional word-maps, one of which is shown in Figure 4. This lets the user select an effect
5. CONCLUSION

It is important for Web Audio applications to have more intuitive interfaces for audio production because the audience for these applications is far broader than those of traditional VST applications. This is due to the ease of access to web-based applications as well as the ease of sharing. However, the paradigm for audio production interfaces on the web is still dominated by traditional interfaces involving low-level signal parameters. We have created a Web Audio node that facilitates the creation of natural language interfaces for audio production on the web. This project is open source and we welcome pull requests. Our hope is that developers who use our Web Audio node will create more intuitive audio production interfaces for the web. Our Web Audio node and additional documentation can be accessed at https://interactiveaudiolab.github.io/audealize_api.

6. REFERENCES