Guitar pedal board using WebAudio

Michel Buffa
University of Nice
France
buffa@i3s.unice.fr

Maxime Demetrio
University of Nice
France
nourielazri@gmail.com

Nouriel Azria
University of Nice
France
Maxime.demetrio@gmail.com

ABSTRACT

The proposed demo is a guitar pedal board coded from scratch and inspired by the Guitar FX Chrome Application[1]. The demo is functional and proposes a graph based GUI for assembling effects such as: guitar amplifier emulation, Auto Wah, Delay, Distortion/Overdrive, Reverb, Chorus, etc. It includes a preset management system and saves incrementally, client side, any change in the parameters of the current preset. The latency is system and browser-dependent, but on a Mac Book Pro / Google Chrome the latency is as low as 14ms, making it playable in real time. The MVVC design is interesting for modularity and makes easy adding new effects. During the demo a guitar will be plugged into a computer using an external sound card, and the pedal board will process the sound in real-time. Lots of efforts have been put on the GUI and ease of use of the application, as well as on the core effect: the guitar amplifier simulation. The project is called "Guitar Processor" and is open source[2].

Latency and Web Audio problems

The getUserMedia API enables to use any sound input[3]. However by default, some browsers, such as Google Chrome implement an automatic gain adjustment algorithm[4] that is not suited for music applications. This option can be switched off using a constraint passed to the `navigator.getUserMedia` function (set the `echoCancellation` property to false). However, when we used this constraint, in the current implementations, it has not been possible to switch inputs (we reported the bug).

The second problem is latency. The intrinsic latency of a system (browser / OS / Sound Card) can be measured using Stephen Band's sound.io/latency[5] Web application, that measures the output-to-input latency (making a sound and listening for it), and results vary deeply from one configuration to another. Google Chrome/Mac OS x gives 9ms, where Firefox on the same computer gives 140ms. A "good latency" for playing real time an instrument is noticeable above 10ms. Add to this all the processing in our amp simulation and effects. So we did our best to avoid using the Web Audio nodes that involve additional latency. Unfortunately, this information is not specified, nor the nodes expose in a way or another their latency. Some APIs like JUCE propose some methods with its components[6] for "guessing the latency". In an interesting thread on stack overflow[7], it appears that only specific Web Audio nodes could involve latency overheads. They are: the WaveShaper node when set in oversampling mode (avoid oversampling!), as it has to look ahead a few samples to make some mean computations, the Delay (of course!), and the convolver node with large impulses (use small impulses!).

In our guitar amp simulator, we followed these good practices and manage to get a very low latency processing time, nearly unnoticeable, on Mac OS/Google Chrome. We measured the latency (make a sound and listen to it) and we got 15ms of latency (30ms round trip).

Pedal board application

Finally, in parallel to the work conducted in order to play with the "core sound" of a guitar amplifier, fixing latency and sound quality problems, we developed a graph based GUI for designing a chain of effects. Most of the effects are based on examples found on the Web; small improvements have been done to suit our ears.

Web Audio Conference WAC-2016, April 4–6, 2016, Atlanta, USA.

[1] Guitar FX Chrome Application : https://chrome.google.com/webstore/detail/guitarfx/nbjaofogegcpkeobcnpdkkpdioogeajj

[2] Guitar Processor : https://bitbucket.org/Felours/ntdp-projet_stage_guitarfx

[3] https://webrtc.github.io/samples/src/content/devices/multi/

[4] http://googletesting.blogspot.fr/2015/10/audio-testing-automatic-gain-control.html

[5] sound.io/latency

[6] getOutputLatencyInSamples from JUCE: http://www.juce.com/api/classAudioIODevice.html#a693804fbf5a7cceb31ece10a9f03bd11

[7] http://stackoverflow.com/questions/21428511/playing-and-recording-with-a-constant-latency-on-browser-with-javascript-usermed