More native echo cancellation!

By Oskar Sundbom
Software Engineer working on WebRTC and Audio in Chromium

We're continuing on from the previous experiment and in Chrome M68, we have added an experimental `MediaStreamTrack` constraint to control which echo canceller is being used, added support for a native echo canceller on Windows as well as improved the functionality of the native echo canceller on macOS. As before, all of this is behind an Origin Trial, so you'll have to sign up, or start Chrome with a command line flag, if you want to try it out. For more information, see below.

What's new?

First and foremost, it's now possible to control which echo canceller is being used by including a new constraint in your `getUserMedia` calls, e.g:

```
echoCancellationType: type
```

where `type` can be one of:

- `browser` to use the software implementation provided by the browser; or
- `system` to use the implementation provided by the underlying system. Currently, this is one of the implementations on macOS and on Windows.

If you leave the constraint out, Chrome will select echo canceller like it always has: if there's hardware echo cancellation, it will be used, otherwise Chrome's software echo canceller will. Without specifying the constraint, Chrome will never chose one of the two experimental echo cancellers that are part of this trial.

As `echoCancellationType` works like any other constraint, it's possible to specify `system` as an ideal value and have Chrome use it if it's available, or fall back to the `browser` one otherwise. The `browser` echoCancellationType is always available in Chrome. To figure out which echo canceller was picked, you can call `getSettings()` on the getUserMedia audio track and check the value of the `echoCancellationType` field.

Finally, you can check what echo cancellers are available for a `MediaStreamTrack` by calling `getCapabilities()` on it. However, `echoCancellationType` is not yet implemented for

`InputDeviceInfo`.

Windows echo cancellation support

We've expanded the native echo canceller support to include Windows using the Voice Capture DSP component. As with the macOS echo canceller, we want to evaluate its performance, and see if there are cases where it performs better than our software solution, if only for being placed closer to the audio hardware. Contrary to the case with macOS, our initial testing on Windows hasn't been very promising. We will continue to tweak the implementation to see if we can get it to perform better. For now, it's probably best to avoid experimenting with the Windows echo canceller on any larger scale. Try it out in controlled settings, such as on your local machine, but don't expect it to work flawlessly!

Improved macOS echo cancellation support

During the previous experiment, the macOS implementation lacked the ability to correctly track which output device was being used. This meant it would be unable to cancel echo from any device that wasn't the computer's default device. In many cases, this might not have been a problem, since macOS can automatically switch default devices when headsets, etc. are plugged or unplugged. It wouldn't work correctly in all cases, though.

This functionality has been added to Chrome M68 and is implemented both for the macOS and Windows echo canceller. Chrome's software echo canceller has not been affected by this lack of functionality, as it uses an internal loopback to get the playout audio to cancel.

How to enable the experiment

To get this new behavior on your site, your need to be signed up for the "Experimental support for native AEC" Origin Trial. If you just want to try it out locally, the experiment can be enabled on the command line:

```
chrome --enable-blink-features=ExperimentalHardwareEchoCancellation
```

Passing this flag on the command line makes the new `echoCancellationType` constraint globally available in Chrome for the current session. Using this constraint, you can then test the native echo cancellers in your app, as described above. This is the same command line flag as in the previous trial; on Chrome M68 it will enable the new functionality. Enabling the new origin trial will only activate the new functionality – it will not trigger the previous trial in older versions of Chrome.

Filing feedback

As with the previous experiment, we're interested in the qualitative performance of the macOS and Windows echo cancellers; primarily the former. We would also like feedback on how well the new `echoCancellationType` constraint works in practice, how easy it is to use, etc. This includes its inclusion in `getSettings` and `getCapabilities`.

We're also interested in how Chrome interacts with other applications when using these native echo cancellers, as well as any stability issues or other problems with the implementation.

If you're trying this out, please file your feedback in this bug ↗. If possible, include what hardware was used (OS version, hardware model, microphone / headset / etc.). If doing more large-scale experiments, links to comparative statistics on audio call quality are appreciated; whether objective or subjective.

Last updated July 2, 2018.