Lament of the Machine God

Thomas Rompré

The original intention behind this piece was to make a voice modulator to recreate the voice of the tech-priests of the *Adeptus Mechanicus* of the *Warhammer 40,000* universe: transhumanists zealots whose multitude of implants cause their voice to sound more like a mixture of a groaning machine with a deep, bassy human voice hidden deep within. However, i did not simply want to make a voice modulation patch, I also wanted the ability to create this effect of a choir of tech-priests singing, similar to a gregorian chant in a church, as mixes of industrial/electronic music and religious chanting are thematically appropriate for *Warhammer 40,000* I do this purely out of interest for voice modulation (as well as my previous experience with it creating and designing audio for video games and other creative works).

The process begins with some basic research on how to reproduce this kind of voice, my main resource for this being this video tutorial for audacity, although many more are available, none were with Max MSP. Therefore, the main challenge of producing this patch was reverse-engineering the audacity effects used in that tutorial inside of Max.

Luckily, a lot of them are fairly easy and typical sci-fi voice modulation techniques previously seen in class, such as ring modulation, flanging, reverb and delays, etc. It was mostly a question of finding the correct range of values and the correct order to attach them into in order to produce a desired sound. I also tried some reproducing some other effects used in those tutorials, such as double-reversing audio, but there was a point where it would dilute the audio to the point where it would no longer be recognizable as a voice, defeating the point of voice modulation.

Where we somewhat deviate from simply porting audacity effects to Max is by attempting to create the illusion of a choir: where several different individual “sing” the audio input but with a human-like error margin that reinforces the illusion of a singing group, such as voices having slightly different pitch, not being perfectly synced up, and so forth. We produce this effect by creating several different signals out of the main signals - already run through ring modulation - and then applying a delay effect,

which increases exponentially from split signal to split signal. We then can manually adjust the gain of each of these signals in order to make them more or less audible, in the case of a choir, we usually want the “main” signal to be center stage while the other voices are more distant, like backup singers. For example, by way of reducing gain decreasingly from signal to signal, we get the illusion of more scale and distance. We then apply a reverb effect, mostly consisting of comb filters and a lowpass filter with a modulable *size* value to determine how strong that reverb will be. After which we apply flanging to each signal, where we split the signals into two and shift the pitch of both to slightly different values, then layer them into the output to play, creating a sort of otherworldly effect to the voice. To create the illusion of dissonant, different voices, the pitch of each of these signals can be manually adjusted.

What was particularly satisfying about this patch to me however was not only its ability to produce a sound that was very similar to what I had in mind when I set out to make this project, but also its ability to create something drastically different with some slight variable changes. In lieu of a unified, echoing choir, we can have dissonant maddening voices bouncing around with shorter amounts of reverb, higher gain, and greater pitch shift ranges for the flange, just to name an example.

I see the value in this software and how control over audio processing at its lowest level can grant much greater control over audio design and production as well as create completely emergent types of audios. It is, however, a tool to be used almost solely by those extremely well-versed in acoustics and audiophonics. It lacks the simplicity, intuitiveness and compatibility that the layman or someone less interested in the lower-end of things requires to create. Its disregard for typical conventions of node-based programming can also be very counter-intuitive. While I could have very easily created a similar effect, even better with a software such as audacity, I do recognize it’s shortcomings and the benefits of having used Max for this project, it’s ability to take in live audio, make modifications on the fly and modify effects and processing at a lower level made this project possible and this sort of emergent experimental sound that this patch ends up

becoming capable of creating would've been impossible in a typical audio editing software.

It couldn’t hurt them to hire a UX designer though.