ByteNoise

Additive synthesis

Additive synthesis, also known as Fourier synthesis, is a method of making sounds (usually pleasant sounding timbres to be used in music) using sine waves as building blocks. It is based on the work of Jean Baptiste Joseph Fourier.

Some brief theory

Pretty much any timbre that sounds melodic is made up of sine waves of related frequencies. The fundamental harmonic is the main one, which is simply the note you want to play. The next one is twice as fast and called the second harmonic. The next is three times the speed and called the third harmonic, and so on. By combining sine waves whose frequencies (speeds) are multiples of the fundamental one, but whose amplitudes (loudness) differ, you can make pretty much any melodic sound you can think of.

How to make some waveforms common in audio synthesis using just a handful of sine waves

To create a sawtooth wave, include all integer harmonics. The amplitude of each harmonic N should be 1 / N the amplitude of the fundamental harmonic. In other words, the fundamental harmonic should be set to full blast, the second harmonic set to half that volume, the third to a third of its volume, and so on, with as many harmonics included as possible.

To create a square wave, include only odd harmonics (the frequency should be F(2N?1) where F is the fundamental harmonic). The amplitude of each harmonic N should be 1 / N the amplitude of the fundamental frequency. Put simply, ignore the second, fourth, sixth harmonic and so on, and keep the rest as they were.

To create a triangle wave, again include only odd harmonics (so the frequency should still be F(2N?1) where F is the fundamental harmonic). The amplitude of each harmonic N should be 1 / N^2 the amplitude of the fundamental frequency.

Python code for the above

In these examples, the code is in a for loop where x is the number of the harmonic (x should count up from 1 to however many sine waves you want to generate; the more you choose, the more accurate the results, but the slower the program). The loop should be fed the variable *time* as a number between 0 and 1, which tells it how far along one cycle of the waveform it should be, and should return *amplitude* as a number between -1 and 1 for the position the speaker should be set to.

Sawtooth wave: `amplitude = amplitude + math.sin(x * 2 * math.pi * time) / x`

Square wave: `amplitude = amplitude + math.sin((x * 2 - 1) * 2 * math.pi * time) / (x * 2 - 1)`

Triangle wave: `amplitude = amplitude + math.sin((x * 2 - 1) * 2 * math.pi * time) / math.pow(2, (x * 2 - 1))`

Devices that use additive synthesis

- The pipe organ: Each pipe produces something very similar to a sine wave.
- The Telharmonium (1897)
- The Hammond drawbar organ (1934): This used metal wheels - called tonewheels - with precisely shaped teeth and magnets to produce the harmonics; the drawbars were used to control the amplitude of each harmonic.
- The Kawai K-5 (1987) and Kawai K-5000 (1996)

Going beyond organ sounds

The reason that organs don't sound very natural is that natural sounds decay over time. More specifically, different frequencies decay at different rates (and higher frequencies usually fade away before lower ones). So to make more interesting sounds using Fourier synthesis, you should wire up each sine wave oscillator to an attenuator to make it more quiet, which in turn could be controlled by an LFO or an envelope to make each sine wave's amplitude fade in and out independently from the others. That is the key to the true power of Fourier synthesis, which allows you to produce evolving sounds rather than mere repeating waveforms. How you do this is up to you. Use your imagination and the sky's the limit!

References

- http://www.soundonsound.com/sos/jun00/articles/synthsec.htm