hyperSTACK

Polyphonic Chorus-based Software Subtractive Synthesizer

User Manual

Introduction

UAA hyperSTACK is a polyphonic chorus-based software VST subtractive synthesizer. It is able to create a rich chorus effect by stacking up numerous oscillators, each with slightly different pitch, tone, and development. These oscillators are perceived as one single instrument with a “fat” sound, much like a choir or an ensemble of strings.

The main features of hyperSTACK include:

- Up to 64 polyphonic audio oscillators, available in fast low-quality mode or alias-free high-quality mode using the BLIT algorithm.
- Sawtooth or Square waveforms.
- Complete pitch vibrato engine, which is run individually for each oscillator;
    - ❖ Individual fast sine-estimation pitch modulation LFO for each oscillator.
    - ❖ LFOs controllable by ADSR envelope, to simulate violin vibrato coming in at the end of the note.
    - ❖ Variation from pitch modulation LFO using cosine interpolated Perlin noise functions, also controlled by the LFO ADSR envelope.
- Chorus detune feature, which automatically evenly detunes every oscillator.
- 8-voice MIDI polyphony.
- Volume ADSR envelope, individual for each oscillator. The individual envelopes can be modulated via the Deviation knobs, which control how much each parameter evenly deviates from the mean.
- Resonating 24dB Butterworth Low-Pass filter, controllable by an ADSR envelope.

Installation

hyperSTACK requires a 32-bit or 64-bit Microsoft® Windows® PC, with a VST-compatible host DAW. However, this VST plug-in is a *MASSIVE* CPU-muncher, so installing on at least a Pentium 4 2.4 GHz with 512MB RAM is recommended, even for basic functionality.

To install, unzip and/or place the HyperStack.dll somewhere into your VST Plugins folder. By default, this is:

```
C:\Program Files\VSTPlugins\
```

Your particular DAW software may require a manual re-scan or refresh of new plugins in order to update its plugins list.

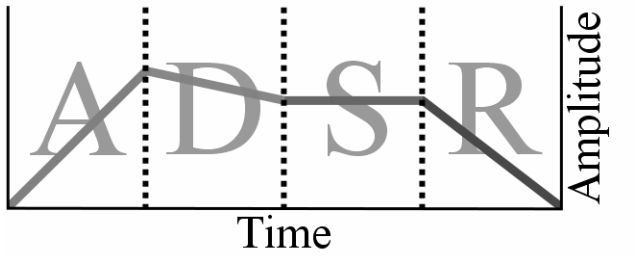
Tested working DAW hosts:

- Image-Line FL Studio 9.6 (native VST)
- Image-Line FL Studio 6.0 (native VST)
- Steinberg Cubase 5.0 (native VST)
- Steinberg Cubase Studio 4.0 (native VST)
- Avid/Digidesign Pro Tools 8 LE (FXPansion VST-RTAS Adapter 2.0)

Subtractive Synthesis

hyperSTACK is a subtractive synthesizer. These types of synthesizers usually start with a sound very rich in harmonics, such as a sawtooth wave (every harmonic up to infinity) or a square wave (every odd harmonic up to infinity), then subtract harmonics from the sound signal via a low-pass filter to achieve a more realistic simulation of an instrument.

The volume of the sound and the low-pass filter cutoff value is usually modulated by ADSR envelopes. ADSR stands for Attack, Decay, Sustain, Release.



Source: http://en.wikipedia.org/wiki/Synthesizer

While this achieves a reasonable result, the there is usually more stages to improve the “naturalness” of quality of the sound. The usual additional effects that the sound goes through are: LFO modulation, amp distortion, equalization, reverb, chorus/flanger. One will find this integrated into the synthesizer on many modern commercial hardware/software synthesizers.

LFO stands for Low-Frequency Oscillators, and modulate the sound in someway by a small amount over time to increase the natural-ness of the sound.

Amp distortion adds distorted higher harmonics lost from the filter. It is used to achieve a “dirtier” sound, similar to that of an overdriven electric guitar.

Equalization allows further control on the different harmonics of the sound, by filtering the sound in the frequency spectrum. This can be used to simulate the resonating formants of many instruments; violins, cellos, string instruments, flutes, even organs and vocals.

Reverb adds a sense of space to the sound. This can be used to make the sound less “dry”. It also can be used to fatten up the sound and give the sound more color.

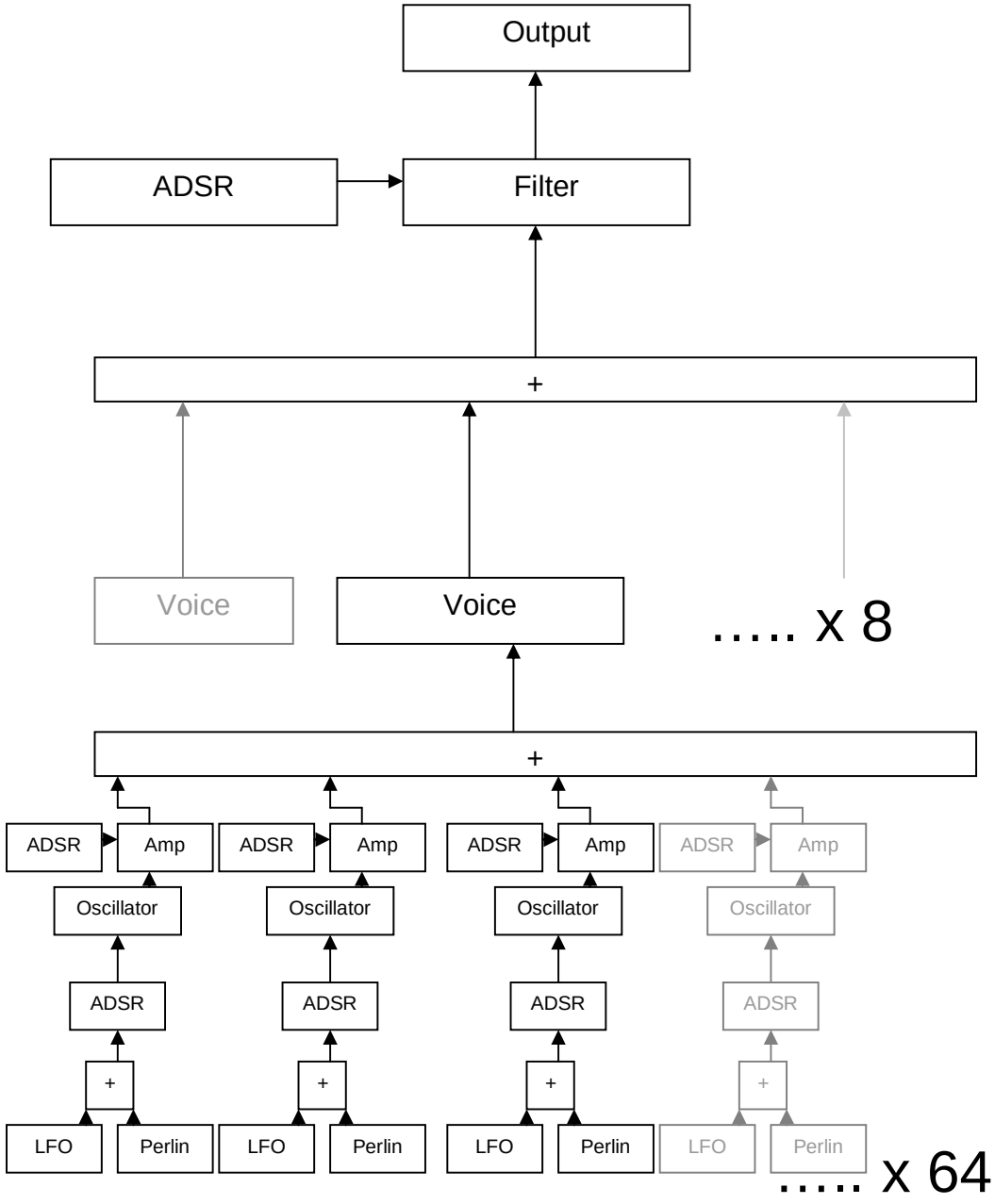
Chorus/Flanger is used to make it sound like there is many of the instrument playing at once. Since hyperSTACK uses individually detuned and modulated oscillators stacked up to achieve this effect in real, there is little need for electronic DSP effect simulation.

Since, except LFO, nearly every single DAW on the market comes with all those effects, these effects have not been programmed and integrated into hyperSTACK itself. This is because one can simply use the ones their DAW provides, or their own versions of these effects. However, it is expected that those effects are used externally, in order to achieve the full potential subtractive synthesis, and the hyperSTACK plugin, provides.

Internal Wiring

There are 8 separate voice-generators for polyphonic signals. Each individual voice generator controls 64 separate oscillators, 64 separate LFO/Perlin noise functions, 64 separate LFO envelopes and 64 separate amplifier envelopes. The result of these oscillators is then sent through a Butterworth 24dB lowpass filter, which takes input from a filter envelope.

Diagram of internal modules:



Interface

The hyperSTACK VST interface has a slick brushed aluminum look, designed to group similar and related parameters for maximum usability. The interface provides a consistent and user friendly interface no matter the platform or the DAW host.

A no-GUI version of the plug-in is also available, which uses the default VST-GUI. The look and feel of this version will largely depend on your DAW software.

hyperSTACK does **not** grant control over each individual oscillator. The parameters for the oscillators are randomized, generated or calculated using the deviation parameters.

Settings Panel

The settings panel has the overall plug-in setting parameters, and is not intended to by automated.

- **Oscillators** – This is the number of active oscillators for each note. 1 means a single oscillator is generating sound for each MIDI note, 2 means 2 separate oscillators…etc. Goes up to 64. Oscillators are randomly phase-shifted to avoid distortion. As more oscillators are added in, the volume of each oscillator is adjusted by *1 / N*, where *N* is the number of oscillators, in an attempt to keep a constant volume. However, inevitable phase canceling of the oscillators will mean volume will decrease as more oscillators are added.
- **Quality** – The quality setting of all oscillators. LOW quality is fast, but will generate aliasing especially at higher pitched notes. HIGH quality is alias-free, but takes roughly about twice the CPU of LOW quality to render. If a lot of oscillators are going to be used, using LOW quality for realtime processing then switch to HIGH quality before rendering the track is recommended.
- **WaveShape** – Controls which waveshape signal do the oscillators generate, either SAW or SQUARE waves. Saw waves are bright and rich while square waves are bright but hollow.
- **Semi** – Semi-tone offset. Goes up from -12 (one octave down) to +12 (one octave up).
- **Fine** – Pitch detuning of ALL oscillators. This will NOT detune each individual oscillator, but will detune ALL oscillators at the same time, thus raising/lower the pitch of the overall note.

Master Controls Panel

The master controls panel includes the volume knob and the chorus knob.

The **Volume** knob adjusts the overall output volume level of the whole plug-in. It's great for off-setting the reduction in volume level due to phase-canceling as the number of oscillators increase.

The **Chorus** knob adjusts the amount of even detuning done to individual oscillators. When many oscillators are slightly detuned in respect to each other, the ear perceives them as a single fat sound. This is called the *chorus* effect, and happens naturally with a choir of people singing, or an ensemble of strings. Note that this knob is intended to detune many individual oscillators to achieve the effect, and is not to be confused with the electronic DSP (Digital Signal Processing) chorus effect. The DSP effect is an attempt to mathematically simulate the chorus effect, while hyperSTACK, by stacking many detuned oscillators, is able to achieve this effect for real.

Filter Panel

The filter panel controls the master controls to the Butterworth 24dB resonating low-pass filter. Each note/voice in hyperSTACK has only ONE filter. Filters are not kept individually for each oscillator.

The **Cutoff** knob controls the cut-off frequency of the low-pass filter. This is the frequency above which the signal reduction occurs. A low-pass filter lets lower frequencies through, and blocks higher frequencies. This is usually used, in subtractive synthesizers such as hyperSTACK, in order to reduce the harmonics of the sound to achieve a more natural tone. In general, raising this knob will make the sound “brighter” while lowering this knob will make the sound more “pure”.

The **Resonance** knob controls how much the signal at frequencies around the cutoff frequency are raised. This setting is born out of old hardware low-pass filters, which actually raised the sound signal at the cutoff frequency. Having a high resonance while adjusting the cutoff will achieve a “wah-wah” type sound. Using a single oscillator, cranking resonance up way high then modulating the cutoff knob, then passing the sound through an external guitar amp overdrive will achieve the famous Roland TB-303 sound.

Volume Envelope Panel

The Volume Envelope Panel contains the controls for the ADSR envelope that controls the amp.

Separate ADSR envelopes are kept for each individual oscillator. This can be used to simulate something like a string ensemble, where not everybody bows their string at exact the same velocity, and not everybody stops the note at the exact same time.

- **Attack** – How long does it take for the note to reach full volume. Stuff like strings typically have a slow attack, while something like a piano would have a fast attack. The **Attack Dev.** knob stands for Attack Deviation. A large deviation will mean that the oscillators all have very different attacks, much like an ensemble of amateur violin players who all start bow at completely different speeds. A small deviation means all the oscillators have very similar attacks. No deviation will mean every oscillator takes exactly the same time to reach full volume.
- **Decay** – How long does it take for the note to reach, after attack, from the full volume back down to the sustain level (see below). A piano, for example, may have no attack at all, but a long decay, as the hammer strikes the string instantaneously, and the string is left to vibrate until stopped. The **Decay Dev.** knob stands for Decay deviation, and is similar to the Attack Dev. knob (see above).
- **Sustain** – How loud is the note after decay is complete. The **Sustain Dev.** knob stands for Sustain deviation, and is similar to the Attack Dev. and Decay Dev. knobs (see above).
- **Release** – How long does it take for the note to fall off AFTER the note has been released. For example, a grand piano with sustain pedal kept on will dramatically increase the release time, as the sustain pedal keeps the note going and the strings vibrating AFTER the fingers have been lift off the piano key. With the sustain pedal off, the strings are stopped from vibrating as soon as the fingers are lift off the key, thus greatly reducing release time of the note. The **Release Dev.** knob stands for Release deviation, and is similar to the Attack Dev. and Decay Dev. knobs above.

For additional information, refer to the “Subtractive Synthesis” section of this manual.

LFO Panel

The LFO panel controls the pitch modulation Low Frequency Oscillator. Each oscillator in hyperSTACK has its own separate LFO. This is the heart and soul of hyperSTACK’s vibrato engine, and can be used to effectively simulate a string ensemble or vocal ensemble.

Integrated with the LFO system is the Perlin Noise generator. The LFO and perlin noise signal are added together and treated as one.

Perlin noise, result of work by Ken Perlin, originally used for procedural visual effects in computer graphics, is basically a smooth interpolated noise function. hyperSTACK uses a 2D noise function with cosine interpolation. Perlin noise can be effectively used for humanization of the vibrato; a real human vibrato does not perfectly vibrate to +1 and -1 around the note in a perfect sine wave oscillation like a LFO does. http://en.wikipedia.org/wiki/Perlin_noise

hyperSTACK’s LFO generator will only generate the sine waveshape.

- **LFO Gain** – This knob controls how much modulation happens. This knob controls how much the LFO and noise affects the pitch of the note. A low value will introduce a subtle vibrato in the pitch, while a high value will possibly produce weird alien pitch bending effects. Note that this knob affects BOTH the noise and the LFO functions.

- **LFO Freq** – The LFO wave frequency. A higher LFO frequency will produce very fast pitch modulation, while a very low LFO frequency will produce a note that slowly changes in pitch over time. This knob only affects the LFO generator, and does not affect the Perlin noise system.
- **LFO Freq Dev.** – The LFO wave frequency deviation. Similar in concept to Attack Dev., Decay Dev. parameters mentioned above in the Volume Envelope Panel. This parameter controls how much each oscillator's LFO deviates from the value in the LFO Freq knob above. A high LFO Frequency deviation will mean that everybody is doing vibrato at completely different speeds, while a low LFO frequency deviation will mean everybody is doing vibrato nearly in sync.
- **LFO Amount** – The "volume" of the LFO generator. Unlike the LFO Gain knob above, this knob will ONLY affect the LFO generator, and will NOT affect the Perlin noise generator. Turning this knob off while leaving the Noise amount knob on will turn off the LFO generator, and have the pitch modulation rely completely on noise. Vice versa.
- **Noise Amount** - The "volume" of the Perlin noise generator. Unlike the LFO knob above, this knob will ONLY affect the noise generator, and will NOT affect the LFO signal generator. Turning this knob off while leaving the LFO Amount knob on will turn off the noise generator, and have the pitch modulated by a pure un-noised LFO. Vice versa.
- **Noise Freq** – The Perlin noise frequency. Higher noise frequency will result in noise that change faster, while lower noise frequency will result in a smooth, gradually changing/developing noise function.

LFO Envelope Panel

The LFO Envelope Panel contains the ADSR envelope controls for the LFO. This can be used to gradually introduce the LFO/Noise.

For example, this can be used to simulate somebody playing the violin, where the beginning of the note has no vibrato, and vibrato is slowly introduced towards the end of the note, and as the violin player finishes bowing, the leftover vibration on the string simulated by the “release” in the volume envelope is not vibrato-ed.

Note that the LFO envelope affects BOTH the LFO and Perlin noise signals as if it was one combined signal.

- **LFO Attack** – The LFO/Noise attack time. This is the most useful parameter, as it can be used to introduce pitch modulation gradually at the end of the note (as mentioned above). Low (fast) attack means that the pitch modulation LFO+Noise happens almost immediately after the note, while a High (slow) attack means the pitch modulation LFO+Noise comes in slowly after the note, beginning of the note having no modulation.
- **LFO Decay** – The LFO/Noise decay time. A fast decay means LFO vibrato quickly disappears. A slow decay means that the LFO will gradually reduce.
- **LFO Sustain** – The LFO/Noise sustain level. This is the target level that the envelope decay decays to.
- **LFO Release** – The time it takes for the LFO/Noise to fade away after the note has finished.

Filter Envelope Panel

The Filter Envelope Panels controls the ADSR envelope for the low-pass filter cutoff.

Note that while every other ADSR envelope in hyperSTACK is multiplied with the signal, this envelope is ADDED to the cutoff value. This behavior seems to be popular for filter envelopes for subtractive synths, probably due to them trying to simulate the behaviour of old analogue hardware subtractive synthesizers.

- **F Attack** – The filter Attack time. This knob controls the time it takes for the filter cutoff to reach its maximum frequency. A very fast filter attack will produce a piano-type sound which starts bright immediately, while a slower attack will produce a sound that starts pure, then slowly gets more brighter as the note is held.
- **F Decay** – The filter Decay time. This knob controls the time it takes for the filter cutoff to decay to the sustain level (see below). This knob is often used with the resonance knob in the Filter Panel (see above), to produce some interesting effects (such a Roland TB-303 type sound).
- **F Sustain** – The filter Sustain level. This controls the frequency to which the envelop decays to.
- **F Release** – The filter Release level. This controls the time taken for the cutoff frequency to reach zero after the note has ended.

Presets

hyperSTACK comes with a selection of presets. Since hyperSTACK does not have an EQ module, an overdrive module or a reverberation module, these preset sounds often require additional processing and effects.

To change presets, use the preset interface provided by your DAW VST host software.

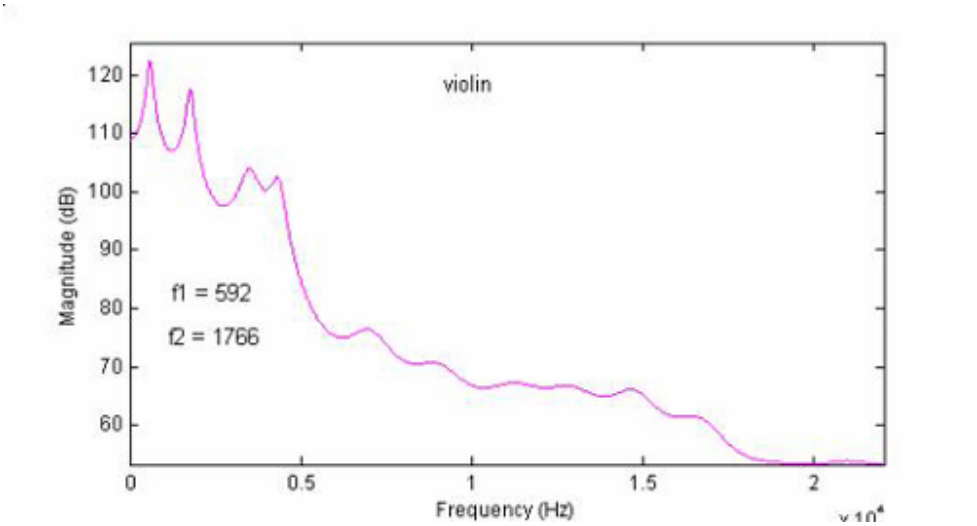
Usage & Function

The main function of hyperSTACK is to serve as a base for chorus-based and vibrato-based sounds. Examples include choirs, string ensembles, woodwind ensembles, supersaws, fat detuned dance music leads, fat basses, organs and so on.

Traditionally, such sounds require the use of the chorus electronic DSP effect in order to simulate such a sound.

hyperSTACK allows the stacking of many many oscillators, each slightly detuned and individually pitch modulated in order to realistically create the effect. This sound can then be passed through effects, formant filters, and such to create a realistic and fat sound.

Example: Violin



Source: https://ccrma.stanford.edu/~jmccarty/formant.htm

Start off with a single saw wave oscillator.

A violin only has one string vibrating at a time, so use only 1 oscillator. A violin typically has a slow attack, as it takes time for the player to bow the string. A violin has a high volume sustain, as the user can keep bowing for a continuous sound.

A violin has vibrato, done by the player shaking his left hand, so we'll need some LFO Gain. A player usually starts shaking his hand about 1 second after the note starts, so adjust the LFO Attack time accordingly. The LFO Freq should be adjusted to a natural hand shaking frequency. A hand isn't perfect, so maybe introduce some noise to the LFO.

Now adjust the cutoff value so it sounds close to a real violin, which isn't quite as bright as a raw saw wave.

As seen in the image above, a violin's first two resonating formants occur at 592 Hz and 1766 Hz. Apply an EQ to the sound, and boost those two frequencies.

Finally, add some reverb, and possibly a tiny bit of mild distortion. There we have it, a violin!

The difference between a violin made with hyperSTACK this way and a recorded violin sound sample is that this sound is slightly different for every note. Thus, when you play many notes at the same time with a violin sample, they won't quite sound right, because each recorded sample is exactly the same. However, when you play many notes with this hyperSTACK-made violin, it will sound much more natural.

Take a look at https://ccrma.stanford.edu/~jmccarty/formant.htm for many other instrument's resonating formants and frequency response graphs.

Example: Choir

Vocal formant table:
http://mitpress.mit.edu/e-books/csound/csoundmanual/Appendices/table3.html

A choir is a group of people singing. Thus, we'll start with a number of oscillators. Say, 8. Nobody sings perfectly robotically in tune, so crank up the chorus quite a bit (0.3 to 0.4) to detune everybody.

There's some natural vocal vibrato, which is similar to the violin above, but vocal vibrato quite a bit slower than a violinist's hand. The volume and LFO ADSR should be somewhat similar to the settings for violin above.

The special thing about vocals is that human vocals have VERY strong formants. The frequency absolutely spikes at those formant frequencies, and everything

else is virtually non-existent. Changing those resonating frequencies will make us perceive the vocal as different vowels and different age/gender.

The URL link above gives a table of various vocal formants. This can be implemented by applying an EQ (equalizer) or several band-pass filters. Only the first 2 or 3 formants are required for the effect to be clearly audible.

Finish it off with some reverb, and there we have it, a choir!

CPU Usage

8 Voices of 64 oscillators with individual LFOs + envelopes + Perlin noise functions = gigantic CPU hog.

On a 2.53 GHz Intel Dual Core, 8 oscillators each running LFOs at LOW quality can take about 20 – 25 % CPU. On high quality, this easily doubles to 40 – 45% CPU.

Perlin noise is a big CPU sink, so the Noise Amount knob on 0.0 if you don’t need it. Perlin Noise can add another 20 – 40% to y our CPU usage.

It is recommended to run at LOW quality for real-time purposes, and then switch to HIGH quality before the project is bounced out or rendered to file.

For more oscillators and more richer and complex sounds, hyperSTACK loses its functionality as a real-time VST. However, it is still a powerful tool for sample synthesis.

Parts of the song can also be rendered to audio, to be mixed real-time. However, these parts will have to be re-rendered if one wishes to edit something.

Automation

Consult the manual of your software DAW host to find out how to automate and/or record VST plug-in parameters.

Although every one of hyperSTACK's knob parameters can be automated, it is not a very good idea to automate the following knobs:

- Oscillators
- Quality
- WaveShape

This is because these parameters are not intended to change mid-way during the song, and can produce pops and click noises when they are changed.

License

hyperSTACK is released open-source under the MIT license:

Credits

- Xi Chen – Designer, Programmer, User Interface Artist, Manual author.
- Willy Mai – Interface utility programming
- STK Synthesis Toolkit - https://ccrma.stanford.edu/software/stk/ - Band-limited waveform class and ADSR class, and various other functions
- http://www.musicdsp.org/ - Code for low-pass Butterworth filter, fast pseudo-sine function.
- Steinberg – VST plug-in interface.
- Hugo Elias - http://freespace.virgin.net/hugo.elias/models/m_perlin.htm - Perlin noise code.
- Wikipedia – manual images.