Oliver Marlan 311271383 (SID)

Final Review: Vox Multi-FX Plug-in

Digital Audio Systems, DESC9115, Semester 1 2012

Graduate Program in Audio and Acoustics

Faculty of Architecture, Design and Planning, The University of Sydney

THE UNIVERSITY OF SYDNEY

Problem Description

I propose to develop a plug-in for DAW systems called “Multi-FX Vocal Processor”.

In modern digital music production, it is often the goal of the mix engineer or musician to create a final product that is interesting, wonderful and wild, in an increasingly competitive music industry.
This process can take many, many hours and usually requires the use of many separate audio processing hardware units or software programs, especially for vocal processing as the voice if often the most prominent and important element in the song.

I propose to develop a multi-effects vocal processor in the form of a DAW plug-in that will make it simpler and quicker for a mixing engineer, producer or musician to create interesting and wonderful vocal effects.
The plug-in will be simple and intuitive to use, but also powerful and versatile, for maximum creative output with minimal input effort.

Specification

The graphical user interface (GUI) will feature user adjustable knobs, buttons and sliders to control the effects parameters. The interface will also include clearly labelled information such as delay time/tempo, stereo width and effect mix/blend to assist the user in their creative decisions.
Each audio processing section will include an on/off switch so they can be bypassed individually if required by the user.

Vocoder
The Vocoder section will allow the user to choose the type of carrier signal to in used by this simple channel Vocoder process. Choosing from white-noise, sine-wave, square-wave will determine the ‘tone’ of the output signal.
The ‘Tone’ selection will be the only user adjustable parameter in the Vocoder section so to keep the plug-in from being cluttered and confusing.

Time Delay
The delay section will implement a simple delay effect where the user can adjust the time interval and feedback of the delay.
The time interval will be continuously adjustable from 10ms to 5s and will also feature a ‘snap to tempo’ setting so that the delay is in time with the master tempo being used within the DAW session.
The feedback control allows the user to adjust the number of times the delay will repeat, selectable from one repeat to fifty repetitions.

Chorus Effect
For the purpose of keeping the plug-in simple for any user, the Chorus section will feature only delay time and chorus depth parameter adjustment. Each parameter will be controllable via a simple slider with values changing as the slider is adjusted.

Stereo Width Control
This will be a graphical control where the user can adjust the stereo width of the final output signal, ranging from narrow mono to wide stereo.

Mix
This is the ability to control the ratio of wet to dry signal, with wet being 100% affected signal and dry being 100% original signal. This is very important, as it will increase the efficiency of the users workflow as they will not have to create any extra tracks, sends or busses within the DAW session.

Implementation

Vocoder
In this section the signal is processed with a simple channel Vocoder system, as implemented in Matlab. The input signal is analysed with an FFT in 32 discrete frequency bands and with a window size of 1024 samples. I have set window size and number of frequency bands so that they are not user adjustable, to keep the plug-in simple yet functional.
The carrier signal, as determined by the user to be white noise, sine wave or square wave, will have a length equal to that of the input signal, as implemented in Matlab using the 'length' function and 'inwave' as a variable.

Time Delay
A sample is placed in memory (RAM), stored, then recalled some time later and output. Generally, when the delay is small, the frequency response of the signal is altered; when the delay is longer, an echo results. [3]

The relationship between input and output of this system can be given by:

$$y(n) = z(n)+gs(n- M) \quad [1]$$

As implemented in Matlab, a series of delay lines are used, a specified number of zero values is added to the start of the signal so that when it is added to the original signal, we hear an echo.
At this stage, the gain of the delay is not adjusted, as would usually be the case, as gain control will happen in the 'Mix' section of this plug-in system.

Chorus Effect
The term 'chorus' refers to a type of delay effect, which is designed to make a single sound, sound as if it was being produced by multiple sources.
If several copies of the input signal are delayed in the range 10 to 25ms with small and random variations in the delay times, we will hear the *chorus* effect, which is a combination of the vibrato effect with the direct signal. [1]
The input- output relationship defining this element is given in time by:

$$\mathbf{y[n] = x[n - D[n]]} \quad [2]$$

For the Chorus section of this plug-in 'Delay Time' refers to the delay intervals, which will be selectable from 10ms to 25ms. The 'Depth' refers to the bandwidth of delay time variation, with small variations being 'shallower' and larger variations

being ‘deeper’. This translates into how prominent for obvious the chorus effect is perceived to be.

Stereo Width Control

This section of the plug-in will employ a simple ‘mid-side’ technique to widen the stereo image of the signal.

This process will split the mono input signal into two channels, one of which will be phase inverted. Listening to these two signals simultaneously, at the same volume will result in a very wide perceived stereo image, we call this the ‘side’ channel. Mixing these two signals with the original mono signal, or ‘mid’ channel, allows us to control the width of the stereo image.

When the user has the GUI control fader set all the way to ‘Stereo’, the output will be 100% Side channel, alternatively, when the user has the fader set all the way to ‘Mono’, the output will be 100% Mid channel. And having the fader control set anywhere between the two, the output will be a respective mix of Mid and Side channels.

Mix

This final stage of the plug-in system determines the relative volume levels or the original signal and the output signal.

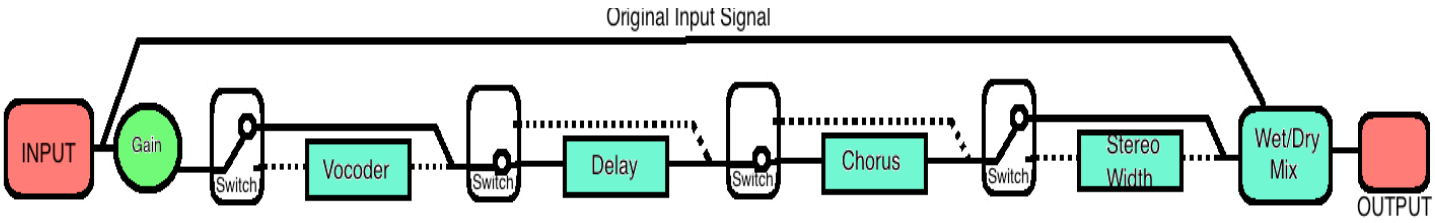
As implemented in Matlab, at this stage, the respective signals are individually multiplied by a factor between 0 and 1, before being added together to produce a single output signal. In this case, zero being 0% of original amplitude and 1 being 100% of original amplitude.

The two signals have an inverse relationship. If the user control is set to ‘100% Wet’, the processed output signal will be multiplied by 1 and the original input signal will be multiplied by 0. Similarly, if the user control is set to ‘75% Wet’, the processed output signal will be multiplied by 0.75 and the original input signal will be multiplied by 0.25.

Signal Flow Diagram

The diagram below shows the signal flow of this digital audio processor. You will notice that the original signal is split at the input stage so that it can be used later in the chain for the wet/dry mix stage.

The switch in this diagram represents the on/off switch that is available on the plug-in GUI, and allows the signal bypass each stage of process, if desired by the user. The broken line indicates no signal flow.



Graphical User Interface

The following is the proposed layout for the plug-in's GUI. It features large, easily adjustable parameter controls with clear labelling.

Evaluation

The plug-in will be evaluated in Pro Tools and compared to other effects processing plug-ins. It will be used on a variety of sound sources to evaluate its potentially diverse sound results. These sound sources include voice, electric guitar, saxophone, acoustic guitar, piano and bass guitar.

A thorough evaluation will require experienced mixing engineers use the plug-in during a mix down of a song. The engineers will be asked to compare this plug-in with other effects plug-ins that they use regularly.
For a further evaluation of its creative potential, experienced musicians will be asked to use the plug-in during their song writing process.

References

[1] *DAFX: Digital Audio Effects,* West Sussex: John Wiley & Sons, 2002.

[2] Audio Effects in Matlab, McGill University
http://www-mmsp.ece.mcgill.ca/Courses/2006-2007/ECSE490A/Experiments/Exp1/EAudioEffects.pdf

[3] *Principles of Digital Audio*, pg. 726, Ken C. Pohlman