LAB REPORT 2

Adrian Clarke 306164981(SID)

Digital Audio Systems, DESC9115, Semester 1 2012
Graduate Program in Audio and Acoustics
Faculty of Architecture, Design and Planning, The University of Sydney

VIBRATOOL.m

Applies a vibrato to an input wave by modulating the delay length with an LFO. This function has been adapted with the purpose of trying to create a more realistic natural vibrato sound, particularly with the violin in mind. This function also utilises the two functions 'proclpc' and 'synlpc'. 'Proclpc' is a function that takes an input wave and separates the source (excitation sound) from the filter (resonant structure), 'synlpc' is then used to resynthesise the two components back together at the output stage. This has been included so that when an instrument such as a violin is processed, its complex resonant structure is left unaltered by the vibrato as it would be when vibrato is performed acoustically on a violin. There are five options for modulation type specified by the input argument 'type', two of which are created by filtering noise (fractal noise and white noise) to a specified frequency, the aim of this being to create a quasi-periodic variation in frequency that is again more akin to a natural reverb. The other types of modulation are included to allow user experimentation.

Syntax

```
output=VIBRATOOL(Input,FS,Width,frequency,q,type,fade)
```

Input Arguments

`Input` = Input wave file to which vibrato is to be applied

`FS` = Sampling frequency of Input signal, generally 44100 or 48000

`Width` = Average delay length in seconds, equivalent to degree of pitch variation away from the fundamental frequency of the input signal. Recommended values for Width are between 2ms-7ms (0.002-0.007 seconds).

`frequency` = Modulation frequency, can also be considered the rate of vibrato. Recommended values for frequency are 2-10.

`q` = The width of the filter used for the two filtered noise modulating signals (only applies to case 0 and 1 of input argument 'type'). Values of 15-60 should give good results. The higher the value of q the narrower the filter used, making the modulating signal closer to the value of 'frequency'.

`type` = Type of modulating signal. There are 5 types of modulating signal, selected by the values 0-4.

0 = Filtered fractal noise

1 = Filtered white noise

2 = Sine wave

3 = Square wave

4 = Sawtooth wave

`fade` = Fade selects a fade-in for the modulating signal selected by the values 0-3. Selecting a fade length longer than the length of your input wave will not produce optimum results.

0 = No fade-in

1 = 1 second fade-in

2 = 2 second fade-in

3 = 3 second fade-in

Usage

Import the wave file to be processed using the command:

```
[Input, FS]=wavread('your_wave_file.wav');
```

You can now call the function by typing:

```
output=VIBRATOOL(Input,FS,Width,frequency,q,type,fade)
```

Remembering to change the input arguments to values outlined in the previous section. Experiment with different values to see what results you get. Your processed file will appear as a variable named 'output'. You can also use default values by leaving off input arguments as such:

```
output=VIBRATOOL(Input)
```

This will create a vibrato suited to a violin sound. You can play back your processed file with the following command:

```
Sound(output,FS);
```

You can also write the output to disk as a wave file with the following command:

```
Wavwrite(output,FS,'wavefile_name.wav');
```

Signal Flow

→ Input

→ Source/filter separation

→ Modulation signal created

→ Fade applied to modulation signal (if applicable)

→ Vibrato applied to ‘source’ of input wave

→ Modulated ‘source’ resynthesised with input wave’s ‘filter’

→ output stored as variable ‘output’

References

- Udo Zoelzer, ed. "DAFX: Digital Audio Effects". Wiley, 2002, pp. 68-69.
- Code adapted from William Martens and Densil Cabrera