Guide to DVD Chapter 1 Examples: *Richard Dobson*

Audio Processing in C:
Delay Lines and Noise Generation

Building the Examples

This guide assumes you are familiar with *make* and the general build process. See the guide document for Book Chapter 2 for a detailed explanation of these. If you have built and stored *libportsf.a* in a single standard location, modify the makefiles accordingly.

gcc (OS X, Linux, MinGW)

Type `make` in the examples directory. This will build the *portsf* library and two example programs: *tdelay* and *vdelay*. Note that the *portsf* directory contains two test programs which demonstrate/exercise the file seeking and overwrite functions: *sfreverse* and *sfrewind*. These are not built by default. See the file *readme.txt* in the *portsf* directory for more information.

Windows Users

It is recommended to use the *MinGW* environment for building programs on Windows. See the *example-guide.doc* for Book Chapter 2 for details.

The Programs

Two source soundfiles are provided: *clap.wav* and *fluteC3.wav*
Two text data files are provided: *tapdat.txt* and *tapmax.txt*

WARNING: Both example programs allow the use of feedback. This can very easily result in overloads (amplitudes increasing without limit, but clamped to digital peak). Note that the programs display the PEAK level – if this indicates 1.0, it almost always means the output has overloaded. Avoid playing back such files! They are best loaded into a waveform editor such as *Audacity* for viewing, and playing if it is appropriate to do so.

tdelay – apply tapped delay process to a soundfile.

Usage:

```
TDELAY [-gDGAIN] infile outfile tapdata.txt feedback wet/dry tailtime

  -gDGAIN:      scales feedback signal level by DGAIN. Must be > 0.0.

  tapdata.txt:  text file giving delay taps as time/amp pairs,
                separated by space or Tab. One tap per line.
                Tap times (in seconds) must be in ascending order.
                There is no upper limit to the number of taps.
                Amplitude values must be > 0.0.

  feedback:     amount of delay output returned to input.
                0 <= feedback <= 1.0.

  wet/dry:      sets level of wet (delayed) signal. 0.0 < wet <= 1.0.
                The dry (source) level is set to 1.0 - wet.

  tailtime:     Extra time (secs) added to accommodate decay tail.
```

Notes: The first tap time must be greater than zero (zero would correspond to the direct signal). Tap times need to be separated by at least the distance of one sample: each tap time is rounded to an integer number of samples. Duplicate tap times are illegal. While a feedback argument is supplied for exploration purposes, note that accumulated multiple taps of this kind invariably degrade to noise with pitched components related to the tap times.

Example: Process the file *clap.wav* to create a decay with increasing density.
(Uses *tapdat.txt*: defines four taps of varying levels.)

```
      ./tdelay clap.wav claptd.wav tapdat.txt 0.6 0.75 15
```

Example: As above, but uses *tapmax.txt*: all tap amplitudes = 1.
This example aims to create a steady output level, as density increases to (hopefully) sustained noise.

```
      ./tdelay clap.wav claptdm.wav tapmax.txt 0.36 0.5 40
```

Display this in *Audacity*, using the dB plot option, to confirm that the output is sustaining. Note how the sound gradually degrades to noise with strong pitched components. The feedback value is shown to be critical - an output that seems sustained for 40 seconds may after 100 seconds prove to be much louder!

vdelay – apply time-varying delay to input soundfile.

Usage:

```
VDELAY [-gDGAIN][-rFREQ] infile outfile delay freq mod feedback
       wet/dry tailtime


-gDGAIN :  scales feedback signal level by DGAIN. Must be > 0.0.
           may be needed with high feedback, to keep signal within range.

-rDEPTH :  vary modulation depth randomly at rate FREQ Hz.
                   0.0 < FREQ <= 100.0

delay   :  sets primary delay time in msecs. Must be > 0.0.
               Overall maximum delaytime is extended by delay * mod.

freq    :  modulation frequency (Hz). Must be > 0.0.

mod     :  percent deviation relative to delaytime. 0 <= mod <= 1.0.

feedback:  set feedback level. 0 <= feedback <= 1.0.

wet/dry :  set level of wet (delayed) signal. 0.0 < wet <= 1.0.
           The dry (source) level is set to 1.0 - wet.

tailtime:  extra time (secs) added to infile to accommodate decay tail.
```

Example: Process the file *fluteC3.wav* to create wide slow random frequency variations.

```
./vdelay -r2 fluteC3.wav flutevary.wav 200 1 0.5 0.5 1 2
```

Example: Apply vibrato (frequency modulation); uses sine modulation with no feedback.

```
./vdelay fluteC3.wav flutemod.wav 200 4.5 0.005 0 1 1
```

Example: As above, with random variation of depth.

```
./vdelay -r2 fluteC3.wav flutemodr.wav 200 4.5 0.008 0 1 1
```