Sweep: Real-world Audio Editing on Linux

Conrad Parker

Table of Contents

Introduction

Sweep [http://sweep.sourceforge.net/] is an open source editor for digital audio. The project was initiated by the author in 2000, initially by combining various portions of pre-existing open source projects, including the waveform editor from *Soundtracker*, with user interaction modelled on that of *the GIMP*. Prior to 2002, Sweep was available in alpha release; the basic functionality was already in place, including discontinuous selections, unlimited undo and redo, and support for LADSPA effects plugins. The support of an animation studio prompted an overhaul of the user interface and the tackling of harder problems related to application latency. This paper discusses that work, including the motivation for major usability improvements and new directions, including the use for live performance, that have arisen as a result.

Motivation

Usability is a critical factor in the design of an audio editor on any personal computer, because the user interface of such devices is strongly geared towards visual manipulation of data objects. When designing the user interface for Sweep, the author sought to overcome several shortcomings in usability that audio editors for personal computers have generally exhibited. Firstly the visualisation of audio data in waveform editors is often poor, providing little indication of the audio content. Secondly, although the data being edited is audio, existing software provides few opportunites to actually hear it, beyond a fixed speed playback that cannot be invoked during editing or even navigation. Lastly, the lag between audio and its visualisation introduced by buffering in the audio device is often poorly managed by editing software, introducing a disconcerting delay between user interaction and audible response.

Computer editors for text and visual media such as video allow the user to visually scan and navigate through a work, and to immediately see the outcome of any editing operation. However this is not the case for audio editors, which generally provide only a rough outline of the waveform, in the form of a graph depicting peak values. This is sufficient to discern major differences, such as that between silence and loud sounds, and to notice the effect of large edits, such as cutting. Such a graph can be used to perform simple editing operations such as *topping* and *tailing*, the removal of silence at the start and end of a recording. However such a representation is completely inadequate for depicting the operation of any more subtle operation, such as the application of a noise-reducing filter or the addition of a reverberation effect.

Whereas navigation through a text document in a visual text editor implicitly provides an indication of the content at the cursor position, it commonly takes a number of seconds simply to find the context of one's place in an audio file. Precise placement of the audio cursor often requires tedious juggling with fixed-speed playback and transport controls such as *fast-forward* and *rewind*. However, in the world of analogue audio editing, such as with tape

reels, the user experience is far more tactile. The tape can be moved at arbitrary speed back and forth past the playback head, allowing the user a detailed scan of the material being edited, and a precise search for suitable edit points such as the onset of musical pauses or the completion of syllables in speech.

The latency introduced by a non-realtime multi-tasking operating system is another crucial factor in the design of interactive audio software. Due to the requirement of fair scheduling, it is not possible for such a system to guarantee that data written to the audio device will be heard at exactly the right moment; if scheduling delays cause an audio application to be starved of access to the audio device up to the time when sound is due to be played, an audible glitch will be heard. Although very brief this sound is often extremely jarring, may cause damage to speakers and if not detected in software can cause a loss of synchronisation between the audio and video or other applications. In order to compensate for unpredictable scheduling, applications can increase the size and number of the audio driver's buffers. Larger buffers can go a long way towards ensuring that no glitches are heard, however this degrades interactivity. The size of the buffers is directly proportional to the time delay between the application writing to the audio device and the sound being heard, and for sounds triggered by interactive events this introduces a delay between user input and the expected sound. It can be quite disconcerting for the user if this delay is more than about 10 ms, especially for musical applications. For audio editors this delay manifests itself during playback as a discrepancy between the cursor position on screen and the sound heard by the user.

These shortcomings are not present when editing or navigating audio in the analogue domain, as is done with recordings on analogue tape reels or cueing of songs on vinyl records; in fact in the latter case, the responsiveness of the analogue medium is so precise that it is regularly used as a performance artform in its own right. Hence it was desired to improve Sweep's usability in these areas such that it would be comparable to editing in the analogue domain, and a suitable measure of success was deemed to be its usefulness as a tool for live performance.

Implementation

Various features have been implemented in order to improve the editing experience. This section discusses Sweep's waveform visualisation, the implementation of scrubbing including modelling the physics of a turntable for playing vinyl records, and improvements in the visual synchronisation to depict application latency.

Visualisation

Sweep 0.1 improved on the visual representation of audio data by combining a display of the waveform peak with an overlay of the average value. Together these provide the user with a notion of both the overall loudness and the dynamic shape of the sound. Additionally, a 3D bevel effect was applied to the waveform rendering, which by emphasising the differences in peak values, accentuates pitch differences at various zoom levels, providing a rough indication of sonic texture. Although not strictly providing any complex analysis, this often provides just enough extra visual texture to distinguish between simple instrumental and vocal portions of a recording. An example of Sweep's waveform rendering is shown in *Figure 1*.

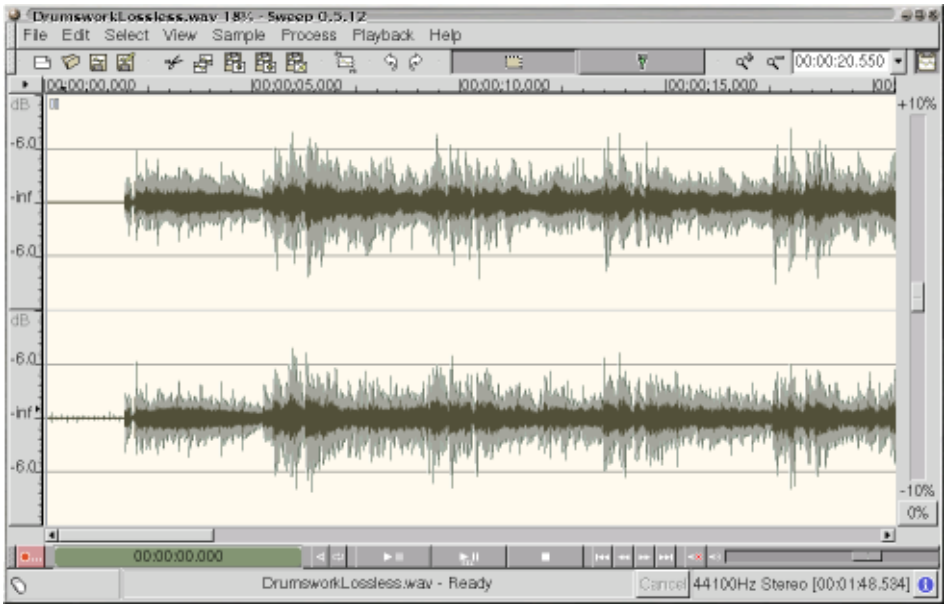

Figure 1: Screenshot of waveform view in Sweep

Scrubbing

The major addition to Sweep's usability was the implementation of interactive scrubbing. Generally speaking, scrubbing in a digital media editor allows the user to locate specific items of interest or jump directly to specific points in time by interacting with a timeline or other visual representation of time.

Sweep features a number of innovative, complementary scrubbing methods:

- A scrub tool allows the user to jump directly to specific portions of the waveform on screen, and gives immediate audio feedback. By slowly dragging the mouse cursor over the waveform, the user can interactively listen to the waveform to sample accuracy.
- There is immediate audible feedback when selecting a region or moving the edges of a region. This makes the editing task more intuitive because the user can hear the region edges while they are being selected.
- The timeline above the waveform is available as a scrubbing mechanism during playback, and otherwise allows direct placement of the cursor.
- Simply dragging the horizontal scrollbar during playback allows the user to very quickly move through the file with audio feedback.

Sweep's scrubbing was modelled on the interactivity of audio in the analogue domain, as experienced with tape reels and vinyl records. Although it is not possible to directly edit vinyl records, the records themselves are such a directly responsive format that a skilled user such as a professional disc jockey is able to use them to quickly cue and mix together songs, and for some musical genres such as hip-hop, the skilled practitioner incorporates the audible scanning of the record under finger-tip control into the music, in an artform known as *turntablism*. This advanced level of interactivity was used as a benchmark -- if a digital audio editor could be created with such direct responsiveness that it could be used artistically, it would surely provide a much needed usability boost to the more mundane task of editing.

The audible characteristics of vinyl, especially when played on the turntable of a professional disc jockey, are subtly different and inherently more pleasing than the simple fast playback of a tape reel. Three factors come into play: wear on the record groove, non-linear filtering introduced by forced motion of the stylus, and controlled momentum of the turntable under the action of a slipmat.

Firstly the "smoother" sound of vinyl is somewhat due to physical wear introduced by contact of the stylus each time a record is played, such that over time the groove is widened and high-frequency details are smoothed over. As this is a general trait of vinyl records and introduces a constant distortion of the sound, it is not desirable to include it directly when modelling the navigational properties of the turntable in a digital audio editor.

Secondly, the physics of moving a stylus quickly through the groove of a vinyl record introduces a complex filtering. The microscopic shape of a record groove is depicted in *Figure 2*, with stereo channels encoded as horizontal and sideways variations. Upon forced motion the stylus' increased momentum causes it to skip over the high-frequency details encoded in the groove. This filtering removes much of the annoying high frequency components which are introduced by the increase in playback speed. Although the actual filtering introduced by a stylus on vinyl is non-linear and would be costly to implement in software, it is usefully approximated by the application of a simple lowpass filter.

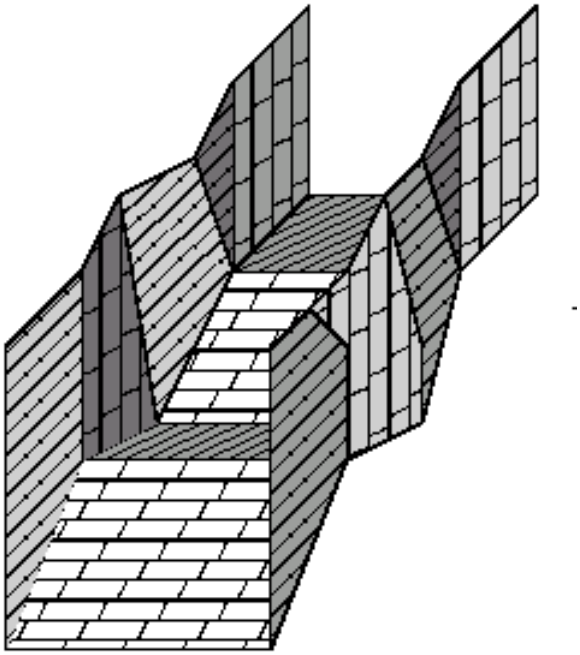

Figure 2: Cutaway diagram of vinyl groove.

Lastly, the weight of a turntable provides a fair amount of momentum, such that when a record is sped up by the disc jockey's finger, it takes some time to slow down to the drive speed of the turntable. This momentum also provides a more subtle smoothing of the record's motion, such that any sudden changes invoked by the disc jockey produce a somewhat less marked change in the record's playback. A similar amount of momentum was modelled in Sweep's scrub tool, such that if desired the cursor can be thrown back and forth along the waveform display, and such that sudden changes in direction and speed are smoothed over to provide non-jerky responsiveness.

Monitoring playback latency

Recent efforts have vastly improved the ability of the Linux kernel to schedule interactive events, including low latency work by Andrew Morton [http://www.zip.com.au/%7Eakpm/linux/schedlat.html] and Ingo Molnar [http://people.redhat.com/mingo/lowlatency-patches/], and Montavista's work on kernel preemption maintained by Robert Love [http://www.tech9.net/rml/linux/]. This work has been so effective that with a properly tuned kernel the latency introduced by audio buffering can be reduced to the vicinity of 1 ms. However this currently requires some configuration on the user's part, and is specific to Linux. It is also important to realise that the latency percieved by a user is not only introduced by the kernel, but also by the application, and it is the application's responsibility to take the total latency into account when synchronising audio with visuals. The basic configuration window for selecting the amount of device buffering requested by Sweep is shown in *Figure 3*.

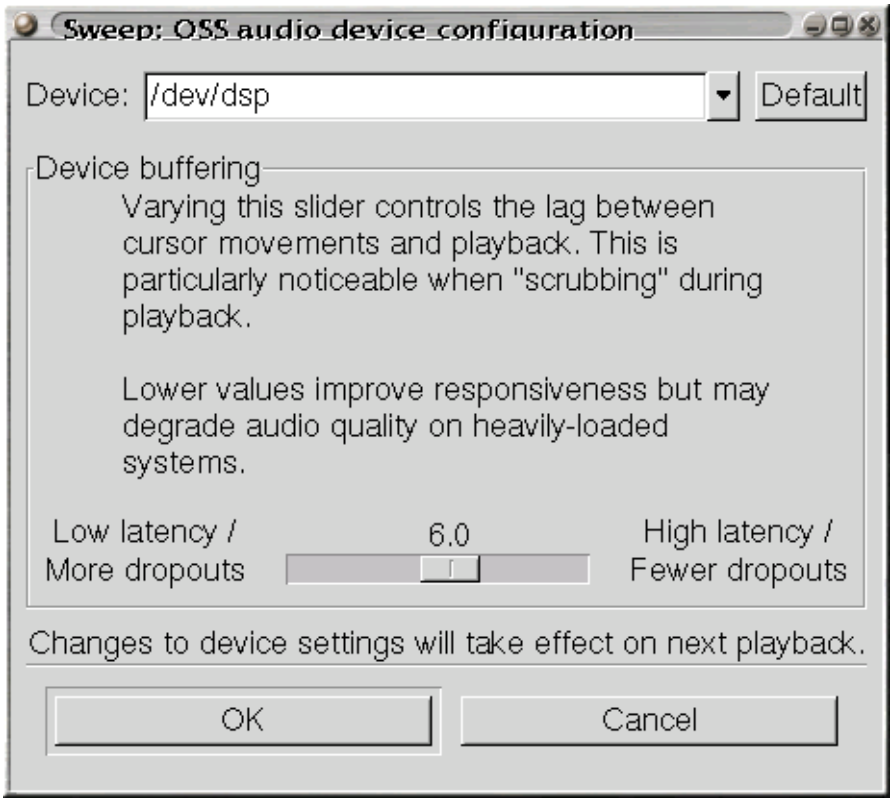

Figure 3: Sweep's device buffering configuration

For the sake of portability and acceptable behaviour when running stock kernels, it was necessary in Sweep to introduce some visual feedback of the delay caused by device buffering. During playback, Sweep displays two cursors simultaneously, as shown in *Figure 4*: the white cursor to the right is under the user's control, and can be moved by the transport controls and the scrub tool; the green cursor to the left always displays the position of the audio that can currently be heard. Hence if the user scans or scrubs through the file, the white cursor is moved immediately but the green cursor may lag slightly due to buffering in the audio device, and due to motion smoothing introduced by the modelling of momentum. Thus the user has a true representation of their influence over the playback position, and is not misled by contradictory audio and visuals. This also provides an obvious visual representation of the application latency, which is otherwise a fairly abstract concept.

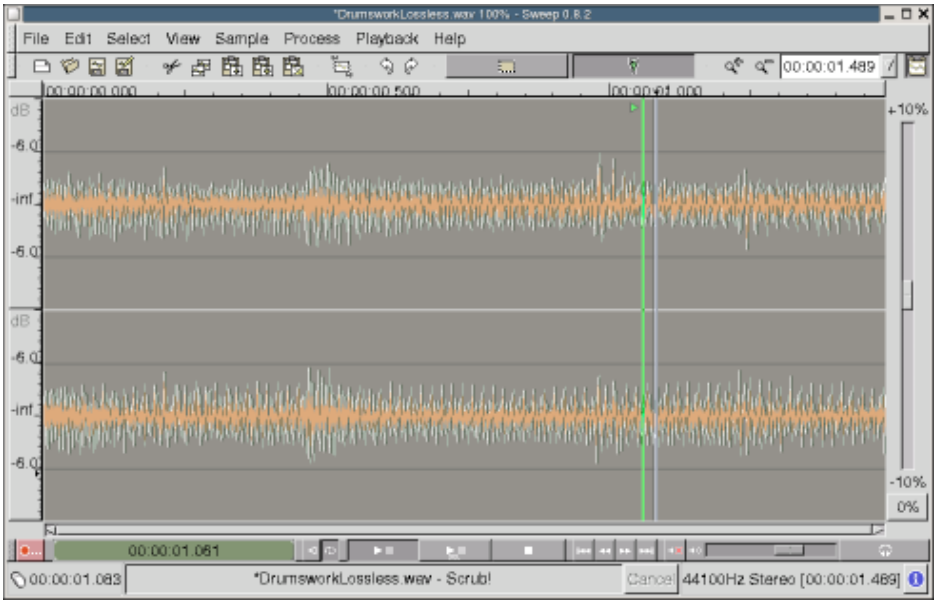

Figure 4: Sweep’s cursors: playback (left) and user (right)

Conclusion

The usability of Sweep has been vastly improved through various means, with a goal of recreating the style of interaction possible in the analogue domain. Along with Sweep’s pre-existing waveform view of peak and average data values, the implementation of vinyl-like scrubbing and accurate monitoring of playback latency has greatly improved the overall usability of the program.

A review in the March 2003 issue of *Linux Format* magazine presented Sweep as "a capable performance application that edits as well", emphasising the usefulness of the software as "a DJ’s best friend".