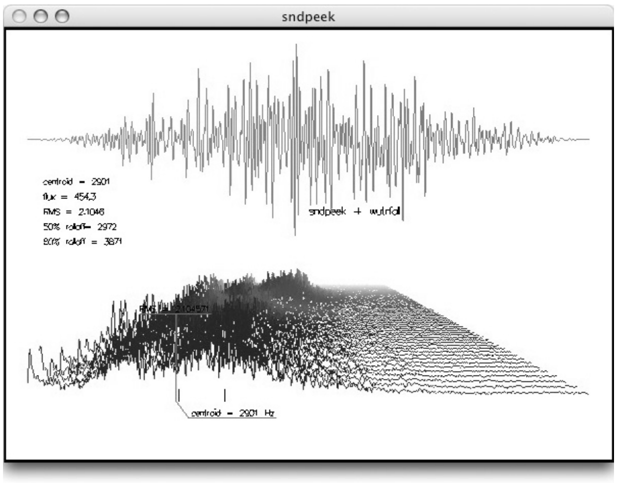
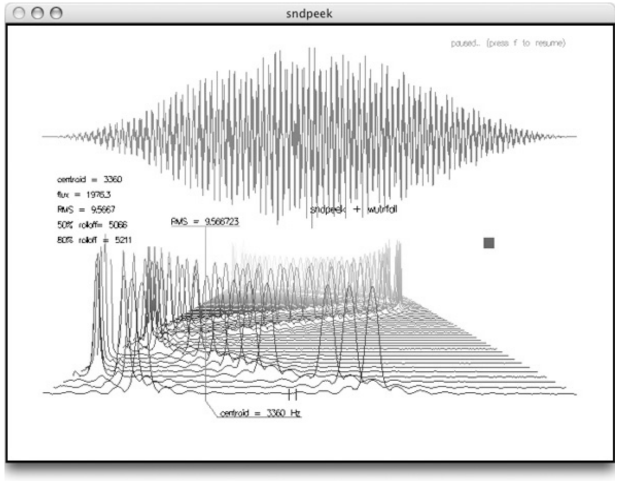
SNDTOOLS: REAL-TIME AUDIO DSP AND 3D VISUALIZATION

Ananya Misra   *Ge Wang*   *Perry R. Cook*[†]
Princeton University
Department of Computer Science ([†]also Music)





ABSTRACT

We present **sndtools**, a set of cross platform, open-source tools for simultaneously displaying related audio and visual information in real-time. The distribution includes tools to extract spectral information, perform linear predictive coding analysis and resynthesis, manipulate pitch and time using a phase vocoder, and map text to Morse code. Each tool has closely related audio and visual (graphical or text) components and can be used for instructive purposes or experimentation with sound. We show that hardware-accelerated graphics tools such as OpenGL can be used to enable real-time 3D visualization of DSP algorithms.

1. MOTIVATION

Advances in computer graphics rendering have made it possible to render large amounts of information at high speed and low cost. The graphics environment OpenGL, for example, offers powerful hardware-accelerated computation, with common functions, such as rotation, scaling and lighting, optimized to take up few resources. In addition, its widespread support across platforms allows OpenGL programmers to take advantage of standard library routines. The programming interface also provides flexibility to design customized widgets and displays. This suggests that both the power and the flexibility of OpenGL can be leveraged to create cross-platform real-time sound visualization displays.

Existing audio visualization tools are often not real-time. Those that are [5, 9], tend not to use hardware-accelerated graphics tools, which limits the complexity of audio and visual computations they can perform. They are also often designed for a specific platform and are difficult to port.

The **sndtools** package presents several real-time audio-visual displays that take advantage of efficient graphics tools to devote more processing power to audio analysis and synthesis. Thus, complex operations such as spectral feature extraction, linear predictive coding and phase vocoder-based pitch and time manipulations can be performed and visualized in real-time. As a side effect of this processing freedom, the package also provides a reusable library of analysis and synthesis tools. All the programs in the **sndtools** distribution are open source and run on most popular platforms, including MacOS X, Windows and Linux. (Visit URL provided at end of paper for source code and executables.) Thus, **sndtools** subscribes to the notion of free sound, where ”free” can mean both ”no-cost” and, more generally, ”accessible”. We hope this accessibility encourages users to experiment with the tools provided.

In the rest of this paper, we describe the components of the **sndtools** distribution. **sndpeek** is a waveform and spectrum visualizer with several other features. **rt_lpc** performs linear predictive coding to analyze and synthesize sound and displays the results visually as well as via audio output. **rt_pvc** is a phase vocoder that also has real-time display. **morse** translates text to Morse code and provides a text visualization of the result, synchronized with

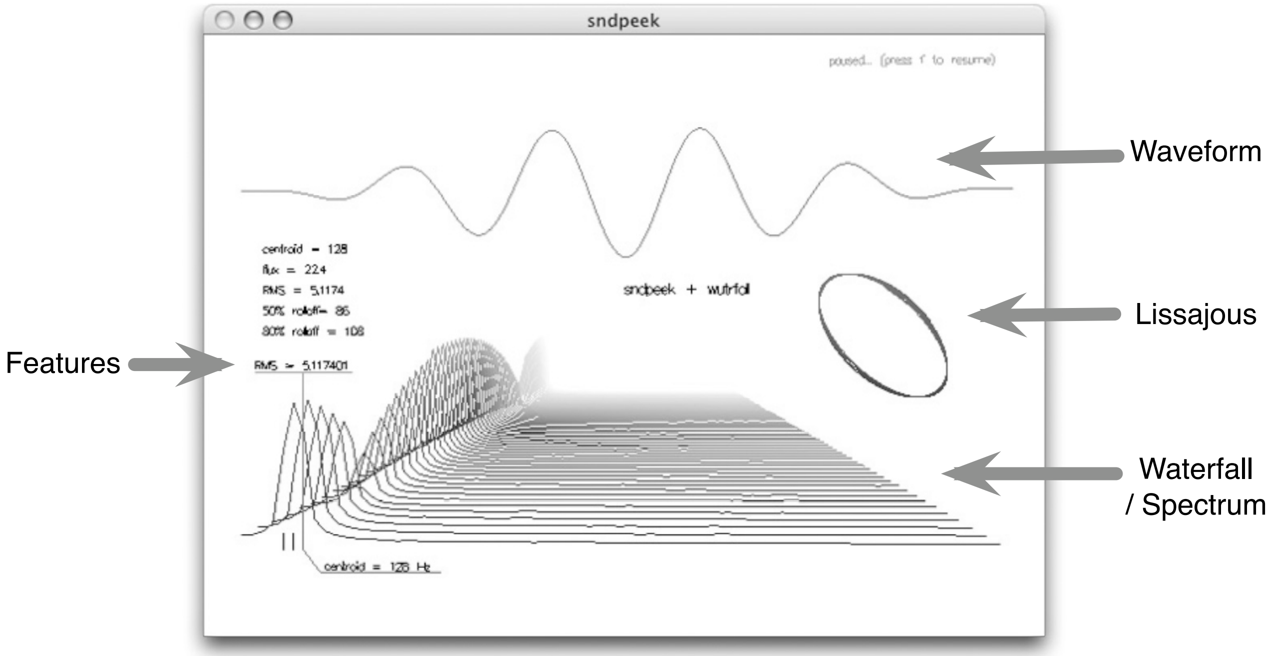


Figure 1. **sndpeek** in action.

audio. Although it does not rely on high-level graphics tools, it resembles the other components in computing and simultaneously presenting related visual and audio information. We then conclude and describe possibilities for future work.

2. SNDPEEK

sndpeek performs spectral analysis on audio input and displays the information graphically. Figure 1 shows a screen shot of **sndpeek** in action.

In essence, **sndpeek** provides real-time 3D visualization of components including:

- *Time-domain waveform*, which can be input from a microphone or from a .wav, .aiff, .snd, .raw, or .mat file with optional playback.
- *FFT magnitude spectrum* scaled by a weighted square root function so that peaks are easier to decipher than with linear or dB scaling
- *3D waterfall plot* - cascading FFT magnitude spectra where previous frames fade into the background
- *Lissajous plot* showing correlation between the left and right channels (stereo signals), or *phase delay similarity plot* showing the signal versus a delayed version of itself (for mono signals)
- *Spectral features* - features such as centroid, rms, rolloff and flux are extracted using the MARSYAS framework [10] and displayed in real-time

The display itself can be manipulated in several ways. Options include scaling and rotating the display to facilitate zooming in on information from different viewpoints, and the ability to freeze the display on a single frame for closer observation. The spacing between spectra in the waterfall plot can also be modified, and the spectrum view can be toggled between dB and the default scaling.

sndpeek can also be adapted for use in other systems. For example, it is an essential part of the Audicle [11].

3. RT_LPC

rt_lpc performs real-time linear predictive coding (LPC) [1, 4, 6, 7] analysis and resynthesis on audio input. The input sound is analyzed to obtain a number of filter coefficients and an error signal, as well as a pitch estimation. The coefficients are then used to synthesize and play back in real-time a signal close to the original sound. The controls available can be used for pedagogical purposes to show how different variables affect the LPC analysis and synthesis, or for fast trial-and-error to obtain the optimal settings for some desired output.

The visual components of the display include:

- *Original waveform* - the waveform of the input sound
- *Predicted waveform* - the waveform predicted by the LPC coefficients
- *Error waveform* - difference between the original and the predicted waveforms
- *Vocal tract shape visualization* obtained from the LPC coefficients using Durbin Recursion [3].
- *Pitch* - whether the sound is pitched or unpitched
- *FFT magnitude spectrum and waterfall plot* and other features from **sndpeek**

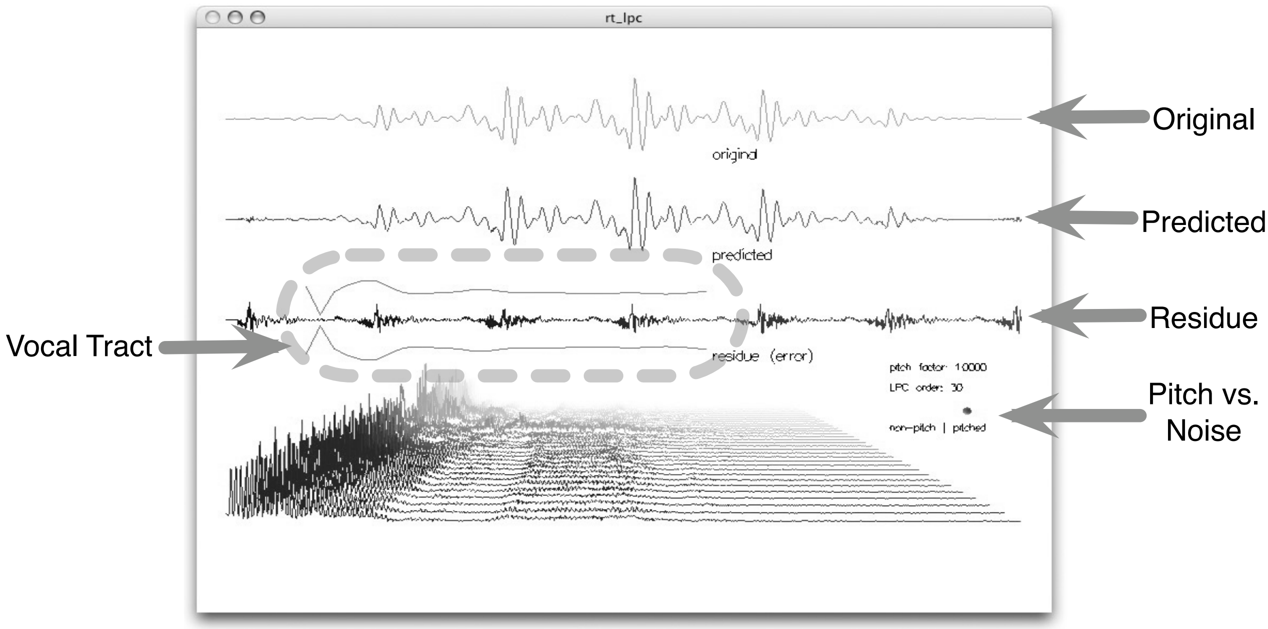


Figure 2. **rt_lpc**: real-time LPC analysis and resynthesis.

- *Information* on the current values of adjustable parameters. These include:
  – *LPC order* - the number of coefficients analyzed
  – *Pitch shift factor* - for pitched sounds, the pitch can be modified by some factor before synthesis

In addition, the pitch pulse source can be selected to provide an impulse or to model a glottal pulse. Optional emphasis and deemphasis filters can also be applied before and after the LPC is performed.

Apart from real-time LPC visualization, **rt_lpc** also offers a modular LPC library for use in other applications. The library consists of an LPC data structure, *analyze* and *synthesize* functions, and several helper units.

4. RT_PVC

rt_pvc is a real-time phase vocoder [2, 8] that can be used to time-stretch and pitch-shift sound while minimizing artifacts arising from phase incoherence. It computes the FFT spectrum for each windowed frame of input audio, and carries out pitch-shifting and time-stretching by manipulating consecutive spectra. It then performs phase adjustment to account for discrepancies caused by moving the bins and frames in the previous step.

Since the phase vocoder is real-time, time stretching and compression can cause changes in delay in the playback. The tool keeps track of this delay, allowing the synthesis to fall behind or catch up with the live input.

The visual display includes:

- *Original waveform* - the waveform of the input sound
- *FFT magnitude spectrum and waterfall plot* similar to the **sndpeek** display

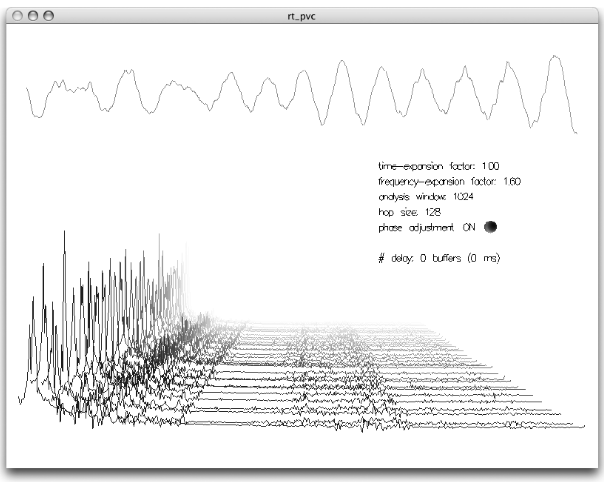


Figure 3. **rt_pvc**: real-time phase vocoder.

- *Information* on the current parameter values, including:
  – *Time-expansion factor* - adjustable amount by which the signal is lengthened (or shrunk) in time
  – *Frequency-expansion factor* - adjustable factor for changing frequency or pitch
  – *Analysis window size* - the length of sound analyzed as one frame
  – *Hop size* - adjustable spacing between the beginnings of consecutive analysis windows
  – *Phase adjustment* - the option to adjust phase or not, for demonstrating the difference made by phase adjustment
  – *Buffer delay* - the delay accumulated so far due to time-stretching

Like **rt_lpc**, **rt_pvc** can be used for both pedagogical purposes and for testing the effects of changing parameters in real-time. Moreover, it also offers a modular phase vocoder library for use elsewhere. The library contains *analyze* and *synthesize* functions as well as functions for phase fixing and unwrapping, frequency shifting, overlap add (for time-stretching), and cross synthesis. It also includes helper functions for windowing and for handling multiple buffers.

5. MORSE

morse, a barely nontrivial but fun command-line utility, reads in text and outputs the Morse code version of it as ASCII dots and dashes. These are mapped to short and long beeps, which are played in real-time by synchronizing the printing of each symbol with the playing of the associated sound.

Here is an example of **morse** at work.

```
> morse
Type characters (case insensitive)
free sound
FREE SOUND
..-. .-. . .   ... --- ..- -. -..
```

In this example, the user begins the **morse** program and types in ”free sound”. The program reprints the input in Roman characters and in Morse code. The Morse code version is accompanied by the corresponding sounds. While the association with sound can make learning Morse code easier and more interesting, the program has also been used in computer music performances.

6. CONCLUSIONS

Taking advantage of widely available computer graphics tools allows us to efficiently render visual displays, freeing more resources for audio analysis, manipulation and synthesis. This allows real-time visualization and audio processing to take place simultaneously, even for relatively complex analysis and synthesis operations. Built on this premise, **sndtools** offers real-time audio and visual display of common sound manipulation tools such as spectral analysis, LPC and phase vocoders. It also provides libraries of these tools, as well as other mini-tools that can be used for instruction, composition or just for fun.

Plans for future work involve integrating more tools into the distribution. New tools can include both real-time 3D visualizations of other common audio analysis and synthesis algorithms, as well as interesting smaller-scale units. Our final goal is to have a comprehensive set of open-source, cross-platform, real-time sound tools.

7. ACKNOWLEDGEMENTS

We thank Ahmed Abdallah and Paul Botelho for co-authoring **rt_pvc**, and Matthew Hoffman for code and ideas on **rt_lpc**.

SndTools is open-source and freely available at:

`http://sndtools.cs.princeton.edu/`

8. REFERENCES

[1] Atal, B. ”Speech Analysis and Synthesis by Linear Prediction of the Speech Wave”, *Journal of the Acoustical Society of America 47.65(Abstract)*, 1970.

[2] Dolson, M. ”The Phase Vocoder: A Tutorial”, *Computer Music Journal 10(4):14-27*, 1986.

[3] Durbin, J. ”The Fitting of Time-Series Models”, *Rev. Inst. Int. Stat. 28(3):233-243*, 1960.

[4] Lansky, P. and K. Steiglitz. ”Synthesis of Timbral Families by Warped Linear Prediction”, *Computer Music Journal 5(3):45-49*, 1981.

[5] Lauer, C. and N. Reithinger. ”A Lightweight Configurable Signal Analysis Toolset for Multiple Applications”, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

[6] Markel, J. and A. Gray. *Linear Prediction of Speech*. Springer, New York, 1976.

[7] Moorer, J. ”The Use of Linear Prediction of Speech in Computer Music Applications”, *Journal of the Audio Engineering Society 27(3):134-140*, 1979.

[8] Moorer, J. ”The Use of the Phase Vocoder in Computer Music Applications”, *Journal of the Audio Engineering Society 26(1/2):42-45*, 1978.

[9] ”Scopes and Realtime Visualizers”, From http://www.linux-sound.org/scopes.html, Retrieved March 5, 2005.

[10] Tzanetakis, G., and P. R. Cook. ”MARSYAS: A Framework for Audio Analysis”, *Organised Sound 4(3)*, Cambridge University Press, 2000.

[11] Wang, G. and P. R. Cook. ”The Audicle: a Context-sensitive, On-the-fly Audio Programming Environ/mentality”, *Proceedings of the International Computer Music Conference (ICMC)*, Miami, USA, 2004.