excerpt from Techniques chapter of doctoral thesis " The use of real-time interactive music systems in music composition and performance", Z. Settel 2001 U. of Montreal

EVENT DETECTION

Event detection is a technique, which is used to identify features in an audio signal (presumably from a musical source). The features can be physical, such as signal intensity, psychoacoustic, such as instrument pitch, or musical, as in the case of a "sharp attack". The features can be continuous sample streams (e.g. time-varying amplitude) or discrete events, as with the start of a note. When these features are recognized, the event detector outputs corresponding events. Event detection provides a way to obtain control signals based on (or analogous to) the playing of live performers. I often use these signals to control sound generation and sound processing (though rarely for synchronization of the electronics to the live playing, e.g. score following). In my works, event detection, or tracking, often provides the main control sources for the electronics. The principal techniques for event detection used in my work are described below.

Envelope tracking

Even small latencies (10 ms or 20 ms) in an event detection system can be noticeable when, for example, the derived control signals are mapped on to sound generators (e.g. sluggish triggering of percussion sounds based on the player's sharp attacks). Worse, the latencies are also "felt" by the performer who triggers the sluggish action, thereby encumbering the player's performance. Thus, for control mappings to audio where there is a tight temporal coupling of performer and resulting audio, it is necessary to use low-latency event detectors. This rules out pitch followers, which by nature require at least a period and a half of captured waveform before they are comfortable about pronouncing their guess as to what the pitch is. This also rules out many frequency-domain analysis-based techniques[1] since they need to collect signal (512 samples of it equals 11 ms at 44.1 kHz sampling rate) before analyzing and processing the data. But time-domain techniques such as amplitude envelope following require little, or no "state" (buffered samples). The envelope follower's control signal output responds rapidly and reliably to even the smallest changes in the input signal. The resulting control signal is potentially rich in nuance (dynamic amplitude resolution) and sensitivity, while being virtually instantaneous (extremely low-latency, i.e. 2 ms or less) in terms of responsiveness.

---

[1] these techniques use the Fast Fourier Transform (FFT) to convert time-domain signals into frequency-domain (complex) signals.

Below is a view of the envelope follower. For flexibility and economy, it has two outputs: one control rate, and one audio rate. The control output is used for subsequent analysis (feature recognition), while the audio output is most often used for direct gating purposes.

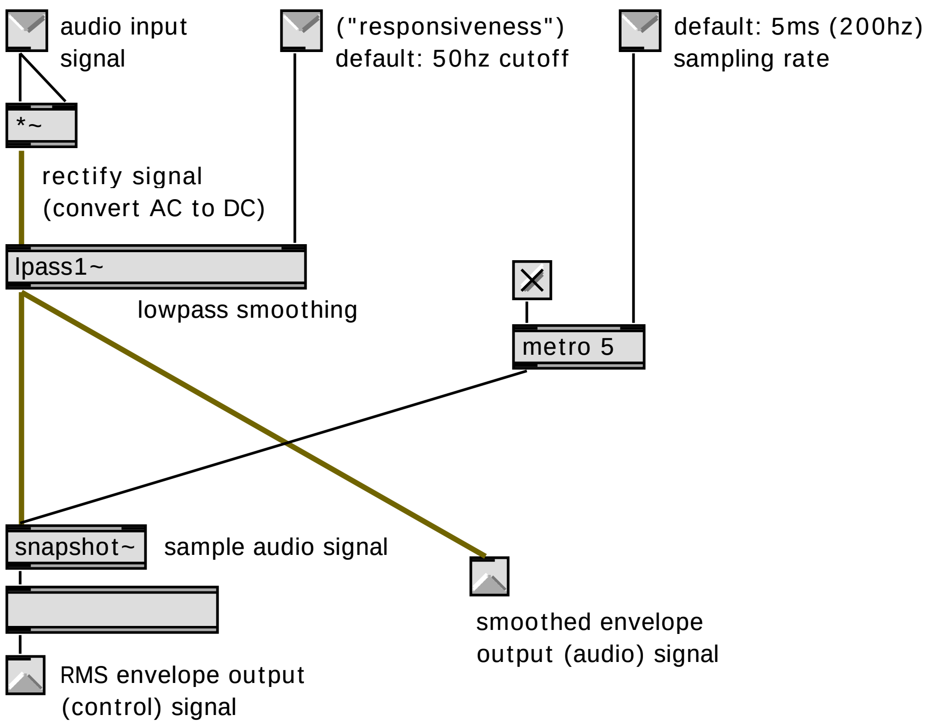

Figure 1: View of the basic amplitude follower

With the RMS (root mean squared) envelope control information, it is possible to recognize additional features in the audio input signal (performer's playing). The following features are recognized and produce corresponding control information:

- "silences" (rest event)
- large positive amplitude changes (spike event)
- RMS amplitude (amplitude value)
- smoothed audio envelope (envelope signal)

A rest (relative silence) can generate a very useful control event, often used for resetting a sound generator, or checking registers, etc. The 'rest' event is generated as soon as the value of the RMS envelope drops below a certain threshold.

The 'spike' event is triggered when the positive change of the RMS envelope, between successive samples (5 ms interval) is greater than a certain amount. Described another way:

a'= (a[t] – a[t-n]). If a' > thresh, then bang

This provides for detection of attacks in the audio input signal (the performer's playing), or *subito* dynamic level increases (say, from *mp* to *ff*).

The 'amplitude' event provides an instantaneous amplitude value, which is often consulted when an attack is reported and the current amplitude value is needed to generate new parameters for a given sound generator or processor.

The 'envelope signal' is used for gating since it's more economical to do gating directly from the envelope follower's audio output than to do so using the RMS amplitude values. Plus the envelope signal is squared (see above, no square root), and sounds better to me when used as a gating function.

Zero-cross tracking

This technique is a very simple: it's based on a counter that is incremented each time the amplitude value of its input signal goes positive or negative (i.e. crosses the value 0). Below is a 10 ms plot of a noise sample showing how often a noisy signal crosses the value of 0.

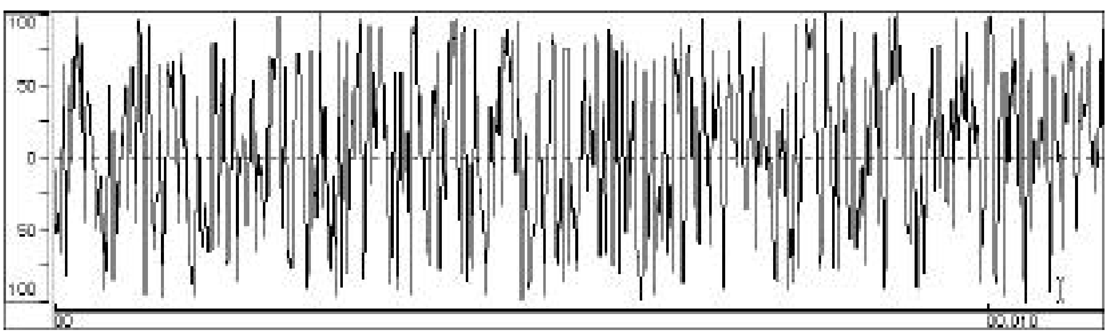

Figure 2: 10 millisecond noise waveform

The zero-crossing counter is examined and then reset to zero periodically (say every 10 ms). If at any time the examined value of the counter exceeds a certain threshold, a 'zero' event is triggered:

- zerocrossing density (zero event)

In practice, using the ‘zerocross’ object[2], the ‘zero’ event signals a high zero crossing density in the input waveform. Since noisy signals tend to cross zero often, the ‘zero’ event can be a reliable detector of the presence of noise in the input signal. In speech, it makes for an excellent detector of the “v”, “f” or “z” sibilant (or hissing) quality of sound.

Spectrum tracking

Though an interesting and potentially useful technique for detecting changes of spectral brightness (the spectral energy’s center of mass), my work with the spectral centroid[3] has not yet been fully integrated in my compositions. The robustness of this technique is limited by the fact that the quality of brightness of an acoustic instrument is often a function of both register and loudness. Using a pitch follower in conjunction with the “centroid” object[4], it is possible to decouple the two, but the responsiveness of the brightness tracking suffers from the pitch-tracker’s latency, and its sometimes unstable (presence of inharmonic energy) output. The only control output by the centroid object is:

- spectral center of mass (frequency value)

Pitch tracking

I have developed a pitch following interface around the ‘pt’ object[5]. The ‘pt’ object provides two kinds of pitch information: event-driven MIDI pitch events, and polled (every 7 ms.) instantaneous frequency. The control output is:

- note event (MIDI pitch value)
- instantaneous frequency (frequency value)

For the most part, the MIDI events are not used, since the envelope follower does a much faster and more stable job of recognizing note onsets (except for legato passages, in which case the MIDI events are relied on). Unlike the MIDI events, with 7-bit integer pitch resolution from 0 to 127, the resolution of polled frequency values is 32-bits. When a note onset is detected by the envelope follower and its pitch needs to be known, the instantaneous frequency value at that moment is used.

---

[2] ISPW/MSP object written by Francois Dechelle, IRCAM, 1991.
[3] spectral center of mass as a measure of brightness; D. Wessel, D. Bristow & Z. Settel, “Control of Phrasing and Articulation in Synthesis”, 1987.
[4] MSP object written by Ted Apel of UCSD, 1998.
[5] pitch tracker written by Miller Puckette, IRCAM, 1990

The detection interface

My event detection interface allows for the adjustment of the responsiveness (sensitivity) of the various event detectors to live performance situations for each performer.

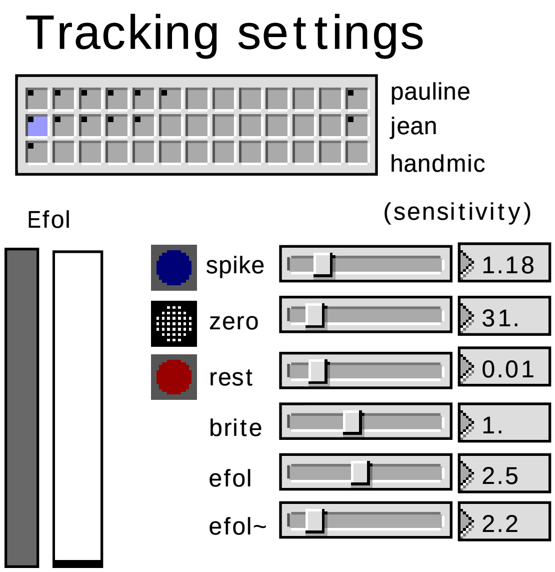

Figure 3: Event tracking interface