GRAFTING SYNTHESIS PATCHES ONTO LIVE MUSICAL INSTRUMENTS

Miller Puckette
University of California, San Diego

ABSTRACT

A new class of techniques is explored for controlling MUSIC N or Max/MSP/pd instruments directly from the sound of a monophonic instrument (or separately acquired inputs from a polyphonic instrument). The instantaneous phase and magnitude of the input signal are used in place of phasors and envelope generators. Several such instruments are described; this technique is used in SMECK, the author's open-source guitar processing patch.

1. INTRODUCTION

One of several possible approaches to real-time expressive control of computer music algorithms is to use the acoustical sound of a live instrument directly as a source of control. This has the advantage of tapping into the skill of a trained musician (rather than requiring the player to learn entirely new skills). Further, using the sound of the instrument is sometimes preferable to using physical captors; the captor approach is more expensive, harder to migrate from one instrument to another, and downright unavailable for some instruments such as the human voice. Also, arguably at least, the sound of the instrument itself is what the musician is focussed on producing, so it is in a sense more direct to use the instrumental sound than data taken from captors.

We will be concerned with the direct manipulation, at the sample level, of almost-periodic signals. These may come from a monophonic pitched instrument or from a polyphonic string instrument with a separated pickup. Rather than analyzing the signal to derive pitch and amplitude envelopes for controlling a synthesizer, we will work directly from the signal itself, extracting its instantaneous phase and magnitude, for direct use in a signal processing chain.

2. PRIOR WORK

Much instrument-derived synthesis takes an analysis/synthesis approach, in which time-windowed segments of incoming audio are analyzed for their pitch (such as in guitar synthesizers) and/or spectrum (as in Charles Dodge's *Speech Songs*).

Algorithms that can act directly on the signals without the intermediary of an analysis step can attain higher robustness and more expressivity (because they can avoid the latency of analysis). These have used the incoming audio signal simultaneously as a processing input and to compute the parameters of the audio effect such as a delay line [2], or an all-pass filter or frequency shifter [3]. In these approaches, audio signal processing (at a low latency) is parametrized by (higher-latency) analysis results: for instance, filtering or modulating instrumental sound depending on its measured pitch [5]. The approach adopted here [7] is similar to, but generalizes, the frequency shifting approach.

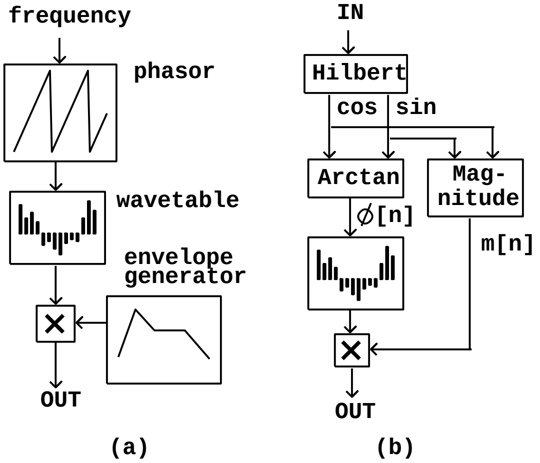

Figure 1. A MUSIC N instrument: (a) classically implemented; (b) adapted to use an incoming sound's phase and amplitude

3. GENERAL TECHNIQUE

Our strategy is to extract time-dependent phase and amplitude signals from a quasi-periodic waveform, which can then be plugged into a wide variety of classical or novel MUSIC-style instruments. For example, the simple instrument shown in Figure 1 (part a), an amplitude-controlled wavetable oscillator, is transformed into the instrument of part (b). Whereas the instrument of part (a) has frequency, start time, and duration as parameters, that of part (b) has an audio signal as input coming from an external instrumental source. The phase and amplitude are computed from the incoming audio signal.

To extract the phase and magnitude, the input signal is first converted into a pair of signals in phase quadrature us-

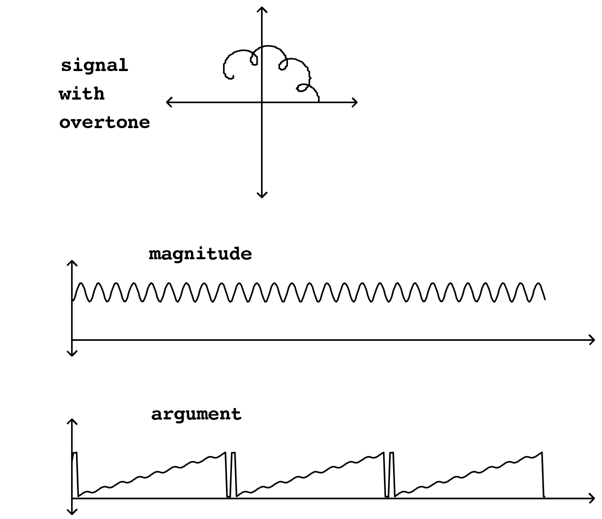

Figure 2. Extracted "magnitude" and "phase" of a sum of two sinusoids.

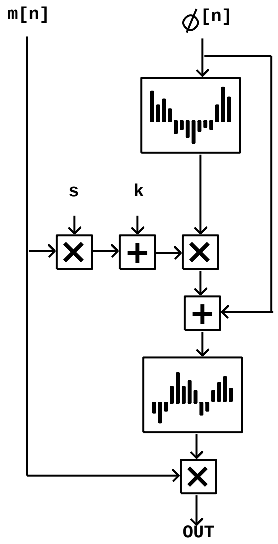

Figure 3. Phase modulation. The parameter k is a fixed modulation index, and s is the sensitivity of modulation index to input amplitude.

ing an appropriately designed pair of all-pass filters, labeled "Hilbert" in the figure. Denoting these by $x[n]$ and $y[n]$, the magnitude and phase are computed as

$$m[n] = \sqrt{x^2[n] + y^2[n]}$$

$$\phi[n] = \arctan\left(y[n]/x[n]\right)$$

If the incoming signal is a sinusoid of constant magnitude m and frequency f, then the calculation of $m[n]$ retrieves the value m and ϕ is a phase signal (a sawtooth) of frequency f. If the input signal contains more than one sinusoidal component, the result is more complicated, but if one sinusoidal component is dominant then the effect of the others is simultaneously to modulate the signals $m[n]$ and $\phi[n]$; i.e., they are heard as amplitude and phase modulation [7].

This is illustrated in Figure 2, which shows the phase and magnitude measured from a synthetic input consisting of first and ninth partials with amplitudes in the ratio four to one. At top the two phase-quadrature outputs of the "Hilbert" filters are graphed on the horizontal and vertical axes. As shown underneath, the measured magnitude and phase are, to first order, those of the fundamental, modulated sinusoidally at eight times the fundamental. (Although the phase signal appears to have jumps, it is continuous provided phases zero and 2π are taken as equivalent).

The particular way a patch responds to changes in amplitude in the input signal can be specified at will. For example, the patch of Figure 1(b) respects the amplitude of the incoming signal in two ways. First, the amplitude of the output follows the amplitude of the input, since the table lookup portion depends only on phase and the amplitude of the input is used linearly to control the amplitude of the output. Second, scaling the input by a constant factor does not affect the waveform but only scales the output correspondingly. Both of these properties are sometimes desirable but other behaviors are possible and often desirable.

As a special case, if the wavetable in Figure 1(b) is chosen to be a single cosine cycle, the instrument recovers an all-pass-filtered version of the incoming sound:

$$u[n] = m(n)\cos(\phi[n]) = x[n].$$

It is thus possible to move continuously and coherently between the nearly unaltered sound of the original instrumental signal and highly modified ones.

4. PHASE MODULATION

As a first example of a useful type of audio modification, the instrument of Figure 3 adds phase modulation to the sound of the incoming instrument. There are two parameters, a fixed index k, and a sensitivity s to allow the index to vary with amplitude. To analyze this we first put $s = 0$ so that the index is fixed at k. If the incoming sound is a sinusoid, then the amplitude signal is constant and the phase signal is a regular sawtooth, so the result is classic 2-operator phase modulation [1] with index k. If we now allow s to be nonzero, the modulation index becomes amplitude-dependent.

If the input is a near but not exact sinusoid, deviations from sinusoidality will appear as phase modulation on both

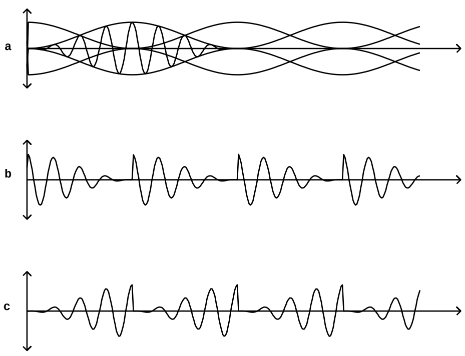

Figure 4. Formant generation: (a) a format as an overlapped sum of wave packets; (b, c) decomposition of the sum into realizable parts.

the carrier and modulation oscillators, and also as amplitude modulation on the output and index of modulation, and, if $s \neq 0$, on the index of modulation of the 2-operator system. In any case, the amplitude of the output still follows that of the input faithfully.

5. WAVE PACKETS

One powerful approach to synthesizing pitched sounds is formant synthesis as in VOSIM [10], CHANT [9], PAF [8, Ch. 6], and so on. Our approach here is to use wave packet synthesis (although the PAF also works well in this context). Figure 4 shows a wave packet series with an overlap factor of two.

Each packet is the product of a Hann window with a sinusoid; each sinusoid's phase is zero at the center of its packet. The fundamental frequency is controlled by the spacing of the packets, the formant center frequency is the frequency of the individual "packetted" sinusoids, and the bandwidth is increased by decreasing the duration of the packets so that the overlap factor falls below two.

Packets last up to two periods of the incoming signal, so they are not computable as single-valued functions of the phase. While it would theoretically be possible to construct individual packets using a two-to-one unwrapping of the phase, this would lead to occasional unwrapping errors for real-world incoming signals. A better approach is to separately synthesize series of rising and falling half-packets as shown in Figure 4, parts (b) and (c).

Figure 5 shows a block diagram that realizes a wave packet series. The two inputs $\phi[n]$ and $m[n]$ are the extracted phase and magnitude of the incoming signal. Control values k and h (which may also vary in time) control the formant central frequency and the formant bandwidth, each as a multiple of the frequency of the incoming signal.

The wave packet formulation of formant synthesis also allows for the use of more complex waveforms in place of the sinusoids, permitting us to generate periodic sounds with arbitrary, time-varying spectral envelopes [6]. To do this, we replace the wraparound tables in the block diagram of Figure 6 with a two-dimensional table, whose rows each act as waveforms to packetize. Collectively, the rows describe a time-varying waveform, whose variation is controlled by a separate control parameter (shown as an envelope generator in the figure).

An interesting application is computer-determined scat singing, in which instrumental notes are shaped into scat syllables, chosen to reflect the timing and melodic contour of the instrumental melody.

6. RESULTS

The testing ground for these techniques has been a guitar with a separated pickup (the Roland GK-3). Most of the available Roland gear only provides MIDI or synthesized outputs, but for this work the six raw string signals are needed; they may be obtained via the KeithMcmillin Stringport [4]. The six guitar strings are each treated separately according to the algorithms described here. The system has been used by the author in many performances starting at Off-ICMC 2005. A comprehensive patch with many presets is available both in Max/MSP through the KeithMcmillin website, and in Pd from `crca.ucsd.edu/˜msp`.

7. CONCLUSION

This paper has described an exceedingly general technique. Almost any classical MUSIC N-style computer music instrument may be adapted so that one or more of its phase and amplitude controls are replaced by ones obtained from an incoming quasiperiodic instrumental input. Two examples were shown here (FM and wave packet synthesis using phase bashing). Other techniques included in the published patches but not described here include have been antialiased "classic waveforms" (sawtooth, square, etc), and waveshaping using both Chebychev polynomials and Gaussian or Cauchy pulse shapers; these can be combined with classical processing techniques including filtering and frequency shifting.

The electric guitar is well adapted to this approach since it is an expressive musical instrument that is naturally played through a loudspeaker (the instrument's own physical sound does not compete with the processed output.) Other electric stringed instruments also work well (as long as a separated pickup is used); close-miced voice can also be made to work in practice.

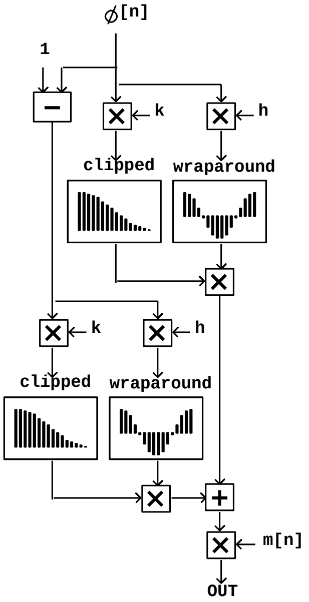

Figure 5. Block diagram for computing series of wave packets as shown in Figure 4.

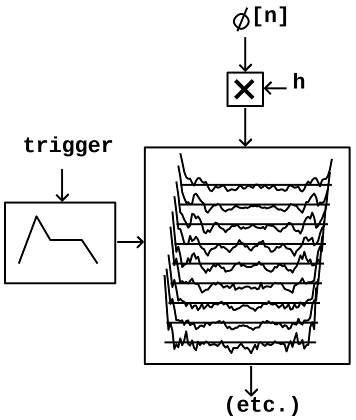

Figure 6. Complex-waveform wave packets controlled by the measured phase (horizontal axis) and a synthetic parameter ("front-to-back" axis).

8. REFERENCES

[1] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *Journal of the Audio Engineering Society*, vol. 21, no. 7, pp. 526–534, 1973.

[2] V. Lazzarini, J. Timoney, and T. Lysaght, "The generation of natural-synthetic spectra by means of adaptive frequency modulation," *Computer Music Journal*, vol. 32, no. 2, pp. 9–22, 2008.

[3] V. Lazzarini, J. Timoney, J. Pekonen, and Vesalimaki, "Adaptive phase distortion synthesis," in *Proceedings of the International Conference on Digital Audio Effects*. Hamburg: www.dafx.de, 2009, pp. 28–35.

[4] K. McMillen, "Computer input device for polyphonic stringed instruments," in *Proceedings of the International Computer Music Conference*. Ann Arbor: International Computer Music Association, 2008.

[5] C. Poepel and R. Dannenberg, "Audio signal driven sound synthesis," in *Proceedings of the International Computer Music Conference*. Ann Arbor: International Computer Music Association, 2005, pp. 391–394.

[6] M. S. Puckette, "Phase bashing for sample-based formant synthesis," in *Proceedings of the International Computer Music Conference*. Ann Arbor: International Computer Music Association, 2005, pp. 733–736.

[7] ——, "Patch for guitar," in *On-line proceedings, Pd Convention*, 2007. [Online]. Available: http://pure-data.ca/en/

[8] ——, *The Theory and Technique of Electronic Music*. Singapore: World Scientific Press, 2007. [Online]. Available: crca.ucsd.edu/~msp/techniques/

[9] X. Rodet, "The chant project: from the synthesis of the singing voice to synthesis in general," *Computer Music Journal*, vol. 8, no. 3, pp. 15–31, 1984.

[10] S. Templaars, "The vosim signal spectrum," *Interface*, vol. 6, pp. 81–96, 1977.