Alias-Free Digital Synthesis of Classic Analog Waveforms

Tim Stilson (stilti@ccrma.stanford.edu)
Julius Smith (jos@ccrma.stanford.edu)
CCRMA (http://www-ccrma.stanford.edu/)
Music Department, Stanford University

Abstract

Techniques are presented for alias-free digital synthesis of classical analog synthesizer waveforms such as pulse train and sawtooth waves. Bandlimited impulse trains are generated as a superposition of windowed sinc functions. Bandlimited pulse and triangle waveforms are obtained by integrating the difference of two out-of-phase bandlimited impulse trains. Variations for efficient implementation are discussed.

1 Introduction

Any analog signal with a discontinuity in the waveform (such as pulse train or sawtooth) or in the waveform slope (such as triangle wave) must be *bandlimited* to less than half the sampling rate before sampling to obtain a corresponding discrete-time signal. Simple methods of generating these waveforms digitally contain *aliasing* due to having to round off the discontinuity time to the nearest available sampling instant. The signals primarily addressed here are the impulse train, rectangular pulse, and sawtooth waveforms. Because the latter two signals can be derived from the first by integration, only the algorithm for the impulse train is developed in detail.

2 Related Techniques

Additive synthesis (see [Roads 1996] for a summary of this and other synthesis techniques discussed in this section) can be trivially bandlimited simply by not generating harmonics higher than $F_s/2$. A bandlimited impulse train at fundamental frequency f_1 can be generated using additive synthesis by summing $N = \lfloor (F_s/2)/f_1 \rfloor$ cosine oscillators.

Periodic wavetable synthesis [Mathews 1969] (not to be confused with sample playback synthesis which is also called wavetable synthesis these days) can be made free of aliasing by use of bandlimited interpolation when accessing the wavetable [Smith and Gossett 1984]. In this case, the wavetable contains a 1 followed by all zeros (an impulse).

Discrete-Summation Formulae (DSF) [Moorer 1975] can be used to synthesis a bandlimited impulse train algorithmically based on a closed-form expression for a sum of cosines. The Systems Concepts Digital Synthesizer implemented this method in hardware, and it is used in CSound's buzz and gbuzz unit generators. A disadvantage of DSF relative to the previous two is that, when calculating harmonics all the way to $F_s/2$, the number of harmonics changes with frequency, which causes the highest harmonic to "pop" in or out as the pitch frequency glides down or up. The previous methods (and the method we will discuss) can allow the harmoncs to die out (or come in) slowly and imperceptibly.

Formant synthesis techniques, such as VOSIM [Kaegi and and S. Tempelaars 1978], Chant [Rodet *et al.* 1989], and linear prediction [Roads 1996] can be modeled as an impulse train driving a formant filter where the timing of the impulses is rounded to the nearest sampling instant. This impulse-time rounding causing pitch-period jitter which is a form of aliasing. Eliminating this jitter in Chant or VOSIM requires resampling the filter impulse response each period which would be very expensive. Using a bandlimited impulse train as described here to drive a formant filter will eliminate this pitch-period jitter.

3 Bandlimited Impulse Train (BLIT) Synthesis

The standard operation before sampling is to apply an anti-aliasing filter. The ideal anti-aliasing filter has a continuous-time impulse response that is a sinc function with a zero-crossing interval of one sample:

$$h_s(t) \triangleq \mathrm{sinc}(F_s t) \triangleq \frac{\sin(\pi F_s t)}{\pi F_s t}.$$

The ideal unit-amplitude impulse train with period T_1 seconds is given by

$$x(t) = \sum_{l=-\infty}^{\infty} \delta(t + lT_1)$$

Applying the anti-aliasing filter h_s to this signal and sampling at the sampling rate $F_s = 1/T_s$ gives

$$y(n) = \sum_{l=-\infty}^{\infty} \mathrm{sinc}(n + lP)$$

where $P = T_1/T_s$ is the period in samples (not normally an integer). The above expression can be interpreted as a *time aliasing* of the sinc function about an interval of P samples, and it can be shown to be given by

$$y(n) = (M/P)\mathrm{Sinc}_M[(M/P)n] \qquad (1)$$

where

$$\mathrm{Sinc}_M(x) \triangleq \frac{\sin(\pi x)}{M \sin(\pi x/M)}$$

This function provides a closed-form expression for the sampled bandlimited impulse train (BLIT), and it can be used directly for synthesis in a manner similar to DSF. While P is the period in samples, M is the number of harmonics. It is always odd because an impulse train has one "harmonic" at DC, and an even number of non-zero harmonics, provided no harmonic is allowed at exactly half the sampling rate (which we enforce). Note that M/P is always close to 1. When P is an odd integer, $P = M$, and $y(n)$ is simply $\mathrm{Sinc}_M(n)$. As P departs from M, Eq. (1) implements a time scaling along with a compensating amplitude scaling. We can relate the number of harmonics M to the period P of the impulse train as

$$M = 2\lfloor P/2 \rfloor + 1$$

i.e., M is the nearest odd integer to the period P in samples.

Sum of Windowed Sincs (BLIT-SWS)

A more efficient method for synthesizing digital impulse trains may be based on the windowed-sinc method for general bandlimited interpolation [Smith and Gossett 1984]. The technique is equivalent conceptually to bandlimited periodic wavetable synthesis of an impulse train, as mentioned in the previous section: Bandlimited interpolatation is to convert the sampling rate of a discrete-time unit sample pulse train from a pitch which divides the sampling rate to the desired pitch. The rate conversion causes each unit sample pulse $\delta(n)$ to be replaced by a windowed sinc function $w(t)h_s(t)$ sampled at some phase which generally varies each period.

Because the windowing imposes a finite fall-off rate in the harmonics, some aliasing is inevitable. We can, however, control this by our choice of window. It is also helpful to helpful to have a small oversampling factor so that there is a good sized guard band between the upper limit of human hearing and half the sampling rate. This reduces the window length required.

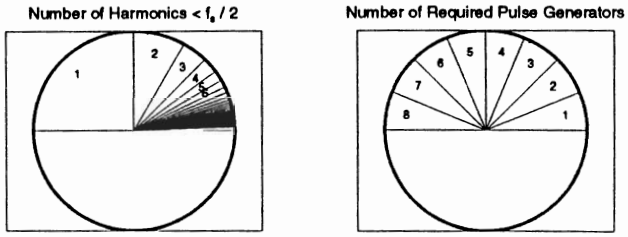

Figure 1: *Comparing number of harmonics to number of overlapped pulse instances for a pulse 8 samples long.*

A further optimization comes from comparing the number of sincs that must be overlapped in the BLIT-SWS method to the number of harmonics of the BLIT that land below $F_s/2$. At very high frequencies, the number of bandlimited harmonics becomes quite small. Indeed, in the top octave, only the fundamental is in band. Thus for a large percentage of the frequency range, it is quite likely that it may be more efficient to generate a BLIT (or any other harmonic waveform) by simple summation of sines. At lower frequencies, we can again revert to the BLIT-SWS method because it is obviously more efficient at low frequencies, where the number of harmonics is very large.

Like additive synthesis and bandlimited wavetable synthesis, and unlike DSF, in BLIT-SWS synthesis the highest harmonic need not audibly "pop" in or out as it comes down from or gets up to half the sampling rate, since the window function can be chosen to exhibit any harmonic falloff rate.

Example Spectrum

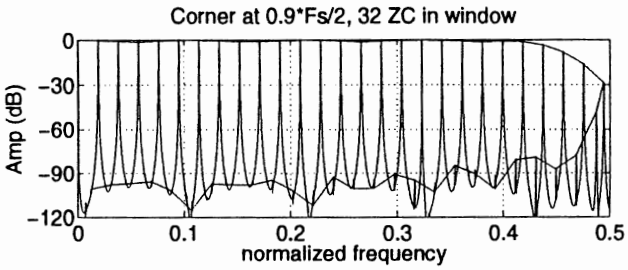

Figure 2: *Spectrum of BLIT-SWS impulse train with a line drawn through the harmonic peaks, both in-band and aliased.*

Figure 2 shows the spectrum of a bandlimited impulse train generated using the BLIT-SWS method with 32 sinc zero-crossings under a Blackman window. Note that, as is done in bandlimited interpolation, the cut-off frequency of the windowed sinc function is lowered below half the sampling rate so that its transition band is folded in half as it falls

into half the sampling rate and reflects. Only the upper 10% of the spectrum is heavily aliased. If the limit of human hearing is 20 kHz, this means we need a 2 kHz guard band, so the sampling rate should be at least 44 kHz.

4 Square-Wave and Sawtooth-Wave Generation

The next major class of analog waveforms are Square waves and Sawtooth waves.[1] We will show how to easily derive these from a BLIT via integrations which are linear transforms (that can be implemented with trivial filters), so that they preserve the bandlimited nature of the BLIT.

Successive Integration of BLIT

Sawtooth

A sawtooth function can be generated as follows:

$$Saw(n) = \sum_{k=0}^{n} BLIT(k) - C_1$$

where $C_1 = \frac{1}{\text{Period}} \sum_0^{\text{Period}} \text{BLIT(k)}$, the DC component of the BLIT.

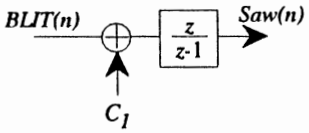

Figure 3: *Direct Sawtooth Generation*

Which is trivally impementable with a single sum and one-pole digital filter. The offet C_1 is the average value of $BLIT$, which should be subtracted off to keep the integration from ramping off to infinity (or saturating).

Rectangle

$$\begin{aligned} Rect(n) &= \sum_{k=0}^{n} \text{BLIT(k)} - \text{BLIT(k} - \text{k}_0\text{)} - C_2 \\ &= \sum_{k=0}^{n} \text{BP-BLIT}_{k_0}(k) - C_2 \end{aligned}$$

Where BP-BLIT is a "BiPolar" BLIT, whose pulses alternate sign. See Section 5 for discussion on hacks for efficiently generating BP-BLIT when using DSF BLITs. It turns out that a bipolar impulse train has a DC component of zero, which means that $C_2 = 0$. The rectangle width is controlled with k_0, which can be varied to give PWM (pulse-width modulation). The range of k_0 in these equations is [0,Period]. Depending on the implementation of the BLIT, the PWM control may also be in the range [0,1].

[1] To avoid confusion, we will use the following naming convention: A "square wave" is a rectangle wave with 50% duty cycle (i.e., "rectangle wave" means a wave that can have other duty cycles). A "triangle wave" can have asymmetric up/down slopes (including the 50% duty-cycle version), and a "sawtooth" wave must have infinite-slope transitions (either up or down). Thus a sawtooth wave is a triangle wave with either 0% or 100% duty cycle. All waves can have unipolar, bipolar, or arbitrary-offset versions.

Triangle

$$Tri(n) = \sum_{k=0}^{n} Rect(k) - C_3$$

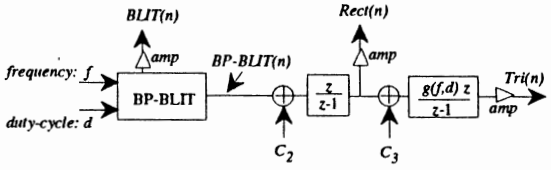

Figure 4: *Rectangle and Triangle Generation*

The offset C_3 is a function of the rectangle wave duty cycle and of a DC offset that arises from the initial conditions of the integration that produces the rectangle wave. For the Triangle to have the same amplitude as the rectangle wave, a frequency- and duty-cycle-dependant scaling must be performed on the Triangle integration:

$$Tri(n) = \sum_{k=0}^{n} g(f,d)(Rect(k) - C_3)$$

$$g(f,d) = \frac{2f}{d(1-d)}$$

Where f is the frequency in units of (cycles/sample), and d is the duty cycle ($d \in [0,1]$).

Appropriate Scalings/Offsets and Non-Steady-State Fixups

In order to keep the integrators from ramping their outputs to infinity, any DC offset in the input to the integrator must be avoided. As already noted, BP-BLIT has no DC offset, so there need be no special offsets for the first integration. The initial conditions of the first integrator, however, can produce a DC offset on the output that must be canceled before the second integration. The value of this offset is also dependant on the duty-cycle of the signal so that the correct (zero-offset) initial condition will depend on: (1) desired phase, and (2) desired duty cycle. It is important to remember that the old state (right before the change) acts as the initial condition for the integrator when the parameters are changed. Since the Second integraotr has a frequency-dependant gain, chaning frequency will also cause an offset that must be accounted for.

Leaky Integrators

The use of pure integrators can numerical errors to accumulate in the integrators, causing unwanted offsets in the signals, making the second integration essentially useless. Therefore, we move the integrator poles wfrom the unit circle slightly. These "leaky" integrators slowly forget bad initial conditions and numerical errors. Forthermore, in steady-state, the outputs of the integrators will have no DC component (because BP-BLIT has none), *regardless of initial conditions*, since the leaky integrators eventually forget them. Thus, if one can live with occasional transient DC offsets (which decay at the leak rate), then just the presence of the leaky integrators can handle all offset cases. These transients can be reduced by temporarily increasing the forgetting rate of the integrators.

If one still requires the absence of any DC offset, transient or not, then the presense of the leaky integrators makes the problem of computing appropriate offsets much more difficult.

Defs of Amplitude

Moore presents a discussion of amplitude compensation in his DSF paper. Similar compensation is necessary in BLIT generation. The compensation to be used depends on how one defines amplitude, which depends on how the signal is to be used. If the signal is to be used as an audio signal, signal power or some psychoacoustic loudness measure is appropriate, but if the signal is to be used as a control signal, a maximum-value (Chebychev) measure is more appropriate.

5 Generating Bipolar BLITs with DSF

Although it is not the most efficient method for generating BLITs, DSF can be used, and is quite interesting theoretically. Here we describe how to generate BP-BLIT using DSF without having to resort to the straight-forward difference-of-shifted-BLITs scheme.

First, we note that BLITs can be generated via DSF by replacing the sin by cos in the DSF formulas.

50% duty cycle: Next, it can be shown that using a negative a in the DSF formula $\sum_{k=1}^{N} a^k \sin(0+kf_1t)$ produces a signal that is shifted from the positive-a signal by exactly half a cycle, this gives a slightly more efficient (or elegant...) way of producing the shifted BLIT than offsetting t. This leads to showing that:

$$\sum_{k=1}^{N} a^k \cos(b+ck) - \sum_{k=1}^{N} (-a)^k \cos(b+ck)$$

$$= 2a \sum_{k=1}^{N/2} (a^2)^k \cos(\,(b+c)+2ck)$$

Thus a 50% duty-cycle bipolar DSF BLIT can be generated almost as efficiently as a single DSF BLIT. For other duty cycles, there is another variation on DSF that is of interest.

If we let a be complex and take either the real or imaginary part of the DSF, this imposes a $\sin(k\angle(a))$ (or cosine) amplitude envelope onto the harmonics, which is equivalent to a comb filtering, which in turn is equivalent to summing a real DSF with a shifted version of itself (possibly with a sign flip). See Figure ??. Thus we generate BP-BLIT simply by making a complex.

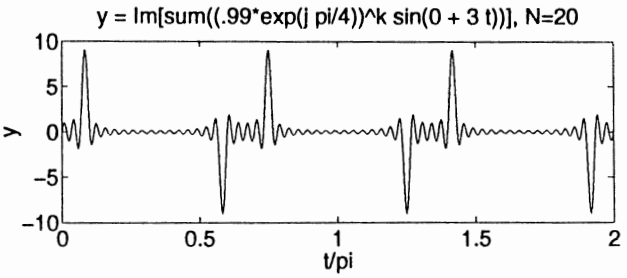

Figure 5: *Using complex multiplies in DSF to generate BP-BLIT*

References

[Kaegi and and S. Tempelaars 1978] Kaegi, and W. a S. Tempelaars. 1978. "VOSIM—A New Sound Synthesis System." *J. Audio Eng. Soc.*, 26(6):418–24.

[Mathews 1969] Mathews, M. V. 1969. *The Technology of Computer Music.* Cambridge, MA: MIT Press.

[Moorer 1975] Moorer, J. A. 1975. "The Synthesis of Complex Audio Spectra by Means of Discrete Summation Formulae." *J. Audio Eng. Soc.*, 24(Dec.):717–727 (Also svailable as CCRMA Report No. STAN-M-5).

[Roads 1996] Roads, C. 1996. *Computer Music Tutorial.* Cambridge, MA: MIT Press.

[Rodet *et al.* 1989] Rodet, X., Y. Potard, and J. Barrière. 1989. "The CHANT Project: From the Synthesis of the Singing Voice to Synthesis in General.". *Pages 449-465 of:* Roads, C. (ed), *The Music Machine.* Cambridge, MA: MIT Press.

[Schafer and Rabiner 1973] Schafer, R. W., and L. R. Rabiner. 1973. "A Digital Signal Processing Approach to Interpolation." *Proc. IEEE*, 61(June):692–702.

[Smith and Gossett 1984] Smith, J. O., and P. Gossett. 1984. "A Flexible Sampling-Rate Conversion Method." *Pages 19.4.1-19.4.2 of: Proc. ICASSP, San Diego*, vol. 2. New York: IEEE Press (An expanded tutorial based on this paper is available at `ftp://ccrma-ftp.stanford.edu/pub/DSP/Tutorials/` The tutorial can be browsed online at `http://www-ccrma.stanford.edu/~jos/`).

A longer version of this paper can be found at (`http://www-ccrma.stanford.edu/ stilti/papers`)