A GENERAL ANTIALIASING METHOD FOR SINE HARD SYNC

Pier Paolo La Pastina and Stefano D'Angelo

Orastron Srl
Sessa Cilento, Italy
pierpaolo.lapastina@orastron.com | stefano.dangelo@orastron.com

ABSTRACT

Hard sync is a feature appearing in many analog synthesizers: it consists in retriggering a slave oscillator, regardless of its phase, every time a master oscillator completes its cycle. If this process is naïvely implemented digitally, it is subject to aliasing. While for sawtooth, square, and triangle waves several effective antialiasing methods have been developed, the literature is sparser concerning sine hard sync, arguably because discontinuities of infinite order are introduced which are more difficult to handle. In this paper, we introduce a new antialiasing algorithm for sine hard sync which is obtained by filtering the hard-synced sine with a FIR lowpass kernel, as opposed to existing methods based on the windowed sinc function. We show that our method yields lower computational cost and better aliasing reduction.

1. INTRODUCTION

Synthesizers were born in the late 19th century and gained popularity especially in the 1960s and 1970s. In that era, they were mainly based on subtractive synthesis: first a sound with a rich spectrum was generated by one or more oscillators, then it was fed to a user-controlled filter to shape its harmonic content. In the 1980s, digital signal processing gradually overtook analog electronics as the main technology underlying synthesizers. This allowed for more flexibility and affordability but also introduced new technical challenges.

Indeed, it is well-known that geometric waveforms (sawtooth, square, triangle) are not bandlimited, therefore a digital oscillator that simply samples these waveforms yields aliasing distortion. In order to mitigate this problem, several oscillator algorithms have been proposed in the last decades, including those based on wavetable synthesis [1, 2, 3], oversampling [1], discrete summation formulas [4, 5], frequency-domain methods [6, 7], bandlimited impulse trains (BLITs) [8, 9], bandlimited step functions (BLEPs) [10, 11],differentiated polynomial waveforms (DPWs) [12, 13], and polynomial transition regions (PTRs) [14, 15]. The choice of the most appropriate algorithm depends on the efficiency and aliasing reduction requirements of the application at hand.

Hard sync is a feature present in some analog and digital synthesizers with two or more oscillators. Here we describe briefly its mechanism. Suppose that one oscillator, called the *master*, is running at a certain frequency, while another oscillator, called *slave*, is running at a possibly different frequency. When the master completes a cycle, the slave is retriggered, regardless of its current phase (see Figure 1). In many cases, this causes the output of the slave oscillator to have a very rich and complex spectrum, giving the impression of harmony.

In particular, even if the original slave oscillator waveform was bandlimited, the resulting synced output may not be. Thus, in the case of digital synthesizers, aliasing noise is likely to be introduced. When it comes to geometric waveforms, [10] provides good antialiasing algorithms for this scenario. The BLIT algorithm is based on the fact that the first or the second derivative of any geometric waveform is an impulse train, therefore it makes sense to replace each impulse with a lowpass filter kernel and then perform integration once or twice. If one employs the ideal lowpass filter, each impulse ends up being substituted by a sinc function, which needs to be windowed for the method to be of practical use. The BLEP variant consists in pre-integrating the windowed sinc function in order to obtain a bandlimited step and using it to replace discontinuities in the trivially-generated signal. More generally, these methods allow to replace any discontinuity in a signal, or of its derivatives, with a "bandlimited discontinuity". A similar approach was taken, for example, in [16] to develop an antialiasing method for polygonal oscillators.

This idea is exploited in [17] in order to produce 3 antialiasing methods for sine hard sync. The first two methods replace, in different ways, the discontinuities of the signal and its derivatives with their bandlimited counterparts. However, resetting the phase of a sine wave causes a discontinuity of all its derivatives, i.e., a "discontinuity of infinite order", so these methods introduce further approximations. The third method, called the "frequency shifting method", is instead conceptually based on filtering the continuous-time analytical signal before sampling and can be thus seen as the equivalent of the BLEP method. In this case, the only approximation is due to windowing.

We take inspiration from this last approach but rather develop a more straightforward formulation and replace the windowed sinc function with a FIR lowpass filter, in a similar vein to the polyBLEP method [11]. We give explicit results when the FIR kernel is polynomial (including the classical triangular kernel), a B-spline, or trigonometric. In these cases, the convolution integral can be fully expressed in terms of elementary functions (in particular, sine waves and polynomials) and the resulting algorithms give better aliasing reduction with less computational load. The method is however easily adaptable to different kernels.

The paper is organized as follows. In Section 2, we formalize the problem, showing that our antialiasing algorithm, like the BLEP algorithm, can be described in terms of *residuals*, for which explicit formulas are presented in Section 3. In Section 4, we briefly discuss the results, in particular we analyze the computational cost of our method and compare it with the frequency shifting method from [17] in two experiments, showing that our method gives superior performances with less computational load. Finally, conclusions are drawn in Section 5.

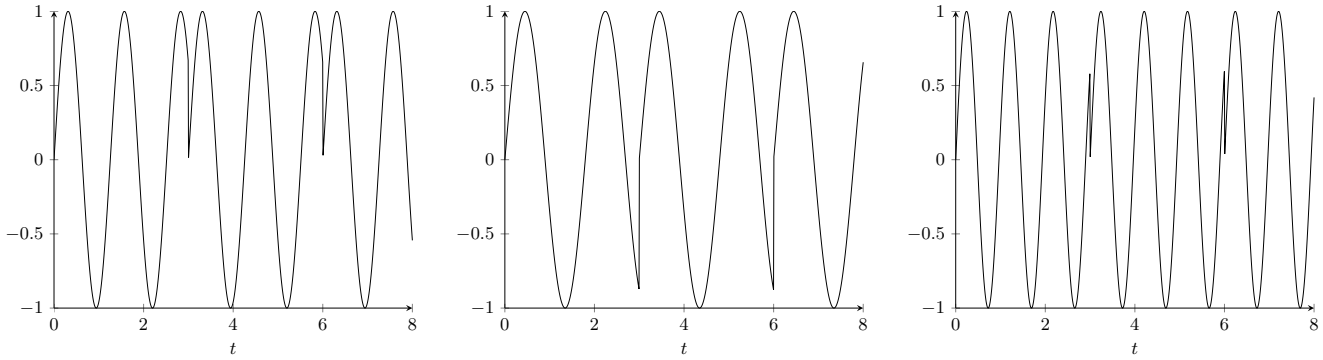

Figure 1: Examples of hard-synced sine waves.

2. PROBLEM STATEMENT

2.1. Filtered hard-synced sine

Let f_0, f_1 be the frequency of the slave and master oscillator, respectively, and set $\omega_0 = 2\pi f_0$, $T = \frac{1}{f_1}$. Suppose that hard sync is triggered at time $t = 0$. Then the resulting waveform will be:

$$x(t) = \begin{cases} \sin(\omega_0 t) & t \leq 0 \\ \sin(\omega_0 \bmod(t, T)) & t > 0 \end{cases}. \tag{1}$$

Consider now an LTI filter which has gain 1 at angular frequency ω_0 and let $h(t)$ be its impulse response. If $x(t)$ is fed into this filter, the output is clearly $y(t) = (h * x)(t)$.

In order to compute this convolution, we rewrite (1) in a more convenient way. If $u(t)$ is the Heaviside function and $s_k(t) = \sin(\omega_0(t - kT))$, then (1) can be expressed as:

$$x(t) = s_0(t) + \sum_{k=1}^{+\infty} (s_k(t) - s_{k-1}(t))u(t - kT). \tag{2}$$

Furthermore, by defining

$$\begin{aligned} f(t) &= (\sin(\omega_0 t) - \sin(\omega_0(t + T)))u(t) \\ &= -2\sin\left(\omega_0 \frac{T}{2}\right) \cos\left(\omega_0\left(t + \frac{T}{2}\right)\right) u(t), \end{aligned} \tag{3}$$

we obtain:

$$x(t) = s_0(t) + \sum_{k=1}^{+\infty} f(t - kT). \tag{4}$$

As a consequence,

$$\begin{aligned} y(t) &= (h * x)(t) \\ &= (h * s_0)(t) + \sum_{k=1}^{+\infty} h(t) * f(t - kT) \\ &= s_0(t) + \sum_{k=1}^{+\infty} (h * f)(t - kT). \end{aligned} \tag{5}$$

By further defining

$$g(t) = (h * f)(t), \tag{6}$$

one gets:

$$y(t) - x(t) = \sum_{k=1}^{+\infty} (g(t - kT) - f(t - kT)). \tag{7}$$

Finally, defining the *residual*

$$R(t) = g(t) - f(t), \tag{8}$$

we obtain:

$$y(t) = x(t) + \sum_{k=1}^{+\infty} R(t - kT). \tag{9}$$

2.2. The antialiasing problem

In order to remove aliasing from the signal (1), one would like to feed it into an ideal lowpass filter. But it is well-known that the ideal lowpass filter has an infinite impulse response, the sinc function. This implies that the corresponding residual $R(t)$ has itself infinite length, and therefore the sum in (9) is really an infinite series, making it impractical to use in real-world applications.

If, instead, we use a FIR filter, the series reduces to a finite sum. To see this, suppose that $h(t)$ is the impulse response of a FIR filter. We will also suppose, from now on, that $h(t)$ is symmetric with respect to $t = 0$, so that its support is $[-\epsilon, \epsilon]$. Then:

$$g(t) = \int_{-\epsilon}^{\epsilon} h(\theta) f(t - \theta)\, \mathrm{d}\theta. \tag{10}$$

But $f(t-\theta) = 0$ for $\theta > t$, so $(h*f)(t) = 0$ when $t \leq -\epsilon$. On the other hand, for $t > 0$, $f(t)$ is a sine wave of angular frequency ω_0. When $t > \epsilon$, the convolution (10) coincides with the convolution with a plain sine wave, and so $(h * f)(t) = f(t)$ because h has gain 1 at ω_0. It follows that the support of $R(t)$ is also $[-\epsilon, \epsilon]$, and so in (9) only a finite number of the supports of $R(t - kT)$ will overlap, making the sum a finite sum.

Up to now, the FIR filters used in the literature for sine hard sync were obtained by windowing the sinc function. In this paper, we will show that one can achieve better results by using other lowpass kernels. In the next section, we will give explicit formulas for $g(t)$ when $h(t)$ is a polynomial (in particular, a triangular) kernel, a B-spline kernel or a trigonometric kernel.

3. EXPLICIT FORMULAS

3.1. Polynomial kernel

Let

$$\rho(t) = \begin{cases} \sum_{k=0}^{N} a_k |t|^k & |t| \leq \epsilon \\ 0 & |t| > \epsilon, \end{cases} \tag{11}$$

where we impose $\sum_{k=0}^{N} a_k \epsilon^k = 0$ in order to ensure continuity. Define:

$$p_1(t) = \sum_{l=0}^{N} \sum_{m=0}^{\lfloor \frac{N-l-1}{2} \rfloor} (-1)^{m+l+1} \frac{(2m+l+1)!}{l!} \frac{a_{2m+l+1}}{\omega_0^{2m+2}} t^l; \tag{12}$$

$$p_2(t) = \sum_{l=0}^{N} \sum_{m=0}^{\lfloor \frac{N-l}{2} \rfloor} (-1)^{m+l} \frac{(2m+l)!}{l!} \frac{a_{2m+l}}{\omega_0^{2m+1}} t^l; \tag{13}$$

$$q_1(t) = \sum_{l=0}^{N} \sum_{m=0}^{\lfloor \frac{N-l-1}{2} \rfloor} (-1)^{m} \frac{(2m+l+1)!}{l!} \frac{a_{2m+l+1}}{\omega_0^{2m+2}} t^l; \tag{14}$$

$$q_2(t) = \sum_{l=0}^{N} \sum_{m=0}^{\lfloor \frac{N-l}{2} \rfloor} (-1)^{m} \frac{(2m+l)!}{l!} \frac{a_{2m+l}}{\omega_0^{2m+1}} t^l; \tag{15}$$

$$G = |q_1(\epsilon)\cos(\omega_0\epsilon) + q_2(\epsilon)\sin(\omega_0\epsilon) + p_1(0)|. \tag{16}$$

One can verify that $2G$ is the gain at angular frequency ω_0 of the LTI filter whose impulse response is $\rho(t)$. Therefore, in order to define a filter with gain 1 at ω_0, we define

$$h(t) = \frac{\rho(t)}{2G}. \tag{17}$$

A direct computation shows that

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0 \frac{T}{2}\right)}{G} \left(\cos\left(\omega_0 \left(t + \frac{T}{2} + \epsilon\right)\right) p_1(-\epsilon) \right. \\ &- \sin\left(\omega_0 \left(t + \frac{T}{2} + \epsilon\right)\right) p_2(-\epsilon) \\ &\left. - \cos\left(\omega_0 \frac{T}{2}\right) p_1(t) + \sin\left(\omega_0 \frac{T}{2}\right) p_2(t) \right) \end{aligned} \tag{18}$$

when $-\epsilon \leq t \leq 0$, and

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0 \frac{T}{2}\right)}{G} \left(\cos\left(\omega_0 \left(t + \frac{T}{2} + \epsilon\right)\right) p_1(-\epsilon) \right. \\ &- \sin\left(\omega_0 \left(t + \frac{T}{2} + \epsilon\right)\right) p_2(-\epsilon) \\ &- 2\cos\left(\omega_0 \left(t + \frac{T}{2}\right)\right) p_1(0) - \cos\left(\omega_0 \frac{T}{2}\right) q_1(t) \\ &\left. + \sin\left(\omega_0 \frac{T}{2}\right) q_2(t) \right) \end{aligned} \tag{19}$$

when $0 < t \leq \epsilon$.

3.1.1. Triangular kernel

Formulas above can be specialized to the case of triangular kernel by taking $\epsilon = 1$ and $\rho(t) = 1 - |t|$ for $|t| \leq 1$. The impulse response becomes

$$h(t) = \frac{\omega_0^2}{2(1-\cos\omega_0)}(1 - |t|) \tag{20}$$

for $|t| < 1$, and we obtain

$$\begin{aligned} g(t) = &\frac{\sin(\omega_0 \frac{T}{2})}{1-\cos(\omega_0)} \left(\cos\left(\omega_0 \left(t + \frac{T}{2} + 1\right)\right) \right. \\ &\left. + \omega_0 \sin\left(\omega_0 \frac{T}{2}\right)(1+t) - \cos\left(\omega_0 \frac{T}{2}\right) \right) \end{aligned} \tag{21}$$

when $-1 \leq t \leq 0$, and

$$\begin{aligned} g(t) = &\frac{\sin(\omega_0 \frac{T}{2})}{1-\cos(\omega_0)} \left(\cos\left(\omega_0 \left(t + \frac{T}{2} + 1\right)\right) \right. \\ &- 2\cos\left(\omega_0 \left(t + \frac{T}{2}\right)\right) + \omega_0 \sin\left(\omega_0 \frac{T}{2}\right)(1-t) \\ &\left. + \cos\left(\omega_0 \frac{T}{2}\right) \right) \end{aligned} \tag{22}$$

when $0 < t \leq 1$.

3.2. B-spline kernel

We recall that the B-spline kernel is obtained by convolving twice the rectangular kernel with itself. It takes the form:

$$\beta(t) = \begin{cases} \frac{1}{2}\left(t + \frac{3}{2}\right)^2 & -\frac{3}{2} \leq t < -\frac{1}{2} \\ \frac{3}{4} - t^2 & -\frac{1}{2} \leq t < \frac{1}{2} \\ \frac{1}{2}\left(t - \frac{3}{2}\right)^2 & \frac{1}{2} \leq t < \frac{3}{2} \\ 0 & \text{elsewhere} \end{cases}. \tag{23}$$

In order to use a filter that has gain 1 at ω_0, we set

$$h(t) = \left(\frac{\omega_0^2}{2(1-\cos\omega_0)}\right)^{\frac{3}{2}} \beta(t). \tag{24}$$

Please notice that this is not a polynomial, but only a piecewise polynomial function, hence the result in Subsection 3.1 does not apply. However, one can perform a similar computation, obtaining

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0 \frac{T}{2}\right)}{2^{\frac{7}{2}}(1-\cos\omega_0)^{\frac{3}{2}}} \left(8\sin\left(\omega_0 \left(t + \frac{T+3}{2}\right)\right) \right. \\ &+ 4\omega_0^2 \sin\left(\omega_0 \frac{T}{2}\right) t^2 + \left(12\omega_0^2 \sin\left(\omega_0 \frac{T}{2}\right) \right. \\ &- 8\omega_0 \cos\left(\omega_0 \frac{T}{2}\right)\Big) t + (9\omega_0^2 - 8)\sin\left(\omega_0 \frac{T}{2}\right) \\ &\left. - 12\omega_0 \cos\left(\omega_0 \frac{T}{2}\right)\right) \end{aligned} \tag{25}$$

if $-\frac{3}{2} \leq t \leq -\frac{1}{2}$,

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0 \frac{T}{2}\right)}{2^{\frac{5}{2}}(1-\cos\omega_0)^{\frac{3}{2}}} \left(\left(4\cos\left(\omega_0 \frac{T+3}{2}\right) \right. \right. \\ &- 12\cos\left(\omega_0 \frac{T+1}{2}\right)\Big) \sin(\omega_0 t) + \left(4\sin\left(\omega_0 \frac{T+3}{2}\right) \right. \\ &\left. - 12\sin\left(\omega_0 \frac{T+1}{2}\right)\right) \cos(\omega_0 t) - 4\omega_0^2 \sin\left(\omega_0 \frac{T}{2}\right) t^2 \\ &\left. + 8\omega_0 \cos\left(\omega_0 \frac{T}{2}\right) t + (3\omega_0^2 + 8)\sin\left(\omega_0 \frac{T}{2}\right)\right) \end{aligned} \tag{26}$$

if $-\frac{1}{2} < t \leq \frac{1}{2}$, and

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0 \frac{T}{2}\right)}{2^{\frac{7}{2}}(1-\cos\omega_0)^{\frac{3}{2}}}\left(\left(8\cos\left(\omega_0\frac{T+3}{2}\right)\right.\right. \\ &- 24\cos\left(\omega_0\frac{T+1}{2}\right) + 24\cos\left(\omega_0\frac{T-1}{2}\right)\Big)\sin(\omega_0 t) \\ &+ \left(8\sin\left(\omega_0\frac{T+3}{2}\right) - 24\sin\left(\omega_0\frac{T+1}{2}\right)\right. \\ &\left.+ 24\sin\left(\omega_0\frac{T-1}{2}\right)\right)\cos(\omega_0 t) + 4\omega_0^2\sin\left(\omega_0\frac{T}{2}\right)t^2 \\ &- \left(12\omega_0^2\sin\left(\omega_0\frac{T}{2}\right) + 8\omega_0\cos\left(\omega_0\frac{T}{2}\right)\right)t \\ &\left.+ (9\omega_0^2 - 8)\sin\left(\omega_0\frac{T}{2}\right) + 12\omega_0\cos\left(\omega_0\frac{T}{2}\right)\right) \end{aligned} \tag{27}$$

if $\frac{1}{2} < t \leq \frac{3}{2}$.

3.3. Trigonometric kernel

Finally, we consider a kernel of the form

$$\tau(t) = \begin{cases} \sum_{k=0}^{N} a_k \cos\left(\frac{k\pi}{\epsilon}t\right) & |t| \leq \epsilon \\ 0 & \text{elsewhere} \end{cases}, \tag{28}$$

where we impose $\sum_{k=0}^{N}(-1)^k a_k = 0$ in order to ensure continuity. Please notice that this family includes Hann, Hamming, Blackman and Nuttall windows. In order to have gain 1 at ω_0, we set:

$$\begin{aligned} K &= \omega_0\epsilon\sum_{k=0}^{N}\frac{(-1)^k a_k}{\omega_0^2\epsilon^2 - k^2\pi^2}; \\ h(t) &= \frac{\tau(t)}{2\epsilon\sin(\omega_0\epsilon)K}. \end{aligned} \tag{29}$$

Then

$$\begin{aligned} g(t) = &\frac{\sin\left(\omega_0\frac{T}{2}\right)}{K}\left(\omega_0\epsilon\sin\left(\omega_0\frac{T}{2}\right)\sum_{k=0}^{N}a_k\frac{\cos\left(\frac{k\pi}{\epsilon}t\right)}{\omega_0^2\epsilon^2 - k^2\pi^2}\right. \\ &+ \pi\cos\left(\omega_0\frac{T}{2}\right)\sum_{k=0}^{N}k a_k\frac{\sin\left(\frac{k\pi}{\epsilon}t\right)}{\omega_0^2\epsilon^2 - k^2\pi^2} \\ &\left.- K\sin\left(\omega_0\left(t+\epsilon+\frac{T}{2}\right)\right)\right) \end{aligned} \tag{30}$$

for $t \in [-\epsilon, \epsilon]$.

4. DISCUSSION

Let us have a closer look at the equations above. Apart from the trigonometric case, the residual is a linear combination of sine waves of angular frequency ω_0 and a polynomial. Please notice that sine waves with same frequency and different gains and phases can be expressed as a single sine wave using trigonometric identities. It follows that computing a residual essentially consists in evaluating a polynomial and a sinusoid.

In the trigonometric case, the polynomial is replaced by a trigonometric polynomial, so the computational load is seemingly increased. However, the trigonometric polynomials in Subsection 3.3 can be seen as polynomials in $\cos\left(\frac{\pi}{\epsilon}t\right)$ and $\sin\left(\frac{\pi}{\epsilon}t\right)$ via Chebyshev polynomials. This means that one needs to evaluate 2 polynomials and 3 sinusoids, hence the computational load is just slightly higher than in the polynomial case.

The overall methodology does not necessarily depend on the chosen kernel type, therefore adapting it to different kernels mostly translates into recomputing the formulas for residuals.

4.1. Experiments

Here we compare the proposed method with the "frequency shifting method" in [17], as it is reported to be the most efficient. We stress that the frequency shifting method depends on the choice of a window function, while our method depends on the choice of a lowpass kernel. For the sake of a fair comparison, we have chosen a Kaiser window ($\alpha = 4$) and a triangular kernel, respectively, both of length 2 samples.

We let $f_s = 44100$ Hz and we call $f_0 = \frac{\omega_0}{2\pi}$ the frequency of the slave oscillator and f_1 the frequency of the master oscillator. In the figures we show the magnitude responses obtained using the trivial hard sync algorithm, the frequency shifting method, and the proposed method. Specifically, we set $f_0 = 2900.33$ Hz, $f_1 = 866.42$ Hz in the first experiment (Figure 2), and $f_0 = 517.88$ Hz, $f_1 = 1888.10$ Hz in the second experiment (Figure 3).

The results clearly show that our method easily provides better aliasing suppression than the frequency shifting method when identical window/kernel length are used. Furthermore, in the transition regions the frequency shifting method requires the evaluation of the exponential integral function, which is considerably more computationally expensive than evaluating sinusoids and polynomials. Therefore, we conclude that the proposed method achieves better results at a lower computational cost.

A GNU Octave implementation of both methods is available on the companion web page for this paper[1].

5. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced a new antialiasing method for sine hard sync that outperforms the best known method both in terms of computational efficiency and aliasing reduction. Explicit formulations are given for polynomial, B-spline, and trigonometric kernels, yet the methodology is applicable to any FIR lowpass kernel.

A similar approach could also be applied to other waveforms, for example to polygonal oscillators studied in [16].

[1] https://www.dangelo.audio/dafx20in22-sinesync.html

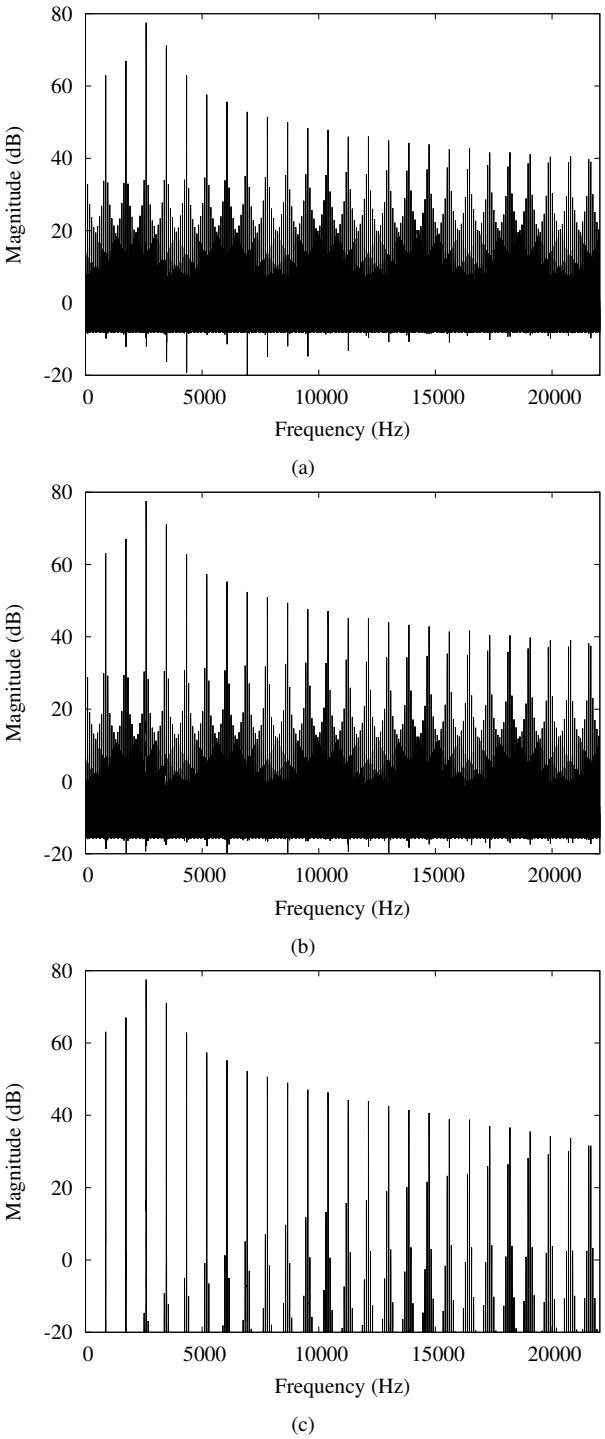

Figure 2: Magnitude spectrum of a sine wave of frequency $f_0 = 2900.33$ Hz synced to a master oscillator of frequency $f_1 = 866.42$ Hz generated at a sample rate $f_s = 44100$ Hz using (a) no antialiasing, (b) the frequency shifting method, and (c) the proposed method.

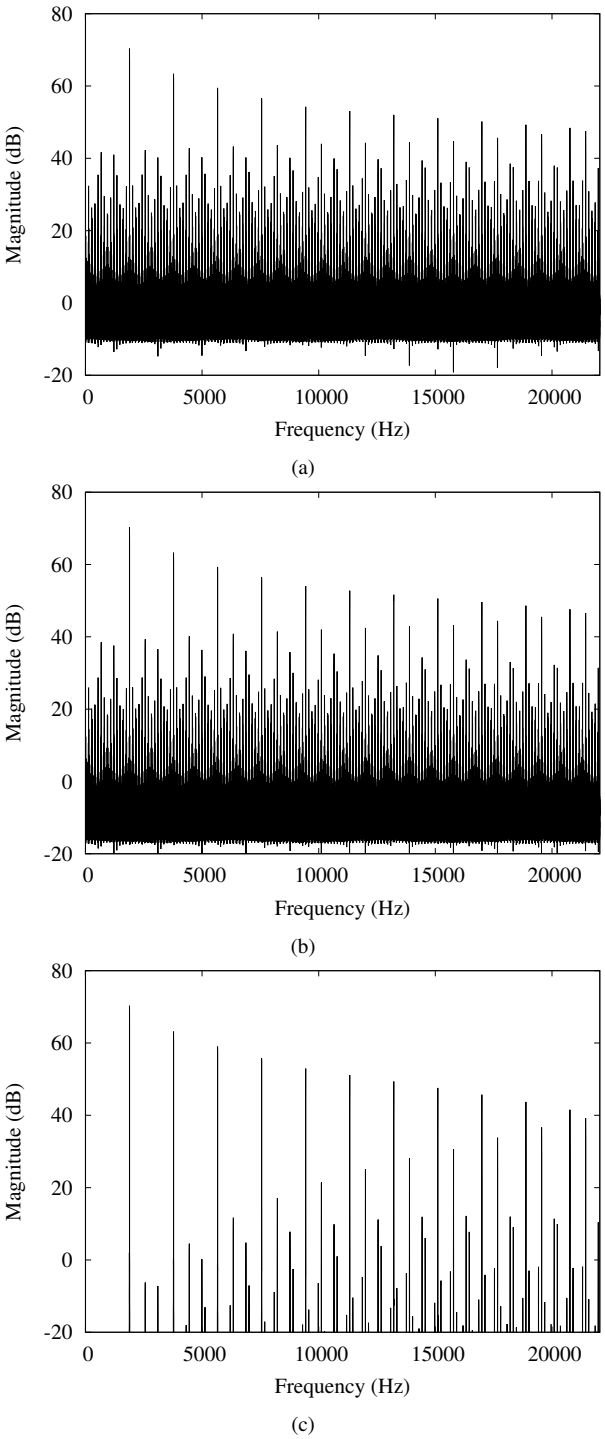

Figure 3: Magnitude spectrum of a sine wave of frequency $f_0 = 517.88$ Hz synced to a master oscillator of frequency $f_1 = 1888.10$ Hz generated at a sample rate $f_s = 44100$ Hz using (a) no antialiasing, (b) the frequency shifting method, and (c) the proposed method.

6. REFERENCES

[1] Hal Chamberlin, *Musical Applications of Microprocessors, 2nd edition*, Hayden Book Company, 1985.

[2] Dana C. Massie, "Wavetable sampling synthesis," in *Applications of Digital Signal Processing to Audio and Acoustics*, Mark Kahrs and Karlheinz Brandenburg, Eds., pp. 311–341. Kluwer Academic Publishers, 1998.

[3] Phil Burk, "Band limited oscillators using wave table synthesis," in *Audio Anecdotes II - Tools, Tips, and Techniques for Digital Audio*, Ken Greenebaum and Ronen Barzel, Eds., pp. 37–53. A. K. Peters Ltd., 1998.

[4] Godfrey Winham and Kenneth Steiglitz, "Input generators for digital sound synthesis," *J. of the Acoust. Soc. of America (JASA)*, vol. 47, no. 2B, pp. 665–666, 1970.

[5] James A. Moorer, "The synthesis of complex audio spectra by means of discrete summation formulas," *J. of the Audio Eng. Soc. (JAES)*, vol. 24, no. 9, pp. 717–727, 1976.

[6] Xavier Rodet and Philippe Depalle, "Spectral envelopes and inverse FFT synthesis," in *Proc. 93rd AES Conv.*, San Francisco, USA, 1992.

[7] Glen Deslauriers and Colby Leider, "A bandlimited oscillator by frequency-domain synthesis for virtual analog applications," in *Proc. 127th AES Conv.*, New York, USA, 2009.

[8] Timothy S. Stilson and Julius O. Smith, "Alias-free digital synthesis of classic analog waveforms," in *Proc. 1996 Intl. Computer Music Conf. (ICMC'96)*, Hong Kong, China, 1996, pp. 332–335.

[9] Timothy S. Stilson, *Efficiently-Variable Non-Oversampling Algorithms for Virtual-Analog Music Synthesis – A Root-Locus Perspective*, Ph.D. thesis, Stanford University, Stanford, CA, 2006.

[10] Eli Brandt, "Hard sync without aliasing," in *Proc. 2001 Intl. Computer Music Conf. (ICMC'01)*, Havana, Cuba, 2001, pp. 365–368.

[11] Vesa Välimäki and Antti Huovilainen, "Antialiasing oscillators in subtractive synthesis," *IEEE Signal Process. Magazine*, vol. 24, pp. 116–125, 2007.

[12] Vesa Välimäki, "Discrete-time synthesis of the sawtooth waveform with reduced aliasing," *IEEE Signal Process. Letters*, vol. 12, pp. 214–217, 2005.

[13] Vesa Välimäki, Juhan Nam, Julius O. Smith, and Jonathan Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Trans. Audio, Speech, and Lang. Process. (IEEE TASLP)*, vol. 18, pp. 786–798, 2010.

[14] Jari Kleimola and Vesa Välimäki, "Reducing aliasing from synthetic audio signals using polynomial transition regions," *IEEE Signal Process. Letters*, vol. 19, no. 2, pp. 67–70, 2012.

[15] Dániel Ambrits and Balázs Bank, "Improved polynomial transition regions algorithm for alias-suppressed signal synthesis," in *Proc. 10th Sound and Music Comp. Conf. (SMC 2013)*, Stockholm, Sweden, 2013, pp. 561–568.

[16] Christoph Hohnerlein, Maximilian Rest, and Julian D. Parker, "Efficient anti-aliasing of a complex polygonal oscillator," in *Proc. 20th Intl. Conf. Digital Audio Effects (DAFx-17)*, Edinburgh, UK, 2017, pp. 125–129.

[17] Vadim Zavalishin, "Generation of bandlimited sync transitions for sine waveforms," Available at https://www.native-instruments.com/fileadmin/ni_media/downloads/pdf/SineSync.pdf, 2009.

[18] John Lazzaro and John Wawrzynek, "Subtractive synthesis without filters," in *Audio Anecdotes II - Tools, Tips, and Techniques for Digital Audio*, Ken Greenebaum and Ronen Barzel, Eds., pp. 55–63. A. K. Peters Ltd., 1998.