Sinusoidal Synthesis Optimization

Georgios Marentakis and Kristoffer Jensen

Department of Datalogy, University of Copenhagen
email: {babis, krist}@diku.dk

Abstract

This paper presents a method for improving synthesis efficiency in sinusoidal models. In the case of synthesizing polyphonic music, many overtones can be pruned, using perceptual and trigonometric methods. The main contribution of this work is a pre-processing algorithm giving a significant reduction of the number of sinusoids to synthesize (from several thousands to around one hundred), consisting of an optimized masking and hearing threshold calculation, and a sinusoidal merging procedure for sinusoids close in frequency. The frequency difference that can be merged has been determined, and increased using analytic band limitation.

1 Introduction

Sinusoidal models for synthesis have been popular in research laboratories for many years, and industry shows sustained interest in this model. It is, however, expensive, since each note/sound demands up to several hundred pure voices. Chaudhary (2001) shows that standard hardware can synthesize up to around 350 partials (or around 1000 using inverse FFT). Therefore, more efficient synthesis methods must be used, when synthesizing for instance polyphonic music.

Previous attempts in increasing the number of sinusoids in synthesis include the inverse FFT (Rodet and Depalle 1992) second-order recursive oscillators (Smith and Cook 1992) in general (Hodes and Freed 1997) or specific hardware (De Bernardinis *et al.* 1997). Other methods include the multirate additive synthesis (Phillips *et al.* 1996) and the group additive synthesis (Kleczowski 1989).

The method introduced in this paper utilizes perceptual criteria (Zwicker and Fastl 1990) and trigonometric results to prune and merge partials in order to minimize the computational load.

The sinusoidal optimization has been developed to be used with the Timbre Engine (Marentakis and Jensen 2001), but it can be used in any block-based sinusoidal synthesis engine, or as a pre-processor to hardware implementations.

2 Perceptual optimization criterion

The perceptual criteria used for pruning sinusoids are frequency masking and absolute hearing threshold in quiet (Zwicker and Fastl 1990). These methods have been used for different purposes in sound analysis, representation and synthesis before. However, they have generally been used in off-line analysis stages, and not many attempts have been made at optimizing the speed of the masking calculations.

Masking and hearing threshold pruning have been used extensively in analysis and compression, such as the popular MPEG-1 Layer 3 (MP3) standard (ISO/IEC 11172-3 1992). In addition, different researchers have used the signal-to-mask (SMR) ratio to determine the partials to prune in synthesis (Chaudary 2001) or in data compression (Garcia and Pampin 1999).

The masking and hearing threshold criteria permit the pruning of a majority of sinusoids in polyphonic music, with no decrease in sound quality.

In the following, optimization of the masking and hearing threshold calculations is done, which results in limiting the pre-processing costs.

2.1 Partial pruning

The psychoacoustic phenomenon of masking has been used for many years to remove redundant information from audio signals. Only simultaneous masking (frequency masking) is considered here, a phenomenon that occurs when a strong sinusoid (masker) reduces the perceived loudness of a weak neighboring (in frequency) sinusoid (masked) to the point where it becomes inaudible. In general, a positive and negative linear slope defines the masking threshold, when using the Bark (Zwicker and Fastl 1990) frequency scale.
In addition to simultaneous masking, the absolute hearing threshold in quiet is used, given as (Terhard 1979)

$$T_q = 3.64\left(f_{kHz}\right)^{-0.8} - 6.5e^{-0.6\left(f_{kHz}-3.3\right)^2} + 10^3\left(f_{kHz}\right)^4. \quad (1)$$

The hearing threshold in quiet is set, assuming that the playback level is close to 0 dB SPL for the smallest possible output level (±1bit) (Painter and Spaniard 2000).
In our work, a negative slope of 22 dB/Bark, a positive slope of 18 dB/Bark, and a masker distance of -4 dB is

assumed. An example of the masking and hearing thresholds are shown in figure 1.

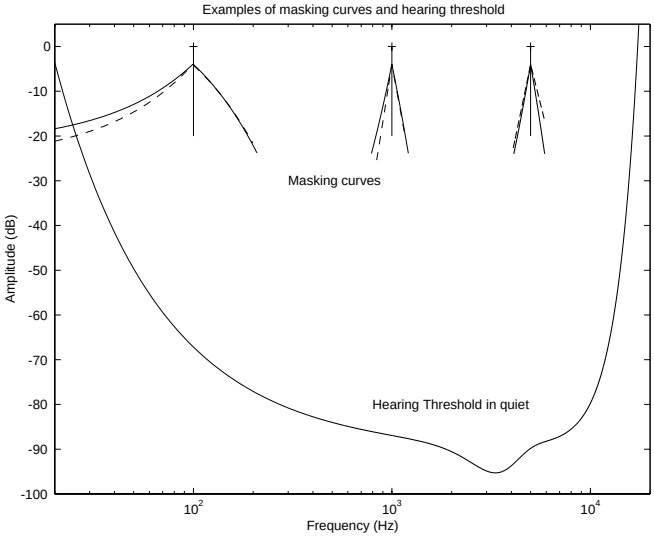

Figure 1. Examples of standard (dashed) and approximated (solid) masking curves and hearing threshold in quiet.

2.2 Implementation considerations

In order to limit the pre-processing costs, a number of simplifications have been done to the masking calculations.

It is important to notice that the optimization can be very different for, for instance, a fixed-point DSP than for a general-purpose computer.

Two steps are necessary for each partial, first determine if it's above the hearing threshold, and then determine if it's masking any neighboring partials (i.e. inside approximately one critical band).

The hearing threshold is stored in Hz shifted a convenient number of bits to the right (8). Therefore, a partial is masked if,

$$a_{dB} < T_q\left(round(f >> N)\right). \tag{2}$$

Several simplifications are done on the masking calculations in order to minimize calculation costs. The positive and negative slopes are set equal to approximately 20 dB/Bark, the slopes are linear in Hz (instead of in Bark), and the masking width is a second order approximation of the critical bandwidth, given as (Zwicker and Fastl 1990),

$$BW_c = 25 + 75\left(1 + 1.4\, f_{kHz}{}^2\right)^{0.69}. \tag{3}$$

The neighboring partial p_1 is masked, then, if,

$$a_1^{dB} < a_0^{dB} - d - \frac{\hat{S}}{\hat{BW}_c}\left|f_1 - f_0\right|, \tag{4}$$

where d is the masking threshold (4dB), $\hat{S}$ is the masking threshold at one critical band distance (20dB), and the approximated second-order critical bandwidth has been found using the weighted Levenberg-Marquardt algorithm (Moré 1977),

$$\hat{BW}_c = 100 + 0.1 f_0 + 10^{-5} f_0^2, \tag{5}$$

The resulting approximated masking curves can be seen in figure 1. If more perceptually accurate masking curves are necessary, d or $\hat{S}$ can be increased slightly.

3 Sinusoidal fusion

We take advantage of the possibility of generating two sinusoids as one sinusoid with time-varying amplitude and frequency to merge partials close in frequency. This reduces the computational load, since two merged sinusoids only demand one synthesized sinusoid. In the scope of sinusoidal synthesis optimization the amplitude and frequency envelopes are approximated using linear segments inside the synthesis buffer.

The merging (after perceptual pruning) removes up to around two hundred sinusoids, without altering the sound quality. It is believed, however, that the merging is useful in musically gratifying situations, for instance when several instruments play the same note.

3.1 Trigonometric identities

The expression

$$y = \sin\left(\int_0^t \omega_0(\tau)d\tau + \varphi_0\right) + a\sin\left(\int_0^t \omega_1(\tau)d\tau + \varphi_1\right) \tag{6}$$

can be written as

$$y = a_{env}(t)\sin\left(\int_0^t \omega_{env}(\tau)d\tau + \varphi_3\right), \tag{7}$$

where the amplitude envelope is (assuming $\omega_d = \omega_1 - \omega_0$ and $\varphi_d = \varphi_1 - \varphi_0$)

$$a_{env}(t) = \sqrt{1 + a^2 + 2a\cos\left(\int_0^t \omega_d(\tau)d\tau + \varphi_d\right)} \tag{8}$$

and the frequency envelope is

$$\omega_{env}(t) = \omega_0(t) + \omega_d(t)\frac{a\left(a + \cos\left(\int_0^t \omega_d(\tau)d\tau\right) + \varphi_d\right)}{1 + a^2 + 2a\cos\left(\int_0^t \omega_d(\tau)d\tau\right) + \varphi_d} \tag{9}$$

and the starting phase is

$$\varphi_3 = \arctan\frac{\sin(\varphi_0) + a\sin(\varphi_1)}{\cos(\varphi_0) + a\cos(\varphi_1)}. \tag{10}$$

3.2 Phase reconstruction

The phases need to be synchronized across block borders, both for the case of merging two sinusoids into one and for the case of merged partials being re-separated.

In the first case, φ_d and φ_3 are calculated using the initial phases. In the second case, the individual phases have to be re-estimated. The phase difference is found using equation (8), and consequently, φ_0 is found from equation (10). Additional information necessary for finding the correct solution is extracted from the slope of the amplitude envelope.

3.3 Analytic band limited frequency envelope

In order to optimize the sinusoidal synthesis, linear frequency and amplitude segments are used inside each buffer. This sometimes introduces problems because of the high bandwidth of the frequency envelope in some cases, which results in aliasing if the sampling theorem is not satisfied. This can be prevented, either by increasing the block sampling rate (i.e. using smaller blocks), or by bandlimiting the frequency envelope. The amplitude envelope decreases fast with frequency, and it has not been found necessary to bandlimit.

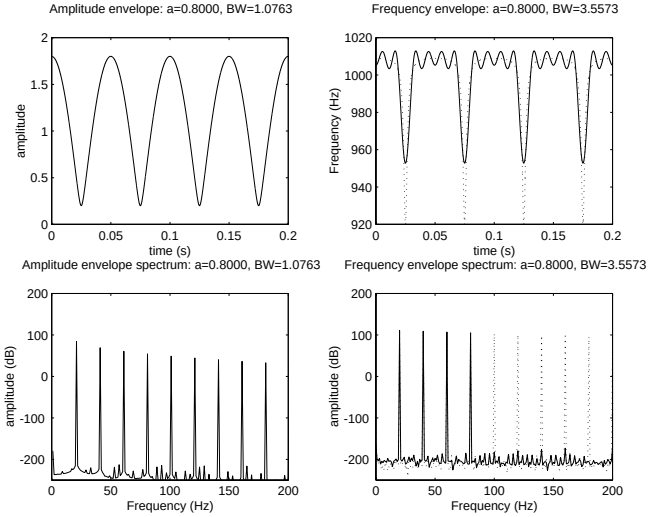

Figure 2. Amplitude and frequency envelopes (original (dotted), bandlimited (solid)) with corresponding spectra. f_0=1000 Hz, f_1=1020, a=0.8.

In order to bandlimit the frequency envelope, the equation (9) can be identified (Gradshteyn and Ryzhik, 1965) as an infinite sum of exponentially decreasing harmonic sinusoids. For the bandlimited frequency envelope, only the first components are used, either by using the expression for the sum of the first N harmonics (Gradshteyn and Ryzhik, 1965), or by simply summing the first N sinusoids. The bandlimited frequency envelope greatly enhances the range of approximation, as will be seen in the next section.

Examples of the amplitude and frequency envelopes are shown in figure 2. The original frequency envelope is shown dotted, and the bandlimited version is solid.

3.4 Range of approximation validity

In order to achieve a good approximation of the frequency and amplitude envelopes, the frequency difference should be chosen appropriately so as to satisfy the sampling theorem with enough components.

The bandwidth of the relevant bandlimited frequency envelope with N components is,

$$BW = N \cdot f_d < f_s / 2B_{sz} \,. \tag{11}$$

N must always be chosen so as not to violate the sampling theorem. Informal listening tests and examination on the resulting spectrum, in particular regarding component level change, and spurious component levels (which must be below the masking threshold), have determined the maximum frequency difference. It has been found to be approximately 30 Hz (N=3 components) for sampling rate 44100 and block size 256, and 60 Hz for half the block size.

4 Efficiency gain

This section evaluates the efficiency gain of the pruning and merging, that is compares the additional cost of finding the pruned and merged partials to the gain from not having to synthesize as many partials.

In the following, a number of notes N_{notes}, with a total number of partials $N_{partials}$, is assumed to be reduced to a smaller number of partials $N'_{partial}$ to actually synthesize.

4.1 Cost of pre-processing

The cost of pre-processing can be divided into three steps; the initial sorting of the partials, the determination of the partials to prune and the determination of the partials to merge.

The structure of the partials to process is helpful in the first step, since each note's partials are assumed to be sorted in frequency. Therefore, to find the next lowest frequency only involves keeping N_{notes}, indexes into the lowest next frequency of each note, and determining the lowest frequency of each of the N_{notes}, index frequencies.

When we have a selected frequency, this partial is compared to the hearing threshold, and if it's above, then the partials in the vicinity (inside one critical band) are compared to the masking threshold (for the selected partial) at the frequency difference, and purged, if below. Finally, the frequency difference is compared to the maximum frequency difference, and if below, then the two partials are merged.

In summary, the sorting and looping takes $O(N_{notes} \cdot N_{partials})$ (or $O(\log(N_{partials})N_{partials})$ if no prior knowledge is available), which is predominant in the pre-processing. Inside the loop, the hearing threshold is only a comparison and the masking

and merging involves only a few neighboring partials, in general. In comparison, the synthesis takes blocksize B_{sz} times O($N_{partials}$). The cost of the optimized synthesis is approximately O($N_{notes} \cdot N_{partials}$)+$B_{sz}$O($N'_{partial}$). Therefore, the pre-processing is only beneficial approximately if the relative number of pruned and merged partials is greater than one minus the number of notes divided by the block size, multiplied with the ratio of the pre-processing cost to the synthesis cost (per partial). This is satisfied, particularly if the pre-processing involves only a small percentage of partials (the ones not pruned by the lower, stronger partials). In our implementation, the optimized code is always better than twice as good as the non-optimized code.

4.2 Pruning Efficiency

In order to test the efficiency of the pruning and merging, a large number (3315) of musical sounds from different instruments, pitches and intensities have been collected. The spectral envelope and mean frequencies have been calculated, and used to calculate the number of partials resulting after pruning and merging. The resulting numbers can be seen in figure 3.

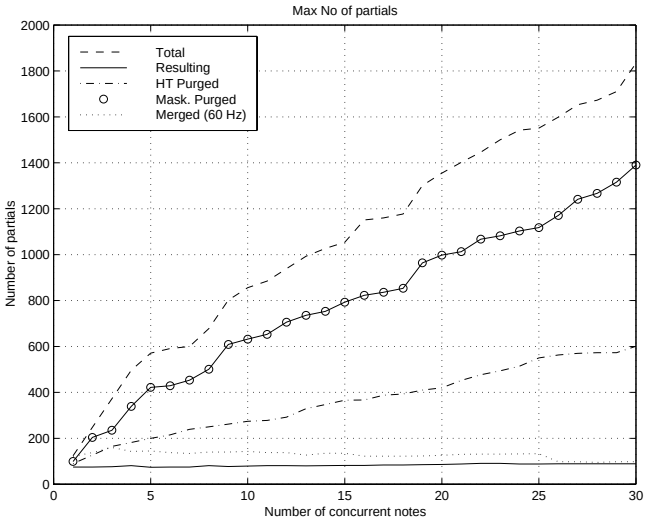

Figure 3. Total and resulting (after optimization) number of partials as a function of number of concurrent notes. The number of purged and merged partials is also shown.

Please observe that it is the maximum number of partials for each curve that is shown, and therefore the sums do not add up. Also, the sounds have generally been analyzed with a small number (up to approximately 64) of harmonic partials, to limit the analysis cost. A larger number of partials per sound would increase the pruning efficiency.

5 Conclusion

This paper has presented a system for the pruning and merging of sinusoids in real-time synthesis. The pruning is done using an optimized masking algorithm, and the merging is done using trigonometric identities. In the merging, the maximum frequency difference was found by bandwidth considerations, and increased by analytic band limitation.

Tests have shown that the number of sinusoids to synthesis can be reduced from several thousands to around hundred for all kind of musical sounds. This leads to a reduction in computational load, which is always better than a factor of two. The sinusoidal optimization does not influence the sound quality in any case.

The sinusoidal synthesis optimization can be used in standard hardware, on DSPs, or as a pre-processor to any hardware/software sinusoidal synthesis method.

References

Chaudhary, A., 2001. "Perceptual scheduling in real-time music and audio applications", *Ph.D. dissertation*, Berkelye University.

De Bernardinis, F., R. Roncella, R. Saletti, P. Terreni, and G. Bertini 1997, "A single-chip 1,200 sinusoid real-time generator for additive synthesis of musical signals", presented at EEE International Conference on Acoustics, Speech, and Signal Processing, Munich, Germany.

Garcia, G., J. Pampin, 1999. "Data compression of sinusoidal modeling parameters based on psychoacoustic masking", *Proc. ICMC.* International Computer Music Conference, Beijing, China.

Gradshteyn, I. S., and I. M. Ryzhik, 1965. *Table of integrals, series and products*, Academic Press.

Hodes, T. and A. Freed, 1999. "Second-order recursive oscillators for musical additive synthesis applications on SIMD and VLIW processors", *Proc. ICMC.* International Computer Music Conference, Beijing, China.

ISO/IEC JTC1/SC29/WG11 MPEG, IS11172-3 1992. "Information Technology-Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s, Part 3: Audio. (MPEG1)"

Kleczowski, P., 1989. "Group Additive Synthesis". *Computer Music Journal* 13(1).

Marentakis, G., and K. Jensen 2001. "The Timbre Engine: Progress report". *Proc of the Workshop on current research directions in computer music*, Barcelona, Spain.

Moré, J. J., 1977. "The Levenberg-Marquardt algorithm: Implementation and theory". *Lecture notes in mathematics*, Edited by G. A. Watson, Springer-Verlag, 1977.

Painter, T., and Spanias, A., 2000. "Perceptual Coding of Digital Audio," *Proceedings of IEEE*, pp. 451-513, Vol. 88, No.4.

Phillips, D., A. Parvis, S. Johnson, 1996. "Multirate additive synthesis". *Proc. of the Int. Comp. Music Conf.*

Rodet X., P. Depalle 1992, "Spectral envelope and inverse FFT synthesis". *93rd AES Convention*, San Francisco, October.

Smith, J. O., and P. R. Cook, "The Second-Order Digital Waveguide Oscillator", *Proc. Int. Computer Music Conf.* (ICMC-92), pp. 150-153, San Jose, 1992.

Terhard, E, 1979. "Calculating virtual pitch", *Hearing Research*, pp. 155-182.

Zwicker, E., and H. Fastl, 1990. *Psychoacoustics: Facts and Models*, Springer-Verlag, Berlin, Heidelberg.