Optimized Multiple Wavetable Interpolation

JONATHAN MOHR
Augustana Faculty
University of Alberta
Camrose, Alberta
CANADA

http://www.augustana.ca/˜mohrj/

XIAOBO LI
Department of Computing Science
University of Alberta
Edmonton, Alberta
CANADA

http://www.cs.ualberta.ca/˜li/

Abstract: One effective approach to music analysis/synthesis is multiple wavetable interpolation, which matches an audio signal at selected breakpoints by determining weightings for several wavetables; the sound is resynthesized using multiple wavetable additive synthesis by interpolating between the weightings for each wavetable at consecutive breakpoints. This article presents a new breakpoint-matching algorithm which uses the single-source acyclic weighted shortest path algorithm to choose breakpoint matches in a globally optimal way.

Key-Words: Music analysis/synthesis, multiple wavetable interpolation, spectral matching.

1 Introduction

Multiple wavetable interpolation [1, 2] is a form of analysis/synthesis in which a digital waveform is converted to the frequency domain by a short-time Fourier transform and reduced to a set of shared breakpoints [3] by piecewise linear approximation (PLA) of the spectral envelopes of its harmonics; the spectrum at each breakpoint is then matched by determining weightings for a small number of selected wavetables, and the sound is resynthesized using multiple wavetable additive synthesis [4, 5] by interpolating between the weightings for each wavetable at consecutive breakpoints.

A number of spectra are selected to comprise a set of basis spectra—a *wavetable bank*[1]—that will be used in weighted additive combinations to approximate the actual spectrum at each breakpoint of each tone to be synthesized. Typically, these basis spectra are selected from the breakpoint spectra, but they could be selected by other means, including spectral principal components analysis (PCA) [8], a genetic algorithm [4, 5], a clustering algorithm [9], an iterative combinatorial method [10], or by hand-selection [1].

Given a particular wavetable bank, a set of breakpoint data representing a particular tone, and the number of oscillators (N_{osc}) to be used in resynthesis, the breakpoint matching algorithm selects, by index, at most N_{osc} wavetables from the bank which, in weighted combination, best match the spectrum at each breakpoint according to some error measure. As Horner, Beauchamp, and Haken have shown [5], the problem of determining the weightings (amplitude factors) of a set of basis spectra that provide the best match to a particular spectrum in a least-squares sense is an instance of the general linear least squares problem and can be solved by use of the normal equations [11].

In contrast to multiple wavetable synthesis [4, 5], which uses all of the selected basis spectra in each match, multiple wavetable interpolation uses only a subset of the basis spectra at each match point (breakpoint). If the subset of the wavetables used at one breakpoint differs from the subset used at the next breakpoint, two or more oscillators must be used to crossfade the changing wavetables between match points in order to avoid audible clicks and spectral discontinuities [1].

Horner [2] tested multiple wavetable interpolation on three different mid-range instrumental tones: a trumpet F4, a piano C4, and a muted trombone B♭3. Basis spectra were selected using a genetic algorithm; only the first 20 harmonics of each spectrum were used to evaluate the quality of the matches to the spectra at selected match points (ten equally spaced in time in the attack portion and ten similarly spaced through the rest of the tone). The need to crossfade between changing wavetables was dealt with by im-

[1] The basis spectra are here collectively referred to as a wavetable bank, although for the purposes of breakpoint matching they are initially represented in the frequency domain as vectors of harmonic amplitudes; at synthesis time, each vector is converted to an actual wavetable—a table of the time-domain amplitude values of one cycle of the waveform—for use by a table-lookup oscillator [6, 7].

posing two simple constraints: if synthesis is to be performed with N_{osc} oscillators, only $N_{\text{osc}} - 1$ wavetables can be used at each match point, and at most one wavetable is allowed to change from one match point to the next. This method might be called *constrained matching*.

A sample implementation of constrained matching was also provided. It performs an exhaustive search to select the best combination of $N_{\text{osc}} - 1$ wavetables to use at the breakpoint with the peak RMS amplitude, then works backward and forward to neighboring match points, changing at most one of the wavetables, using exhaustive search to decide what change to make, if any.

Figure 1 illustrates some of the possible combinations of wavetable selections at consecutive breakpoints, B_i and B_{i+1}, given four oscillators[2]. Of the possibilities illustrated, only that in part (c) would be possible using constrained matching. In addition, Horner's method would allow the same three wavetables to be used at consecutive breakpoints, leaving one oscillator unused for the duration between B_i and B_{i+1}.

This article presents a method that satisfies the requirement that wavetables must be faded in and out at the beginning and end of each span of use, respectively, but is not subject to the restriction of selecting $N_{\text{osc}} - 1$ wavetables at each breakpoint. The method uses the single-source acyclic weighted shortest path algorithm [12] to find a globally optimal set of weighted wavetable matches across all breakpoints, given a particular error measure and a specified method of choosing an initial best match of a specified size at each breakpoint.

2 Optimized Breakpoint Matching

Optimized breakpoint matching is a three-stage process. First, an initial match is selected for each breakpoint spectrum. Next, the set of wavetables of each initial match is overlapped with the sets selected for preceding and subsequent breakpoints in order to give the optimizer some flexibility about when to fade a wavetable in or out. Finally, the optimizer decides which wavetables will actually be used at each breakpoint and assigns wavetables to oscillators.

[2] The dots represent oscillators that have been assigned a particular wavetable at the indicated breakpoint; the open circles indicate oscillators at zero amplitude. The downward- and upward-angled arrows are intended to suggest the fade-out and fade-in of a wavetable to or from zero amplitude, respectively.

2.1 The Initial Match

In the first stage of the breakpoint matching algorithm, an initial match of the desired size is found for each breakpoint. The number of different wavetables used in the match, N_{wt}, can maximally be the same as the number of oscillators to be used in the synthesis stage, but may also reasonably be less than N_{osc}, since the set of wavetables to be considered for final use at a given breakpoint may be augmented with additional tables in later stages of the matching algorithm.

The best possible match at each breakpoint would be found by an exhaustive search of all $\binom{N_{\text{tables}}}{N_{\text{osc}}}$ combinations of wavetables selected N_{osc} at a time, where N_{osc} is the number of oscillators, from a wavetable bank of size N_{tables}. However, the cost of such a search becomes prohibitive for more than 3 or 4 oscillators (depending on the size of the wavetable bank) [13]. Furthermore, finding the best possible match of size N_{osc} for each breakpoint spectrum at this stage of the algorithm does not necessarily produce the best final result, since it is necessary to fade out a wavetable that ceases to be used from one breakpoint to the next or to fade in one that comes into use; as a result of this requirement, a set of matches to breakpoint spectra that has greater consistency (i.e., that uses many of the same wavetables over a number of consecutive breakpoints) may lead to a better overall result than a set of matches with high specificity but greater variety of wavetable usage.

One way to reduce the cost of a search is to focus the search by pruning the search tree. This can be done in the present case by performing an exhaustive search for the best matches of some size less than N_{osc} and then extending the first-level search by a second level that seeks to augment only those sets of wavetables that provided a best match to at least one breakpoint spectrum in the first-level search. For example, if a 4-wavetable match is desired, the search performed at this stage could search for the best 3-wavetable matches in the first level and augment those sets with a fourth wavetable in the second-level search (a "3+1" search). Alternatively, the first-level search could seek only 2-wavetable matches which would be augmented with two additional wavetables (a "2+2"search) in the second-level search.

2.2 Overlapping of Wavetable Sets

The second stage of the breakpoint matching algorithm is intended to provide more flexibility in the subsequent optimization stage which, as part of its

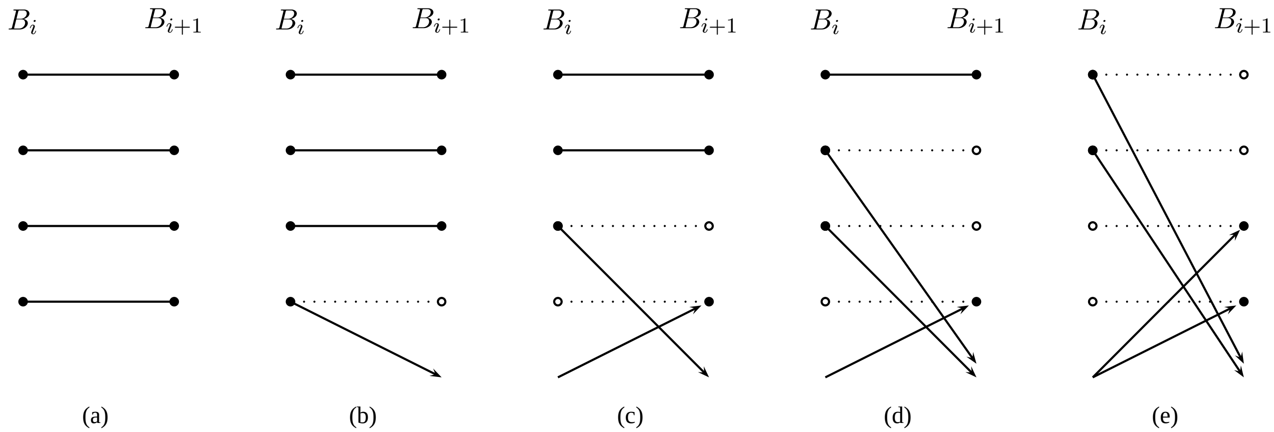

Figure 1. Possible oscillator assignments with four oscillators.

task of assigning wavetables to oscillators, must decide when to fade a wavetable in or out of use. For example, the initial matches for some consecutive set of breakpoint spectra[3] might include a wavetable that passes out of use and then back into use, as illustrated in Figure 2.

In this case, the optimizer is likely to pick one of the two assignments of wavetables to oscillators shown in Figure 2, parts (b) and (c), where the dash indicates the point at which the wavetable assigned to the relevant oscillator at the previous breakpoint has been faded out to zero amplitude and a new wavetable—the one assigned to this oscillator at the next breakpoint—begins a fade-in from zero amplitude. If the use of wavetable 21 at breakpoint w results in a lower overall error level than the use of wavetable 13 at both breakpoints v and x, then the first option will be used; otherwise, the latter option will be preferred. (It is assumed here that wavetables 5 and 8 are very important in achieving a good match at all five breakpoints, since they appear in all five matches.)

However, it is possible—indeed, highly likely—that an even lower overall error level would be achieved by allowing the optimizer to use wavetable 13 instead of 21 at breakpoint w: the higher error resulting from substituting wavetable 13 for 21 at breakpoint w in Figure 2(b) will likely be offset by the lower error levels achieved by using three-wavetable matches instead of two-wavetable matches at both breakpoints v and x, and a three-wavetable match will surely be better than a two-wavetable match at breakpoint w in option (c).

The optimizer could be tuned to look for special cases like this through a look-ahead or look-behind routine, but a more general solution to the problem of selecting the best points for fade-in and fade-out of wavetables can be expected to yield even better results. For example, it could be the case that the use of wavetable 21 at breakpoint w is crucial to achieving a low error measure at that breakpoint but that a two-wavetable match would be more tolerable at breakpoint y than at breakpoint x. If the optimizer were allowed to use wavetables from the initial matches at previous or subsequent breakpoints, then matches such as those depicted in parts (d) and (e) of Figure 2 would also be possible and may provide a lower overall error than either of the options (b) and (c).

The general solution adopted for this breakpoint matching algorithm is to include a stage in which the wavetable sets found by the initial matching phase for each breakpoint are overlapped with the wavetable sets at preceding and following breakpoints before the optimizer makes oscillator-assignment decisions in the following stage. An overlap distance of two breakpoints on either side of the current breakpoint was used in our testing of this algorithm. Increasing the overlap distance increases the number of possibilities that must be evaluated by the optimizer, so it is best to limit the amount of overlap to distances from one to three.

[3] The letters $u \ldots y$ are used as breakpoint identifiers instead of actual breakpoint numbers because this is a hypothetical example. The dashes indicate points at which wavetables are faded out and in.

u:	5	8	13	u:	5	8	13	u:	5	8	13	u:	5	8	13	u:	5	8	—
v:	5	8	13	v:	5	8	—	v:	5	8	13	v:	5	8	—	v:	5	8	21
w:	5	8	21	w:	5	8	21	w:	5	8	—	w:	5	8	21	w:	5	8	21
x:	5	8	13	x:	5	8	—	x:	5	8	13	x:	5	8	21	x:	5	8	21
y:	5	8	13	y:	5	8	13	y:	5	8	13	y:	5	8	—	y:	5	8	—
	(a)				(b)				(c)				(d)				(e)		

Figure 2. A sequence of initial wavetable matches (a) and some possible optimized oscillator assignments. The matches in parts (d) and (e) are possible only if the optimizer is allowed to use wavetables from the initial matches at previous or subsequent breakpoints.

2.3 Optimization of Oscillator Assignments

In the final phase of the optimized breakpoint matching algorithm, a weighted wavetable is assigned to each available oscillator at each breakpoint such that the overall error is minimized, taking into account the need to fade a wavetable in or out when it begins or ceases to be used.

The optimization of oscillator assignments is achieved by modeling the problem as a vertex-weighted directed acyclic graph (DAG) and using the single-source acyclic weighted shortest path algorithm [12, §24.2]. A DAG is constructed such that each vertex represents a particular wavetable set at a particular breakpoint and each edge represents a legal transition to some wavetable set at the next breakpoint. The weight (cost) of each vertex is the least-squares error of the fit of the weighted wavetables to the breakpoint spectrum. The sequence of vertices on the shortest path from the start vertex to the end vertex represents the globally optimal sequence of sets of wavetables, one set per internal breakpoint.

The wavetables of each set are then assigned to oscillators so as to ensure continuity of wavetable assignments from one breakpoint to the next.

2.3.1 Construction of the DAG

The construction of the DAG on which the shortest path algorithm will be invoked must take into account the requirement to fade in a wavetable that will begin to be used at some internal breakpoint and to fade out one that ceases to be used.

The algorithm that adds vertices to the DAG must also generate vertices that do *not* include all the wavetables of the best match at a given breakpoint so that one or more oscillators can be used to fade out a wavetable from the previous breakpoint to the current one and to fade in another wavetable from the current breakpoint to the next. In particular, if the set of wavetables at the current vertex of the graph is S_{current} and the set of wavetables eligible for use at the next breakpoint is S_{next}, then vertices should be generated at the next breakpoint to represent the wavetable set $S_{\text{current}} \cap S_{\text{next}}$ and all the subsets of this set.

If the match represented by the current vertex includes one or more unallocated oscillators or wavetables that have just been faded out to zero amplitude (i.e., if $|S_{\text{current}}| < N_{\text{osc}}$), then any of these "zero wavetables" can be replaced by any member of the set S_{next} at the next breakpoint, and vertices should be added to the DAG to represent these possible matches.

These two possibilities can be used in combination as well: any or all members of the current set can become "zeros" at the same time that any zeros in the current set are replaced by wavetables from S_{next} at the next breakpoint[4].

However, a vertex should *not* be added to the graph at the current breakpoint if there can be no outgoing edges from it to any vertex associated with the next breakpoint. For example, if $N_{\text{osc}} = 3$ and the wavetable sets for the current and next breakpoints are $\{2, 3, 5, 7, 8\}$ and $\{4, 5, 6, 8, 9\}$, respectively, then the algorithm should avoid generating a vertex at the current breakpoint for the wavetable selection $\{2, 3, 7\}$, since all three wavetables would have to be faded out simultaneously by the next breakpoint. It is assumed that each match to a spectrum at an internal breakpoint must consist of at least one wavetable,

[4]The generation of duplicate wavetable sets at the next breakpoint is to be avoided: a zero in the current set cannot be replaced by a member of S_{next} that is already in the wavetable set of an adjacent vertex as a result of the first possibility above.

since only the external breakpoints (the beginning and end of the tone) correspond to silence.

Because a one-wavetable match to a breakpoint spectrum is not likely to be very accurate (except at those breakpoints for which the spectrum was selected for inclusion in the wavetable bank), our implementation of the optimized matching algorithm provides for the specification of a minimum number of wavetables to be used in the final matches. In our testing, the minimum match size was $N_{\text{osc}} - 2$. This constraint reduced the number of vertices added to the DAG, and thus the work done by the optimizer, without significantly affecting the final result of the optimization stage.

The full recursive algorithm for constructing the DAG is presented in pseudocode in [9].

3 Results

A set of 198 tones played by sixteen different instruments, spanning the range from A1 to B6 by minor thirds, was selected from the *McGill University Master Samples* collection [14] for the purpose of testing the proposed analysis/synthesis method. The tones were divided into groups according to pitch ranges in order to allow more harmonics to be used in the synthesis of lower-pitched tones than for higher tones. The tones were analyzed using a period-synchronous phase vocoder from the *SNDAN* sound analysis suite [15], and breakpoints were selected by piecewise linear approximation. The basis spectra for the wavetable bank for each group were selected by applying a clustering algorithm to the breakpoint spectra of the tones in that group (with some hand-tuning of the resulting selections).

The running time of the single-source acyclic weighted shortest path algorithm is $\Theta(V + E)$ if the graph is implemented with an adjacency list representation. However, the size of the graph is not simply determined by the size of the initial best match to each breakpoint spectrum nor by the number of oscillators to be used in synthesis. Because the number of vertices is determined by the number of combinations of wavetables drawn from the set of eligible wavetables at each breakpoint, the size of the graph is related to the complexity of the tone being analyzed.

There is a general relationship, however, between the average number of vertices and edges per breakpoint and the specificity of the initial search for best matches. The graph resulting from an initial "3+1" search is three to four times as large, per breakpoint, as the graph constructed from the results of an initial "2+1" search; a graph derived from an initial "4+0" search is, on average, about one-quarter larger per breakpoint than one resulting from a "3+1" search.

The sequence of matches found by the shortest path algorithm is globally optimal in a mathematical sense, but may not provide the best sequence of matches from a perceptual point of view. For example, several studies [16, 17] have found that the attack portion of a tone has high perceptual relevance, so smaller errors in matching spectra in the attack portion may be more perceptually significant than larger errors in the sustain or release portions. However, the algorithm is globally optimal with respect to a given error measure; an error measure could include a higher weighting for errors in the attack portion of a tone.

As is the case with almost any algorithm, there are trade-offs to be made between the time expended in seeking a result and the quality of the result found, and the breakpoint matching algorithm presented here is no different, as will be seen in the tables of results of multi-level exhaustive searches and oscillator assignment optimization presented in the next two sections.

All timing information was derived by executing `C++` programs on a 500 MHz Intel Pentium II Celeron-based system with 256 MB of memory and 750 MB of swap space, running the Linux operating system, kernel version 2.4.5. Programs were compiled with the GNU `g++` compiler, version 2.95.3, against the `Glibc` C library, version 2.2.5 (Linux `libc` 6), and the `libstdc++-3` Standard C++ library, version 2-2-2.10.0.

3.1 Multi-Level Exhaustive Search Results

A summary of the results of multi-level exhaustive search for initial matches to the breakpoint spectra of the 43 tones in Group 1 (pitches from A♯1 to C♯3) is presented in Table 1, which lists the mean execution time (in seconds) and the mean RMS error for each type of search, in increasing order by time. (Complete tables of results are available in the companion technical report [18].)

The table confirms that, in general, a search that takes more time (i.e., traverses more of the search tree) produces better results, but there are some cases that do not conform to this general model. For example, for a 5-table match, a "3+2" search is to be preferred to a "4+1" search since it produces better results in significantly less time.

These results should not be considered in isola-

Type of Search	Group 1	
	Time	Error
1+0	0.15	196.4
1+1	1.32	134.4
2+0	6.41	132.9
1+2	7.20	101.6
2+1	12.3	100.9
3+0	145	98.6
3+1	157	82.1
2+2	181	81.4
2+2+1	199	70.5
3+2	477	68.6
4+0	2114	79.4
4+1	2132	69.2

Table 1. Summary of multi-level exhaustive search results for Group 1.

Search Type	Group 1		
	T_{opt}	T_{total}	Error
3 Oscillators			
1+0	1.52	1.67	121.9
1+1	0.94	2.25	119.9
2+0	1.00	7.41	118.9
1+2	2.39	9.58	113.7
2+1	2.48	14.8	114.0
3+0	2.87	148	**113.5**
4 Oscillators			
1+2	13.3	20.5	97.8
2+1	13.2	25.5	97.6
3+0	15.6	160	96.9
3+1	46.5	203	**94.5**
2+2	54.4	236	95.1
4+0	66.8	2181	94.5
5 Oscillators			
3+1	220	377	83.5
2+2	253	434	83.2
2+2+1	340	540	81.9
3+2	358	835	81.8
4+0	284	2398	82.5
4+1	351	2483	**81.5**

Table 2. Summary of 3-, 4-, and 5-oscillator optimization results for various initial search types.

tion, but in conjunction with the results of the optimization phase, below. It will be seen that a "3+1" search executes about an order of magnitude faster than a "4+0" search, yet yields about the same or better error rates, on average, after optimization. A "2+1" search is another order of magnitude faster than a "3+1" search, at the cost of an increased average error of about 50%; however, after optimization, the difference in average matching error is reduced to between 2% and 5%.

3.2 Oscillator Assignment Optimization

The mean optimization time, total time, and RMS error of the optimization of the matches to breakpoint spectra of the tones in Group 1 found by each search type for 3, 4, and 5 oscillators are summarized in Table 2. The lowest mean RMS error value for a given number of oscillators is highlighted.

These data suggest that single-level exhaustive searches (especially "4+0" searches) tend not to be worth the computation time they require. Although the "4+0" search took an order of magnitude more time, the optimization of "3+1" matches in Group 1 gave the same (best) result as the "4+0" optimization. Similarly, although "3+0" searches resulted in the best optimized 3-oscillator matches (on average) for tones in Group 1, the "1+2" and "2+1" initial searches were of comparable quality but required an order of magnitude less search time.

In general, the best n-oscillator final matches were achieved by optimizing n-table initial matches. For example, none of the lowest mean RMS error levels are achieved in the 3-oscillator case by optimizing initial "1+0," "1+1," or "2+0" matches; similarly, initial "1+2," "2+1," and "3+0" searches lead to poorer 4-oscillator final matches than initial searches of depth 4.

The detailed results [18] show that the best final matches do not necessarily result from optimizing the best initial matches. For example, the best final 3-oscillator match to the bass clarinet G2 tone resulted from optimizing the initial "1+2" match, even though the "3+0" and "2+1" initial matches were better. Nor do optimizations that take more time always produce better final matches: the best 3-oscillator matches to both the bass clarinet G2 and the horn E2 were found by the fastest of the six optimizations tested.

No single search and optimization method gave the best results in all cases. A good strategy would be to optimize the matches found by two different medium-cost searches and use the better of the final two matches for each instrumental tone.

Constrained Matching		
Oscillators	**Time**	**Error**
3	1.29	136.1
4	11.3	106.7
5	156	90.0

Table 3. Mean matching time and RMS error for constrained matching of tones in Group 1.

3.3 Comparison with the Constrained Matching Method

Horner's constrained matching method [2] was implemented and tested on the same instrumental tones on which our optimization method was tested, using the same wavetable banks for the five groups of tones[5]. Table 3 indicates the average matching time and RMS error of 3-, 4-, and 5-oscillator constrained matches to the breakpoint spectra of the tones in Group 1.

Figure 3 compares the average results of the constrained matching method with those of optimized matching using 3, 4, and 5 oscillators for the tones of Group 1. In the graph, lines connect the data points for a given number of oscillators such that each line traverses the points in the same order as they appear in Table 2. It shows that, while the constrained matching method is faster than any of the types of multi-level exhaustive search optimization for a given number of oscillators, the error levels produced by constrained matching are significantly higher than those of the optimized matches, and are closer to those achieved by optimization with one fewer oscillators.

If oscillators are relatively cheaper than computation time, or if the speed with which results are achieved is more important than the quality of the results, then the constrained matching method is to be recommended. However, since the spectral matching procedure need be performed only once for each tone, a higher quality result using fewer oscillators would typically be desired, and the time required to achieve that goal would be of secondary importance. Thus, the method presented here for selecting optimal matches that fully utilize the available oscillators is likely to be preferred in most cases.

[5] It should be clearly noted that this comparison with Horner's method is restricted to the breakpoint spectrum matching and oscillator assignment components only. Horner did not use large, general-purpose wavetable banks, but used a genetic algorithm to select small sets of basis spectra—two to ten wavetables in size—that were particular to each instrument being matched.

4 Conclusion and Suggestions for Further Research

The proposed three-stage optimization method—selecting an initial match to each breakpoint spectrum, overlapping the matches of adjacent breakpoints to form wavetable sets, then optimizing oscillator assignments by constructing a vertex-weighted directed acyclic graph and executing the single-source acyclic weighted shortest path algorithm on it—avoids the two ad hoc restrictions of constrained matching: that the match to each breakpoint spectrum be limited to a size one less than the number of oscillators to be used in synthesis, and that the matching process begin with the peak-amplitude breakpoint and work outward. It has been demonstrated that the optimization method yields significantly improved matches compared to the constrained matching method.

The main limitation of this method is that, due to the use of shared breakpoints and the calculation of a single weighted average frequency differential for each breakpoint, multiple wavetable interpolation synthesis does not work as well for tones with inharmonic partials as for purely harmonic tones. Lee and Horner [19] used a form of group synthesis, with independent frequency deviations for each group, to simulate the stretched partials of piano tones. So and Horner [20] introduced a hierarchical grouping technique to extend this form of group wavetable synthesis to instruments that exhibit more complicated inharmonicity than the piano, such as plucked string tones. It would be interesting to explore the possibility of optimizing the assignment of wavetables to oscillators across groups of wavetable matches when adapting these methods to multiple wavetable interpolation synthesis.

Another possible avenue of research would be the addition of a stochastic component like that used in spectral modeling synthesis [21] or transient modeling synthesis [22] to the optimized multiple wavetable interpolation model in order to model the noise which is so clearly present in many instrumental tones. The inherently harmonic nature of multiple wavetable synthesis limits its ability to lend realism to the resynthesis of tones that include the scrapes, thumps, chiffs, and air flow noises of actual instruments but that are otherwise harmonic in structure.

Further work is also required to find a perceptually verified error measure. In an attempt to give more emphasis to matching error in the attack and release phases of a tone, Horner [4, 5, 2] used a relative spec-

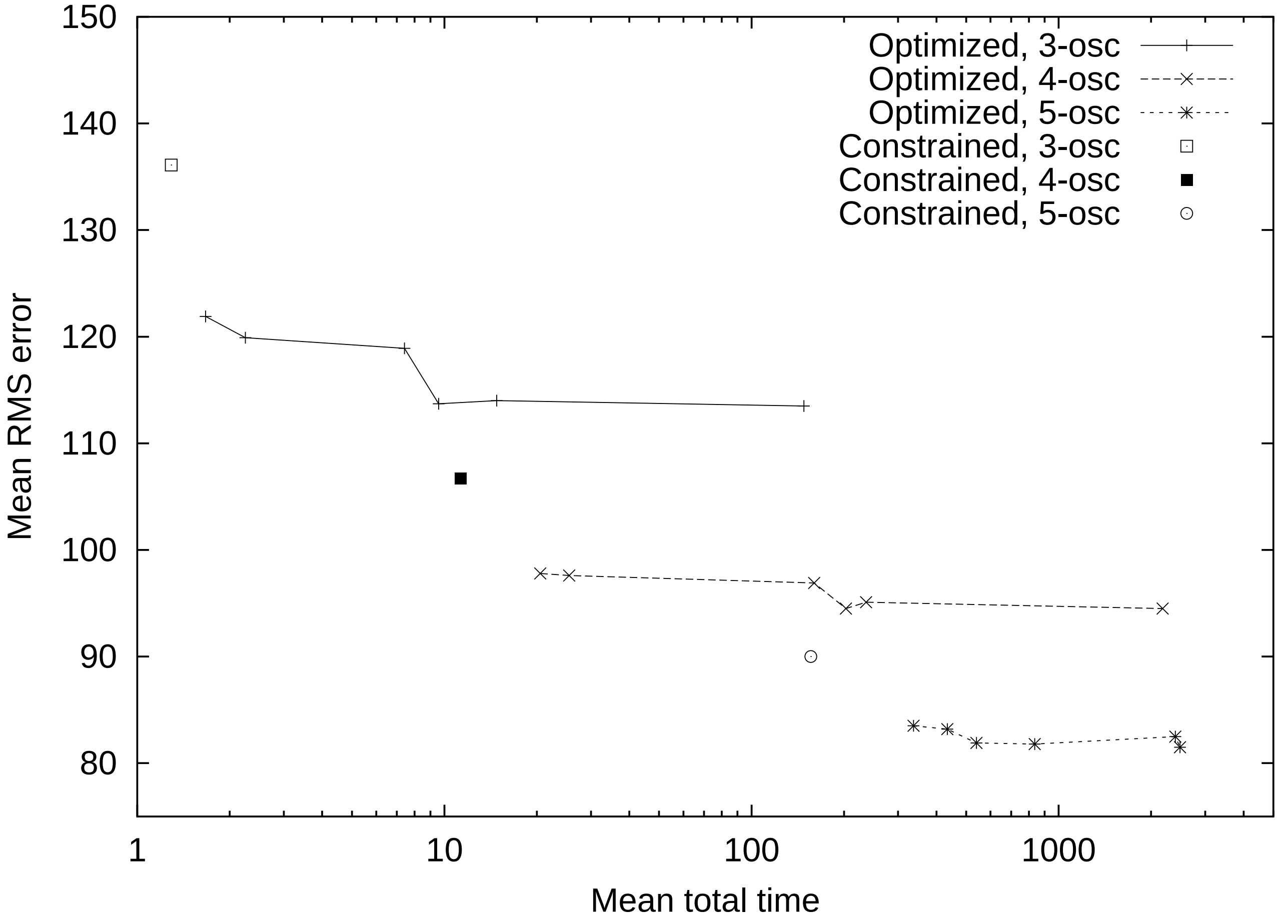

Figure 3. Comparison of Horner's constrained matching with optimized multi-level exhaustive search results.

tral error measure that in effect divides the error at each match point by the RMS amplitude of the signal at that point. However, a relative error measure tends to overemphasize artifacts near zero amplitude. Wun and Horner [23] tested a variant of this relative error measure that takes into account the masking of some partials by other partials within a critical bandwidth, and found that it provided minor improvement. A better solution might be to use an adaptive error measure which, like Horner's relative error measure, gives greater weight to the lower-amplitude parts of the tone but, unlike the relative error measure, does not overemphasize the portions with near-zero amplitude.

References

[1] M.-H. Serra, D. Rubine, and R. Dannenberg, "Analysis and synthesis of tones by spectral interpolation," *Journal of the Audio Engineering Society*, vol. 38, pp. 111–128, Mar. 1990.

[2] A. Horner, "Computation and memory tradeoffs with multiple wavetable interpolation," *Journal of the Audio Engineering Society*, vol. 44, pp. 481–496, June 1996.

[3] A. Horner and J. Beauchamp, "Piecewise-linear approximation of additive synthesis envelopes: A comparison of various methods," *Computer Music Journal*, vol. 20, pp. 72–95, Summer 1996.

[4] A. Horner, *Spectral Matching of Musical Instrument Tones*. PhD thesis, University of Illinois at Urbana-Champaign, 1993.

[5] A. Horner, J. Beauchamp, and L. Haken, "Methods for multiple wavetable synthesis of musical instrument tones," *Journal of the Audio Engineering Society*, vol. 41, pp. 336–355, May 1993.

[6] H. Chamberlin, “Advanced real-time music synthesis techniques,” *BYTE*, vol. 5, pp. 70–94, 180–196, Apr. 1980.

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

[8] G. Sandell and W. Martens, “Prototyping and interpolation of multiple musical timbres using principal components-based analysis,” in *Proceedings of the 1992 International Computer Music Conference* (A. Strange, ed.), pp. 34–37, International Computer Music Association, 1992.

[9] J. Mohr, *Music Analysis/Synthesis by Optimized Multiple Wavetable Interpolation*. PhD thesis, University of Alberta, Edmonton, Alberta, Canada, 2002.

[10] A. Ng and A. Horner, “Iterative combinatorial basis spectra in wavetable matching,” *Journal of the Audio Engineering Society*, vol. 50, pp. 1054–1063, Dec. 2002.

[11] W. H. Press *et al.*, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge: Cambridge University Press, second ed., 1992.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA: MIT Press, second ed., 2001.

[13] J. Mohr and X. Li, “Computational challenges in multiple wavetable interpolation synthesis,” in *Computational Science—ICCS 2003, International Conference, Melbourne, Australia and St. Petersburg, Russia, June 2–4, 2003* (P. M. A. Sloot *et al.*, eds.), no. 2657 in Lecture Notes in Computer Science, pp. 447–456, Springer-Verlag, 2003.

[14] F. Opolko and J. Wapnick, “McGill University Master Samples.” 11-volume set of audio CD’s, 1987. Available from McGill University, Montreal, via `http://www.music.mcgill.ca/-resources/mums/html/index.htm`. (2004-10-31).

[15] J. W. Beauchamp, “Unix workstation software for analysis, graphics, modifications, and synthesis of musical sounds,” in *Ninety-fourth Convention of the Audio Engineering Society*, (Berlin), Audio Engineering Society, New York, 1993. Preprint 3479.

[16] J. M. Grey and J. A. Moorer, “Perceptual evaluations of synthesized musical instrument tones,” *Journal of the Acoustical Society of America*, vol. 62, pp. 454–462, Aug. 1977.

[17] K. Jensen, “The timbre model,” in *Proceedings of the Workshop on Current Research Directions in Computer Music*, (Barcelona), MOSART, Nov. 15–17 2001. Available online at `http://www.iua.upf.es/mtg/-mosart/papers/p08.pdf` (2004-10-31).

[18] J. Mohr, “Optimized multiple wavetable analysis/synthesis: Results in detail,” Tech. Rep. TR02-19, University of Alberta, Edmonton, Alberta, Canada, Sept. 2002. Available online at `http:-//www.cs.ualberta.ca/cgi-bin/-techreport.cgi?action:menu` (2004-10-31).

[19] K. Lee and A. Horner, “Modeling piano tones with group synthesis,” *Journal of the Audio Engineering Society*, vol. 47, pp. 101–111, Mar. 1999.

[20] C. So and A. B. Horner, “Wavetable matching of pitched inharmonic instrument tones,” *Journal of the Audio Engineering Society*, vol. 52, pp. 516–529, May 2004.

[21] X. Serra and J. Smith III, “Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition,” *Computer Music Journal*, vol. 14, pp. 12–24, Winter 1990.

[22] T. S. Verma and T. H. Y. Meng, “Extending spectral modeling synthesis with transient modeling synthesis,” *Computer Music Journal*, vol. 24, pp. 47–59, Summer 2000.

[23] C.-W. Wun and A. Horner, “Perceptual wavetable matching for synthesis of musical instrument tones,” *Journal of the Audio Engineering Society*, vol. 49, pp. 250–261, Apr. 2001.