Audio Time-Scale Modification Using a Hybrid Time-Frequency Domain Approach

David Dorran, Eugene Coyle

Digital Media Centre
Dublin Institute of Technology
Aungier Street, Dublin 8, Ireland
david.dorran@dit.ie

Robert Lawlor

Department of Electronic Engineering
National Univeristy of Ireland
Maynooth, Co. Kildare, Ireland
rlawlor@eeng.may.ie

ABSTRACT

Frequency-domain approaches to audio time-scale modification introduce a reverberant artifact into the time-scaled output due to a loss in phase coherence between subband components. Whilst techniques have been developed which reduce the presence of this artifact, it remains a source of difficulty. A method of time-scaling is presented that reduces the presence of reverberation by taking advantage of some flexibility that exists in the choice of phase required so as to maintain horizontal phase coherence along frequency-domain subband components. The approach makes use of appealing aspects of existing time-domain and frequency-domain time-scaling techniques.

1. INTRODUCTION

Time-scale modification of audio alters the duration of an audio signal whilst retaining the signals local frequency content, resulting in the overall effect of speeding up or slowing down the perceived playback rate of a recorded audio signal without affecting its perceived pitch or timbre.

There are two broad approaches used to achieve a time-scaling effect i.e. time-domain and frequency-domain. Time-domain algorithms, such as the synchronized overlap-add (SOLA) algorithm [1], are generally more efficient than their frequency-domain counterparts, but require the existence of a strong quasi-periodic element within the signal to be time-scaled in order to produce a high quality output. This makes them generally unsuitable for their application to complex audio such as multi-pitched polyphonic music. Frequency-domain techniques, such as the phase vocoder [2] and sinusoidal modelling [3], are capable of time-scaling complex audio but introduce a reverberant/phasy artifact into the time-scaled output. This artifact is generally more objectionable in speech than in music; since music recordings typically contain a significantly higher level of reverberation than speech so that additional reverberation introduced by time-scaling is not as noticeable.

This paper presents a hybrid time-frequency domain algorithm that takes advantage of certain aspects of each broad approach to realize an efficient and robust time-scaling implementation, which reduces the presence of the phasiness artifact associated with frequency-domain implementations.

This paper is structured as follows: Section 2 provides an overview of SOLA; Section 3 outlines the basic operation of the improved phase vocoder [4], which makes use of sinusoidal modeling techniques to improve upon the standard phase vocoder; Section 4 discusses the phase tolerance allowed within phase vocoder implementations [5] and demonstrates how this tolerance can be used to push/pull phases back into a phase coherent state; Section 5 describes the hybrid approach which incorporates both time-domain and frequency-domain features through manipulation of the phase tolerance identified; Section 6 concludes.

2. SYNCHRONIZED OVERLAP-ADD

Time-domain algorithms operate by appropriately discarding or repeating suitable segments of the input; with the duration of these segments being typically an integer multiple of the local pitch period (when it exists). Time-domain techniques are capable of producing a very high quality output when dealing with quasi periodic signals, such as speech, but have difficulty with more complex audio, such as multi-pitched polyphonic audio [6]. It should be noted that fewer discard/repeat segments are required the closer the desired time-scale duration is to that of the original duration [6]. Therefore time-domain algorithms produce particularly high quality results for time-scale factors close to one, since significant portions of the output are directly copied, without processing, from the input.

The SOLA algorithm achieves the discard/repeat process by first segmenting the input into overlapping frames, of length N, with each frame S_a samples apart. S_a is the analysis step size. The time-scaled output y is synthesized by overlapping successive frames with each frame a distance of $S_s + \tau_m$ samples apart. S_s is the synthesis step size, and is related to S_a by $S_s = \alpha S_a$, where α is the time scaling factor. τ_m is a offset that ensures that successive synthesis frames overlap synchronously. Figure 1 illustrates an iteration of this process, whereby an input frame is appended to the current output.

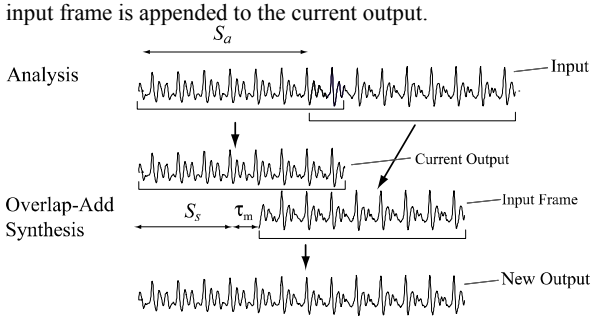

Figure 1: *SOLA iteration*

Standard SOLA parameters are generally fixed, however in [7] an adaptive and efficient parameter set is derived, which is used in the hybrid implementation (section 5) and is given by

$$S_a = \frac{L_{stat} - SR}{|1-\alpha|} \quad (1)$$

$$N = SR + \alpha\left(\frac{L_{stat} - SR}{|1-\alpha|}\right) \quad (2)$$

where L_{stat} is the stationary length (approx 25-30ms) and SR is the search range over which τ_m is determined (approx 12-20ms).

3. IMPROVED PHASE VOCODER

Time-domain techniques maintain 'horizontal' synchronization between successive frames by determining regions of similarity between the frames prior to overlap-adding; as such, time-domain techniques require the input to be suitably periodic in nature. Phase vocoder implementations operate by maintaining 'horizontal' synchronization along subbands; such an approach removes the necessity for a quasi-periodic broadband signal.

Within phase vocoder implementations it is assumed that each subband contains a quasi-sinusoidal component [2]. Standard implementations of the phase vocoder make use of uniform width filterbanks to extract the quasi-sinusoidal subbands, typically through the efficient use of a short-time Fourier transform (STFT).

Horizontal synchronization (or horizontal phase coherence [4]) is maintained at a subband level by ensuring that the expected phase of each sinusoidal component follows the sinusoidal phase propagation rule i.e.

$$\varphi_2 = \varphi_1 + \omega(t_2 - t_1) \quad (3)$$

where φ_1 is the instantaneous phase at time t_1, ω is the frequency of the sinusoidal component, and φ_2 is the expected phase of the sinusoidal component at time t_2.

During time-scale modification magnitude values of the sinusoidal subband components are simply interpolated or decimated to the desired duration. In [8] time-scale expansion is achieved by appropriately repeating STFT windows e.g. to time-scale by a factor of 1.5 every second window is repeated; similarly time-scale compression is achieved by omitting windows e.g. to time scale by a factor of 0.9 every tenth analysis window is omitted. The phase propagation formula of equation (3) is then applied to each subband (or discrete Fourier Transform (DFT) bin), from window to window.

In [4] it is recognized that not all subbands are true sinusoidal components, and some are essentially 'interference' terms introduced by the windowing process of the STFT analysis. [4] notes that applying the phase propagation rule to these interference terms results in a loss of 'vertical phase coherence' between subbands which introduces a reverberant or phasy artifact into the time-scaled output. The solution to this problem is to identify 'true' sinusoidal components through a magnitude spectrum peak peaking procedure and applying the phase propagation rule to these components only. The phases of the subband components in the 'region of influence' of a peak/sinusoidal subband are updated in such a manner as to preserve the original phase relationships [4].

Whilst [4] results in improved vertical phase coherence between a true sinusoidal component and its neighboring interference components, it does not attempt to maintain the original phase relationships that exist between true sinusoidal components. The loss of phase coherence between these components also results in the introduction of reverberation. This problem is addressed in the literature, whereby the phase relationship or 'relative phase difference' between harmonically related components of a harmonic signal is maintained through various techniques e.g. [9-11]. These approaches, however, require the determination of the local pitch period. Whilst the techniques of [9-11] attempt to maintain vertical phase coherence through the manipulation of the phase values of harmonically related sinusoidal components, time-domain approaches implicitly maintain vertical phase coherence by virtue of the fact that the broadband signal is not partitioned into subbands.

4. PHASE FLEXIBILITY WITHIN PHASE VOCODER

In [5] it is shown that displacing the horizontal phase of a pure sinusoidal component from its ideal/expected value, within a window of the phase vocoder, results in a certain amount of amplitude and frequency modulation being introduced into the sinusoidal component. Furthermore, in [5] it is shown, through a psychoacoustic analysis, that if the phase deviation introduced is less than a particular value, the amplitude and frequency modulations will not be perceived. The phase deviation that is 'perceptually tolerated' is dependent on the hop size and window length of the STFT. From [5] the maximum phase deviation tolerated θ for a 50% analysis window overlap is:

$$\theta = \min\{0.5676,\ 2\arctan(3.6L)\} \text{ radians} \quad (4)$$

where L is the duration of the analysis window in seconds.

The workings for the derivation of equivalent equations for a 75% overlap are somewhat verbose and can be determined in a similar manner to the methodology outlined in [5]. For the sake of convenience the equations derived for a 75% overlap are provided here. The maximum phase deviation tolerated θ is given by

$$\theta = \min\{0.27,\ 2\arcsin(2.53L)\} \text{ radians} \quad (5)$$

It should be noted that (5) is an approximation, valid within 0.2% for values of θ less than 0.27 radians.

[5] also shows how the phase tolerance can be used to push or pull a modified STFT representation into a phase coherent state; the basic principle is briefly explained as follows:

Consider the situation illustrated in Figure 2; assume that the phases of synthesis window 1' are equal to those of analysis window 1; the phases of the repeated synthesis window 2' are then determined such that horizontal phase coherence is maintained between true sinusoidal components (peaks), whilst phases of neighboring components are updated so as to maintain vertical phase coherence. Horizontal phase coherence between the peaks of synthesis windows 1' and 2' can be preserved by keeping the same phase difference between them that exists between analysis windows 1 and 2 [8]; then

synthesis window 1' comprises of the magnitudes and phases of analysis window 1 (and is therefore perfectly phase coherent), whilst synthesis window 2' comprises of the magnitudes of analysis window 1 and a set of phases close to those of analysis window 2 (and is therefore generally not perfectly phase coherent). It follows that, in general, synthesis window *n*' comprises of the magnitudes of analysis window *n*-1 and phases close to those of analysis window *n*, for all windows up to the next discard/repeat frame.

In [5] the synthesis phase values of synthesis window *n*' are pushed or pulled toward the phase values of analysis window *n*-1 using the horizontal phase tolerance established. Once the phases of window *n*' equal those of the target phases of analysis window *n*-1 perfect phase coherence is restored. It follows that subsequent windows up to the next discard/repeat window will also be perfectly phase coherent. From Figure 2, once phase coherence is realized (at synthesis window 7' in Figure 2), there is no need for further frequency-domain processing and a segment of the original time-domain input can be simply inserted into the output, in a similar manner to time-domain implementations, as shown in Figure 2. This has the added benefit of reducing the computational costs whilst bringing the time-scaled output into a phase coherent state.

This process requires that a certain number of windows exist before the next discard/repeat operation; for example given a phase tolerance of 0.314 (i.e. π/10) radians, perfect phase coherence is assured to be established for time-scale factors between 0.9 and 1.1, since phase values can be at most +/-π radians from perfect phase coherence. It should be noted that if the phase values of synthesis window 2' were close to those of analysis window 1 then perfect phase coherence would be established quickly; the following section addresses this issue by making use of time-domain techniques in identifying 'good' initial phase values, thereby reducing the transition time to perfect phase coherence.

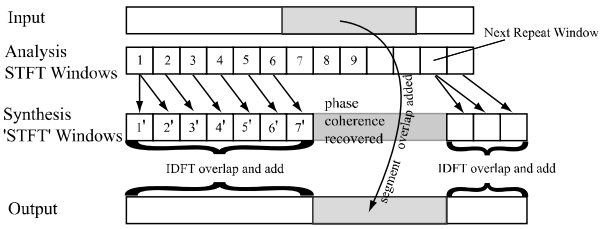

Figure 2: *Time-scaling process*

5. HYBRID IMPLEMENTATION

The original motivation behind the SOLA algorithm [1] was to provide an initial set of phase estimates for the reconstruction of a magnitude only STFT representation of a signal. The same principle is used here to provide a set of phase estimates for use within the procedure outlined in section 4. The remainder of this section describes the approach used to determine the initial phase estimates and their use within the hybrid implementation.

Consider the situation shown in Figure 3, in which a frame extracted from the input is shown overlapping with the current output. As with the standard SOLA implementation the overlap shown is determined through the use of a correlation function. For the m^{th} iteration of the algorithm the offset τ_m is chosen such that the correlation function $R_m(\tau)$, given by

$$R_m(\tau) = \frac{\sum_{j=0}^{L_m-1} y(mS_s + \tau + j)x(mS_a + j)}{\sqrt{\sum_{j=0}^{L_m-1} x^2(mS_a + j)\sum_{j=0}^{L_m-1} y^2(mS_s + \tau + j)}} \quad (6)$$

is a maximum for $\tau = \tau_m$, where x is the input signal, y is the time-scaled output, L_m is the length of the overlapping region and τ is in the range $0 < \tau < \tau_{max}$, where τ_{max} is typically the number of samples which equates to approximately 20ms. S_a and S_s are defined in section 2. The optimum frame overlap L_{ov} shown in Figure 3 is then given by

$$L_{ov} = N - S_s - \tau_m \quad (7)$$

where N is the frame length, defined in section 2.

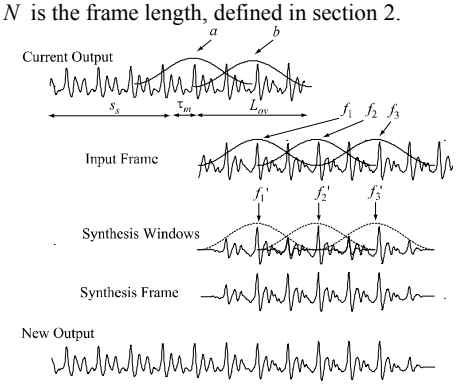

Figure 3: *Hybrid iteration*

Also shown in Figure 3 below the input frame, are the synthesis windows and the synthesis frame; it is this synthesis frame which is appended to the current output within the hybrid approach and not the input frame, as is the case in SOLA. The following details the generation of the synthesis frame.

Window b is first extracted from the output y and is positioned such that it has its center at the center of the 'optimum' overlap, as shown in the diagram. More specifically, for the m^{th} iteration of the algorithm, frame b is given by

$$b(j) = y(mS_s + \tau_m + L_{ov}/2 - L/2 + j).w(j) \text{ for } 0 < j \le L \quad (8)$$

where w is the STFT analysis window, typically hanning, L is the STFT window length, typically the number of samples which equates to approximately 60ms. (Both shorter and longer windows have been proposed in the literature, however 60ms was found to be suitable for an implementation which is intended to cater for both speech and a wide range of polyphonic music.)

The window f_1 is extracted from the input x and is positioned such that it is aligned with frame b. Subsequent windows are sequentially spaced by the STFT hop size H. More specifically, for the m^{th} iteration of the algorithm window f_n is given by

$$f_n(j) = x(mS_a + L_{ov}/2 + H.(n-1) - L/2 + j).w(j) \text{ for } 0 < j \le L \quad (9)$$

F_1' the DFT representation of f_1', is then derived using the magnitudes of F_1 and the phase values B, where F_n and B are the DFT representations of f_n and b, respectively; then

$$F_1'(k) = |F_1(k)| \exp(i \angle B(k)) \text{ for all } k \text{ in the set } P_1 \quad (10)$$

where P_1 is the set of peak bins found in $|F_1|$. All other bins are updated so as to maintain the original phase difference between a peak and bins in its region of influence, as described in [4]. The phase values of STFT window B are chosen since they provide a set of phase values that naturally follow the window labeled a in Figure 3 and therefore maintain horizontal phase coherence. Subsequent synthesis windows are derived from

$$F_n'(k) = |\angle F_n(k)| \exp\left(i\left(\angle F_{n-1}'(k) + \angle F_n(k) - \angle F_{n-1}(k) + D(k)\right)\right) \quad (11)$$

for all k in the set P_n, where P_n is the set of peak bins found in $|F_n|$. As above, all other bins are updated so as to maintain the original phase difference between a peak and bins in its region of influence. For the hybrid case perfect phase coherence is achieved when synthesis STFT window F_n' has the magnitude and phase values of window F_n. D is the phase deviation which is used to push or pull the frames into a phase coherent state. D is dependent on the bin number denoted by k and is given by

$$D(k) = \angle F_{n-1}(k) - \angle F_{n-1}'(k) \quad \text{if princarg}\left(\angle F_{n-1}(k) - \angle F_{n-1}'(k)\right) \le \theta \quad (12)$$

or

$$D(k) = sign\left(\angle F_{n-1}(k) - \angle F_{n-1}'(k)\right)\theta \quad \text{if princarg}\left(\angle F_{n-1}(k) - \angle F_{n-1}'(k)\right) > \theta \quad (13)$$

where θ is the maximum phase tolerance (see section 4).

The number of synthesis STFT windows required is such that an inverse STFT on these windows results in a synthesis frame of duration N+3L/2. This is to ensure that window b is available for the next iteration of the algorithm. It should be noted that the number of the synthesis windows also controls the ability of the algorithm to recover phase coherence; if N is large (which is the case when is α is close to one, see equation (2)) phase coherence is recovered more easily. The synthesis frame x_m is obtained through the application of an inverse STFT on windows F_1', F_2', F_3',…. The output y is then updated by

$$y(mS_s + \tau_m + L_{ov}/2 - L/2 + j) := E(j).y(mS_s + \tau_m + L_{ov}/2 - L/2 + j) + x_m(j) \text{ for } 0 < j \le L-H \quad (14)$$

$$y(mS_s + \tau_m + L_{ov}/2 - L/2 + j) = x_m(j) \text{ for } L-H < j \le N+3L/2 \quad (15)$$

where := in equation (14) means 'becomes equal to' and E is an envelope function which ensures that the output y sums to a constant during the overlap-add procedure.

E is dependent on the STFT hop size H and whether a synthesis window is employed during the inverse STFT procedure. For the case where a synthesis window is employed, which is equal to the analysis hanning window w, and $H = L/4$

$$E(j) = w^2(H + j) + w^2(2H + j) + w^2(3H + j) \text{ for } 0 < j \le L-H \quad (16)$$

It should be noted that for the case where the input is perfectly periodic the initial phase estimates provided by STFT window B are assured to be equal to the target phase values of window F_1 and the time-scaled output is always perfectly phase coherent. For quasi-periodic signals, such as speech, the initial phase estimates are generally close to the target phase, and the transition period to perfect phase coherence is generally short.

For the case where more complex audio is being time-scaled, the transition to perfect phase coherence is relatively long; nevertheless, the reverberant artifact introduced, due to the loss of perfect phase coherence, is perceptually less objectionable in these types of signals, due to the reverberation level generally already present. The hybrid approach described does, however, have the benefit of noticeably reducing the effects of transient smearing without the necessity of explicit transient detection.

As with time-domain implementations, the quality and efficiency improvements offered by the hybrid approach over frequency-domain approaches are most noticeable for time-scaling factors close to one, with results being particularly good for factors in the range 0.8 to 1.2.

6. CONCLUSIONS

A hybrid time-scaling algorithm is presented which draws on the best features of time-domain and frequency-domain implementations. The novel approach reduces the presence of the reverberant artifact associated with frequency-domain techniques without the requirement of explicit pitch detection; the algorithm is also capable of preserving transients without explicit transient detection. The improvements provided by the approach are most noticeable for time-scale factors close to one (0.8-1.2). The algorithm is both robust and efficient and produces very high quality results for both speech and a wide range of polyphonic audio.

7. REFERENCES

[1] S. Roucos, A.M. Wilgus, "High quality time-scale modification for speech," *IEEE Int' conf' on Acoustics, Speech and Signal processing*, pp. 493-496, '85.

[2] M. Dolson, "The phase vocoder: A tutorial," *Computer Music Journal*, vol. 10, pp. 145-27, '86.

[3] R. McAulay, T Quatieri, "Speech analysis/synthesis based on a sinusoidal representation", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 34(4),pp.744–754, '86.

[4] J. Laroche, M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Transactions on Speech and Audio Processing*, vol. 7(3), pp. 323-332, '99.

[5] D. Dorran, R. Lawlor, E. Coyle, "An efficient phasiness reduction technique for moderate audio time-scale modification," *Proceedings of DAFX-04*, pp. 83-88, '04.

[6] J. Laroche, "Autocorrelation method for high-quality time/pitch-scaling", *IEEE Workshop on App's of Signal Processing to Audio and Acoustics*, pp. 131 – 134, '93.

[7] D. Dorran, R. Lawlor, "An efficient time-scale modification algorithm for use within a subband implementation," *Proc. of DAFX-03*, pp. 339-343, '03.

[8] J. Bonada, "Automatic technique in frequency domain for near-lossless time-scale modification of audio," *Proc. of International Computer Music Conference*, '00.

[9] T. Quatieri, R. McAulay, "Shape invariant time-scale and pitch-scale modification of speech," *IEEE Transactions on Signal Processing*, vol. 40(3), pp 497-510, '92.

[10] R. Di Federico, "Waveform preserving time stretching and pitch shifting for sinusoidal models of sound," *Proc. of DAFX-98,* pp. 44-48, '98.

[11] J. Laroche, "Frequency-domain techniques for high quality voice modification," *Proc. of DAFX-03,* pp.328-322, '03.