2006-01-01

A Comparison of Time-domain Time-scale Modification Algorithms

David Dorran
Technological University Dublin, david.dorran@tudublin.ie

Robert Lawlor
National University of Ireland, Maynooth

Eugene Coyle
Technological University Dublin, Eugene.Coyle@tudublin.ie

Recommended Citation

Dorran, D., Lawlor, R. & Coyle E. (2006) A comparison of time-domain time-scale modification algorithms. *120th. Audio Engineering Society, Paris, France, May 20-23, 2006.*

Audio Research Group

Articles

Dublin Institute of Technology *Year* 2006

A Comparison of Time-domain Time-scale Modification Algorithms

David Dorran* Robert Lawlor†

Eugene Coyle‡

*Dublin Institute of Technology, david.dorran@dit.ie

†National University of Ireland, Maynooth

‡Dublin Institute of Technology, Eugene.Coyle@dit.ie

— Use Licence —

Attribution-NonCommercial-ShareAlike 1.0

You are free:

- to copy, distribute, display, and perform the work
- to make derivative works

Under the following conditions:

- Attribution.
  You must give the original author credit.
- Non-Commercial.
  You may not use this work for commercial purposes.
- Share Alike.
  If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

For any reuse or distribution, you must make clear to others the license terms of this work. Any of these conditions can be waived if you get permission from the author.

Your fair use and other rights are in no way affected by the above.

Audio Engineering Society

Convention Paper

Presented at the 120th Convention
2006 May 20–23 Paris, France

A Comparison of Time-Domain Time-Scale Modification Algorithms

David Dorran[1], Robert Lawlor[2], and Eugene Coyle[3]

[1] Digital Audio Research Group, Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland.
david.dorran@dit.ie

[2] Department of Electronic Engineering, Nation University of Ireland, Maynooth, Co. Kildare, Ireland.
rlawlor@eeng.may.ie

[3] Digital Audio Research Group, Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland.
eugene.coyle@dit.ie

ABSTRACT

Time-domain approaches to time-scale modification are popular due to their ability to produce high quality results at a relatively low computational cost. Within the category of time-domain implementations quite a number of alternatives exist, each with their own computational requirements and associated output quality. This paper provides a computational and objective output quality assessment of a number of popular time-domain time-scaling implementations; thus providing a means for developers to identify a suitable algorithm for their application of interest. In addition, the issues that should be considered in developing time-domain algorithms are outlined, purely in the context of a waveform editing procedure.

1. INTRODUCTION

Time-scale modification of audio alters the duration of an audio signal while 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 the pitch or timbre of the original signal. In other words, the duration of the original signal is increased or decreased but the perceptually important features of the original signal remain unchanged; for the case of speech, the time-scaled signal sounds as if the original speaker has spoken at a quicker or slower rate; for the case of music, the time-scaled signal sounds as if the musicians have played at a different tempo.

Transforming audio to an alternative time-scale is a popular and useful digital audio effect that has become a standard tool within many audio multi-processing applications. Some particular uses of this effect are:

- Fast browsing of speech material for digital libraries and distance learning [1].
- Music and foreign language learning/teaching [2], [3], [4].
- Fast/slow playback for telephone answering machines and dictaphones [5].
- Video-cinema standards conversion [6].
- Audio Watermarking [7].
- Accelerated aural reading for the blind [8].
- Music composition [9].
- Audio-video synchronization [10].
- Audio data compression [11], [12].
- Diagnosis of cardiac disorders [13].
- Editing audio/visual recordings for allocated time-slots within the radio/television industry [14].
- Voice gender conversion [15].
- Text-to-speech synthesis [16], [17].
- Lip synchronization and voice dubbing [18].
- Prosody transplantation and karaoke [18].

The main considerations in choosing a time-scaling algorithm are the quality of output produced and the efficiency of the algorithm. Time-scale modification techniques can be broadly categorised into time-domain and frequency domain approaches, with those operating in the time-domain being, in general, more efficient. For quasi-periodic signals, such as speech and monophonic music, the efficiency provided by time-domain algorithms does not result in a lesser quality output, making time-domain techniques the algorithms of choice within predominantly speech processing applications, or when other quasi-periodic signals are being processed. Within the category of time-domain techniques, a variety of implementations exist; however a comparison of these approaches in terms of computational requirements and output quality has not yet been provided, making it difficult for developers to choose the most appropriate algorithm for their application of interest.

Time-domain implementations operate by discarding/repeating suitable segments of the input; this process requires the use of a synchronization procedure, which is generally the most significant drain on processing resources. The central contribution of this paper is the provision of a computational load and output quality comparison of a number of commonly used synchronization procedures, thus allowing developers readily identify the synchronization procedure most suitable for their requirements.

A previous tutorial article on time-domain time-scaling [19] presents the topic within the context of a frequency-domain analysis. In a second contribution, this paper presents an overview entirely within the framework of a less complex waveform editing procedure. This approach leads to an intuitively appealing discussion on the principal issues involved, resulting in an incisive understanding of the effects that the various parameters associated with time-domain techniques have.

This paper is organized as follows: section 2 provides a brief overview of time-domain implementations and the issues arising from their implementation; section 3 takes a closer look at the popular synchronized overlap-add (SOLA) algorithm [20] and the effect the choice of parameters within SOLA based implementations have; in section 4 a number of alternative synchronization procedures are described together with comprehensive analysis of their computational requirements; section 5 presents a summary of the computational requirements of each synchronization procedure within a SOLA based implementation; section 6 describes a number of objective output quality measures and presents the results of these measures as they were applied to each synchronization procedure; sections 7 and 8 provide a discussion of results and conclusion, respectively.

2. OVERVIEW

In the earliest digital implementations, e.g. [21], the input is first segmented into non-overlapping frames (typically 20-30ms in duration) and appropriate frames are then discarded/repeated in order to achieve the desired time-scaling. Similar electro-mechanical approaches are described in [22] and [23]. This approach is commonly referred to as 'cut and splice' and its concept is illustrated in figure 1, where (a) represents the audio input that has been segmented into non-overlapping frames; (b) represents a 50% time-scale compressed version of the input; and (c) represents a 133% time-scale expanded version of the input. In general, to achieve the desired time-scale expansion/compression, frames labelled $round(m/|1-\alpha|)$ are repeated/discarded, where m is a set of consecutive

integers and α is the desired time-scaling factor e.g. α = 2 corresponds to a 200% time-scale expansion and α = 0.33 corresponds to a 33% time-scale compression.

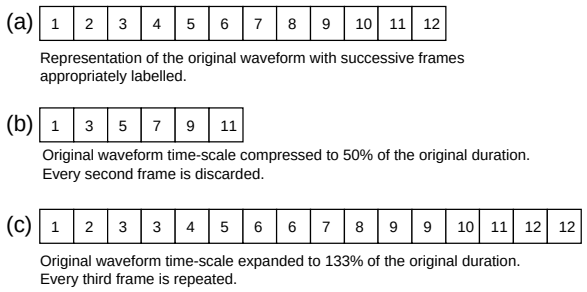

Figure 1 'Cut and splice' time-scale compression and expansion

While the process described above is efficient and relatively straightforward to implement, it does, however, introduce artefacts into the time-scaled output. These artefacts are the result of discontinuities and pitch distortions [21], and their origins can be understood by considering the example shown in figure 2. As can be seen from the figure, the simple repetition of a frame can result in a discontinuity in the synthesized waveform together with some distortion of the pitch, which results in objectionable artefacts being perceived. One method of reducing the effects of the discontinuity is to gradually cross-fade segments together rather than simply appending synthesis frames in a hard splice manner [21], however this technique has a limited effect, since pitch distortions remain, suggesting that more intelligent methods of frame repetition are required.

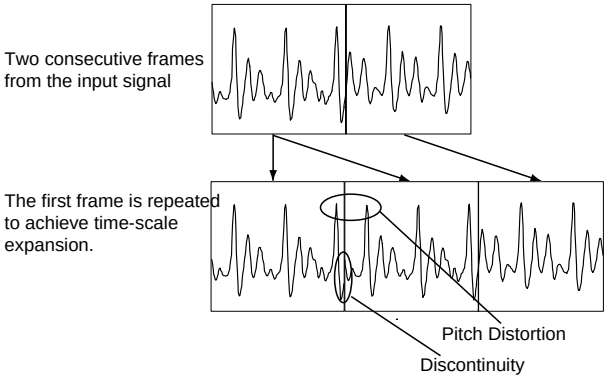

Figure 2 Artefacts arising from 'cut and splice' implementations.

A solution to these problems is proposed in [20], whereby the artefacts introduced by discarding/repeating frames are significantly reduced by overlapping synthesis frames in regions of similarity. As an example, consider the case illustrated in figure 3, i.e. a re-examination of figure 2; by overlapping the repeating frame in a synchronous manner i.e. in a region where the frames are similar, the effects of discontinuities and pitch distortion are removed. This process essentially equates to discarding/repeating segments of the input that are integer multiples of local pitch periods in length. Publications previous to [20] also suggest a pitch synchronous approach [24], [25], [26] and [27]; however, the procedure described in [20] provides the basis for a robust implementation.

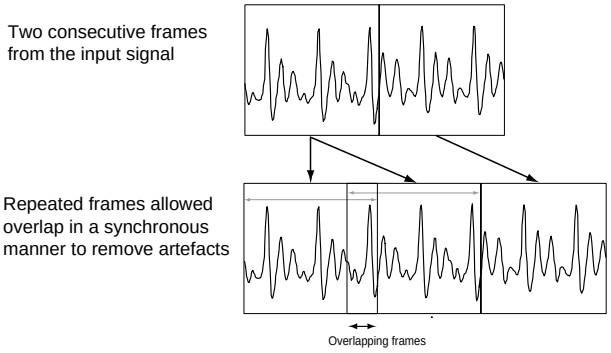

Figure 3 Synchronised overlap removes artefacts associated with 'cut and splice' methods.

3. AN ANALYSIS OF SOLA BASED IMPLEMENTATIONS

The synchronised overlap-add algorithm segments the input signal x into m overlapping frames, of length N samples, each segment being 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 $S_s + k_m - k_{m-1}$ 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. k_m is an offset that ensures that successive synthesis frames overlap in a synchronous manner. k_m is chosen such that

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} y(mS_s + k + j)x(mS_a + j)}{\sqrt{\sum_{j=0}^{L_k-1} x^2(mS_a + j)\sum_{j=0}^{L_k-1} y^2(mS_s + k + j)}} \quad (1a)$$

is a maximum for $k = k_m$, where m represents the m^{th} input frame and L_k is the length of the overlapping region i.e.

$$L_k = N - S_s + k_{m-1} - k \quad (1b)$$

k is in the range $k_{min} \leq k \leq k_{max}$.
$R_m(k)$ is a correlation function which ensures that successive synthesis frames overlap at the 'best' location i.e. that location where the overlapping frames are most similar. Having located the 'best' position at which to overlap, the overlapping regions of the frames are weighted prior to combination. This is generally achieved using a linear or raised-cosine cross-fading function. The output is then given by

$$y(mS_s + k_m + j) := (1 - f(j))y(mS_s + k_m + j) + f(j)x(mS_a + j), 0 \leq j \leq L_k - 1 \quad (2a)$$

$$y(mS_s + k_m + j) = x(mS_a + j),\ L_k \leq j \leq N - 1 \quad (2b)$$

where := in equation (2a) means 'becomes equal to' and $f(j)$ is a weighting function such that $0 \leq f(j) \leq 1$.
A linear weighting function can be expressed as

$$f(j) = 0, j < 0 \quad (3a)$$

$$f(j) = j / (L_k - 1),\ 0 \leq j \leq L_k - 1 \quad (3b)$$

$$f(j) = 1, j > L_k - 1 \quad (3c)$$

3.1. Choice of SOLA Parameters

In early implementations, typically, N is fixed at 30ms for speech and 40ms for music, S_a is $N/2$, k_{min} is $-N/2$ and k_{max} is $N/2$. However, the use of fixed parameters can lead to inefficiency and in some cases result in a poor quality output [5], [28]. Many alternative values for SOLA's parameters are suggested, for example in [29], [30] and [10], however the motivation behind the choice of these parameters is unclear. An understanding of the effects of the various parameters is obtained through an examination of some particular situations. First consider the case where a perfectly periodic signal, of period P, is being time-scaled and two frames of the input are being overlapped. In figure 4(a) if the synthesis frames overlap is allowed vary from P to 1, the correlation function, graphed to the right of the overlapping frames, produces a single maximum, corresponding to the overlapping region of maximal similarity. Allowing frames overlap in the range P to 1 ensures that a maximum correlation will occur, however, consider the case of figure 4(b); if these frames are allowed overlap from P to 1 an unsuitable overlap may be returned due to 'ambiguous' maxima being returned by the correlation function, as shown in the figure. In general, for perfectly periodic signals, to remove the risk of ambiguous results being returned the overlap should be allowed vary from $2P$ to P, as demonstrated in figure 4(c). It should be noted, however, that k is still in the range $0 \leq k \leq P$.

Typically the voiced/quasi-periodic regions of a speech signal have a waveform similar to that of figure 4 (a) and this constraint can be somewhat relaxed since potential ambiguities generally arise in the 'lower amplitude' section of a period of the waveform. It should also be noted that the pitch of an audio signal changes frequently and that P should be chosen so as to equate to the longest likely pitch period of the signal being analysed (typically 10ms). For these reasons, allowing the synthesis overlap to vary from $3P/2$ to $P/2$ will produce adequate results for speech signals.

In order to allow the synthesis overlaps vary from $2P$ to P (or $3P/2$ to $P/2$) the difference between k_{max} and k_{min} should be set equal to P i.e.

$$k_{max} - k_{min} = P \quad (4)$$

If k_{min} is set to zero then k_{max} becomes P.

The next constraint is to ensure the initial synthesis overlap is $2P$ (or $3P/2$ for a more efficient and generally adequate implementation). For convenience the initial synthesis overlap is labelled L_{max}. The length of L_{max} is also constrained by equation (1b) when k is set to its minimum i.e. 0, therefore

$$L_{max} = N - S_s + k_{m-1} \quad (5)$$

If the output, y, is truncated to $mS_s + N$ samples after each iteration then L_{max} becomes independent of the previous synthesis offset and is given by

$$L_{max} = N - S_s \quad (6)$$

Truncating the output also has the effect of altering equation (1b), which becomes

$$L_k = N - S_s - k \quad (7)$$

Since S_s is constrained to be αS_a, there is only one 'unknown' parameter, i.e. S_a, and an analysis similar to that given in [31] is now performed in order to determine a suitable setting for S_a.

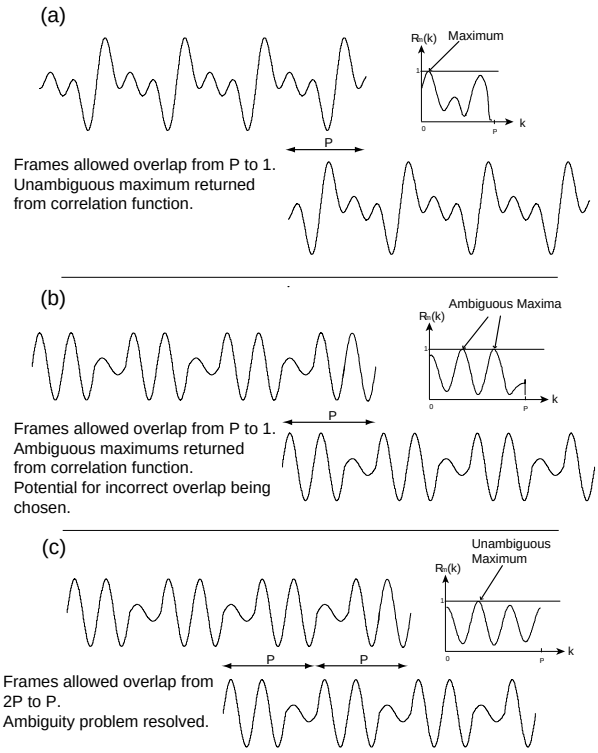

Figure 4 Scenarios involving overlapping synthesis frames

From the description of SOLA given at the start of section 3, the distance between successive synthesis frames is given by $S_s+k_m-k_{m-1}$. The length of the segment discarded/repeated during an iteration of the algorithm is then given by $|S_a - (S_s+k_m-k_{m-1})|$. Consider the case where $k_{m-1} = k_{max} = P$ and $k_m = k_{min} = 0$ i.e. maximum overlap; then a segment of length $|S_a - (S_s - P)|$ is discarded/repeated during the overlap-add process. For high quality time-scale modification the discarded/repeated segment should be short enough to ensure quasi-stationarity during voiced regions, so

$$|S_a-(S_s-P)|\leq L_{stat} \tag{8}$$

where L_{stat} is the duration over which the input is quasi-stationary. Since $S_s = \alpha S_a$

$$|(1-\alpha)S_a+P| \leq L_{stat} \tag{9}$$

Also, $|(1-\alpha)S_a+P|$ is a maximum when $\alpha < 1$, therefore equation (9) should be satisfied when $\alpha < 1$. Since

$$|(1-\alpha)S_a+P|=(1-\alpha)S_a+P \text{ when } \alpha<1 \tag{10}$$

then

$$S_a \leq \frac{L_{stat}-P}{1-\alpha} \quad \text{for } \alpha<1 \tag{11}$$

Now consider the case when $k_{m-1} = k_{min} = 0$ and $k_m = k_{max} = P$ i.e. minimum overlap; then a segment of length $|(1-\alpha)S_a - P|$ is discarded/repeated. As in the case described above, the discarded/repeated segment should be short enough to ensure quasi-stationarity during voiced regions, so

$$|(1-\alpha)S_a-P|\leq L_{stat} \tag{12}$$

$|(1-\alpha)S_a - P|$ is a maximum when $\alpha > 1$, therefore equation (12) should be satisfied when $\alpha > 1$. Since

$$|(1-\alpha)S_a-P|=P-(1-\alpha)S_a \text{ when } \alpha>1 \tag{13}$$

then

$$S_a \leq \frac{L_{stat}-P}{\alpha-1} \quad \text{for } \alpha>1 \tag{14}$$

To achieve high quality time-scale modification equations (11) and (14) must simply be satisfied, however, the number of iterations that are executed is inversely proportional to S_a, therefore S_a should be maximised for the purpose of computational efficiency giving

$$S_a = \frac{L_{stat}-P}{|1-\alpha|} \quad \text{for all } \alpha \tag{15}$$

And since $N = L_{max} + \alpha S_a$, from (6),

$$N = L_{\max} + \alpha\left(\frac{L_{stat}-P}{|1-\alpha|}\right) \text{ for all } \alpha \tag{16}$$

A quasi-stationary segment is a segment in which the frequency content is approximately constant throughout the entire duration of the segment. The duration of stationary segments within an audio input is constantly changing for most naturally occurring sounds. The choice of L_{stat} within the SOLA based implementation described above has a significant effect on both the quality of output and the number of computations required; choosing too small a value results in possibly too many iterations of the synchronization procedure; choosing too large a value results in inappropriate segments being discarded/repeated. For the general

case, where the same parameters are being applied along the entire duration of an input signal, the maximum segment discarded/repeated is L_{stat} and the minimum is $L_{stat} - P$. To ensure the algorithm operates as expected L_{stat} cannot be less than P, since this would suggest that a single period of the lowest likely dominant frequency component could not be discarded/repeated. For typical speech it is found that setting L_{stat} to approximately 25ms results in a high quality output, although varying this parameter for a specific input can yield an improvement in quality.

The above analysis has been performed on pitched signals, such as voiced regions of speech, leaving the question of how unvoiced or noise-like regions are time-scaled somewhat unanswered. However, the unvoiced/noisy regions of speech can also be viewed as being 'quasi-periodic' in the sense that the perceptually important characteristics of noise, i.e. the power within bark bands [32], are effectively constant, and therefore repetitive, over short time segments of the input. Furthermore, the choice of overlap position for noisy signal is not as critical as for periodic segments since discontinuities introduced through a 'poor' choice of overlap will not generally be perceived. For the case where noise energy does not extend over a sufficient duration, i.e. transients, the method described above may result in the undesired repeating or discarding of these short time energy segments; however this problem can be resolved if these segments of audio are detected [33].

3.2. Cross-Fading

The final important consideration within a time-domain implementation is the duration over which a cross-fade is applied between overlapping synthesis frames. The purpose of the cross-fade is to smooth out discontinuities between synthesis frames [21]. Typically, a cross-fade is applied along the entire duration of the 'optimal' synthesis overlap determined during the search stage; however a cross-fade of this duration may be unnecessary and can result in redundant computations. Consider the case of a perfectly periodic signal being time-scaled and synthesis frames are perfectly synchronised, if no cross-fade is applied and synthesis frames are simply appended to each other at an appropriate point within the 'optimal' overlap, then no artefacts will be perceived since discontinuities are not introduced and the period of the signal remains unaltered. However, audio signals are typically not perfectly periodic and some level of cross-fading is required in order to ensure that a smooth transition occurs between synthesis frames. The duration of the cross-fade required to ensure that a smooth transition occurs is dependent on the level of similarity of the synthesis frames and an 'intelligent' method of determining the cross-fade duration for each iteration of the SOLA algorithm could take the value of $R_m(k_m)$ returned by the correlation function into consideration. However a method such as this would require the introduction of a threshold which can introduce problems of its own if the threshold is not adequately set (or if an alternative means for determining the optimal overlap position is used – see section 4). It has been found that fixing the duration of the cross-fade to between 2ms-5ms generally provides an adequate solution, although longer cross-fades will reduce the effects of discontinuities by a greater degree. It should also be noted that the choice of cross-fade (linear and raised cosine have been suggested in the literature) does not have a significant impact on the quality of the results obtained [29].

3.3. Overlap Prediction

Both [5] and [34] make use of 'overlap prediction' in order to reduce the number of computational operations required to time-scale an input signal. The basis of 'overlap prediction' can be understood by considering the situation illustrated in figure 5; for the $(m\text{-}1)^{th}$ iteration an offset of k_{m-1} was determined in the usual manner i.e. through an evaluation of the correlation function described by equation (1); however, in the m^{th} iteration it can be seen that a common segment (the shaded regions) within the synthesis frames exists. Maximum correlation will occur for the synthesis overlap at which the common segments are aligned; therefore the synthesis frames will be overlapped at this position.

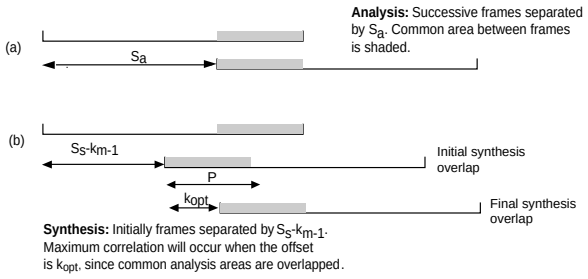

Figure 1 Overlap Prediction

It can be shown [34] that this type of situation occurs when

$$k_{min} \leq k_{m-1} + S_a - S_s \leq k_{max} \quad (17)$$

If this condition occurs then there is no need to calculate the correlation function and the offset, k_m, can simply be set equal to

$$k_m = S_a - S_s + k_{m-1} \quad (18)$$

To determine the benefits of 'predictive skipping' the probability of equation (17) occurring is determined. For this purpose, the values derived in section 2 are substituted into (17)

$$0 \leq k_{m-1} + (1-\alpha) \frac{L_{stat} - P}{|1-\alpha|} \leq P \quad (19)$$

For time scale compression the probability is given by:

Probability $(0 \leq k_{m-1} + L_{stat} - P)$ i.e. 1, since k_{m-1} is in the range 0 to P and $L_{stat} > P$,and

Probability $(P \geq k_{m-1} + L_{stat} - P)$ i.e. $(2P - L_{stat})/P$ if k_{m-1} is equally likely to be any value in the range 0 to P.

For time scale expansion the probability is given by:

Probability $(0 \leq k_{m-1} - L_{stat} + P)$ i.e. $(2P - L_{stat})/P$, if k_m is equally likely to be any value in the range 0 to P, and

Probability $(P \geq k_{m-1} - L_{stat} + P)$ i.e. 1, since k_m is in the range 0 to P and $L_{stat} > P$.

From above, the probability of equation (19) being satisfied is $2P - L_{stat}$ for all time-scale modifications. Therefore, the benefit of prediction is dependent on the maximum length of segment that can be discarded/repeated during a single iteration of the algorithm i.e. L_{stat}; if L_{stat} is greater than or equal to $2P$ then prediction provides no additional advantage.

3.4. Alternative Approaches

There are a number of alternative approaches to a SOLA based implementation, such as [35], [36] and [5], which are also capable of producing a high quality output, however they perform the same basic operation of discarding/repeating suitable segments of the input and offer no significant advantage over the SOLA based implementation described above.

One other popular time-scaling approach is the pitch synchronous overlap-add (PSOLA) [37] algorithm. This approach explicitly identifies the local pitch period of the input along the entire duration of the input, and then discards/repeats pitch periods appropriately, similar to SOLA based methods. For time-scaling purposes there is generally no benefit in using PSOLA over SOLA; although SOLA based implementations tend to be more robust and efficient since there is no requirement for explicit pitch period detection (although pitch detection is implicit within the similarity measure employed, since discarded/repeated segments are typically integer multiples of the local pitch period during voiced regions of speech [38]). The main advantage of PSOLA is in its use for pitch-scaling applications. One simplified and commonly used approach for pitch-scaling uses a process of time-scaling followed by resampling; this results in an altering of the formant structure of the pitch-scaled output which has the effect of modifying the timbre characteristics of the signal. By employing a PSOLA method for pitch-scaling, the formant structure of the pitch-scaled output is preserved, as explained in [39] and [40], resulting in a more natural sounding pitch-scaled output.

4. ALTERNATIVE SYNCHRONIZATION PROCEDURES

In section 3 a SOLA based implementation is outlined which makes use of a normalized correlation function to identify a suitable synthesis overlap. A number of computationally efficient alternatives to normalized correlation have been proposed in various publications. In this section these alternatives are summarized and a computational evaluation of these procedures, including normalized correlation, is provided. The evaluation is based upon the number of basic arithmetic, shift and compare operations each approach requires. A summary of this comparison is given in section 5.

4.1. Basic Unbiased Correlation

The normalizing denominator of equation (1) has the effect of reducing the magnitude of the correlation function when a high energy noise burst (transient), or any other high energy segment, exists within one of the synthesis frames. This normalizing process comes at a relatively high computational expense and the less complex unbiased correlation function has been used in [36], and is given by

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} y(mS_s + k + j)x(mS_a + j)}{L_k} \quad (20)$$

where L_k is the length of the overlapping region as given by equation (7).

4.1.1. Computational Requirements

Correlation can be efficiently determined through the use of an FFT-based convolution technique [41]. This approach requires the following steps to be taken:

1. Calculate the FFT of the overlapping synthesis segments. This involves two L_{max} point real input FFT's.
2. Multiply the resulting FFT's. This involves L_{max} complex multiplications i.e. 4. L_{max} multiplies and 2.L_{max} additions.
3. Calculate the inverse FFT of the result of step 2. This involves one L_{max} complex input inverse FFT.

An N-point radix-2 FFT of a complex input requires approximately $2N\text{Log}_2N$ real multiplications and $3N\text{Log}_2N$ real additions [41]. It can also be shown that two N-point FFT's of two real inputs can be efficiently determined using one N-point complex FFT and $2N - 4$ real additions [41]. An N-point inverse FFT of a complex input requires approximately the same number of operations as an N-point FFT. In addition, the unbiasing denominator term of equation (20) requires one division per overlap i.e. P divisions.

Having determined the correlation function, the maximum value of $R_m(k)$ must be found. This requires P comparisons.

4.2. Normalised Correlation

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} y(mS_s + k + j)x(mS_a + j)}{\sqrt{\sum_{j=0}^{L_k-1} x^2(mS_a + j)\sum_{j=0}^{L_k-1} y^2(mS_s + k + j)}} \quad (21)$$

4.2.1. Computational Requirements

The numerator of equation (21) can be determined in the same manner as described in subsection 4.1. Within the SOLA based implementation described in section 3, the correlation function is determined for a minimum synthesis overlap of $L_{max} - P$ to a maximum of L_{max}. If the denominator is first determined for the minimum synthesis overlap, it can then be computed efficiently through an iterative process for successive, increasing, synthesis overlaps. Each summation term of the denominator initially requires $L_{max} - P$ multiplies and $L_{max} - P$ additions for the minimum overlap. For subsequent synthesis overlaps one addition and one multiplication per summation term is required, followed by one multiplication of the summation terms and the application of a square root to the resulting product. Also, for each of the P possible synthesis overlap positions one division of the denominator into the numerator is required.

There are numerous methods for determining the square root. One common approach is the Newton-Raphson algorithm [42]. This approach requires one shift (to determine a division by 2), one addition and one division per iteration with an adequate result, for this application, generally being returned after 10 iterations.

P comparisons are then required to determine the maximum of the correlation function.

4.3. Simplified Normalised Correlation

The simplified normalized correlation function is suggested for use within time-scale modification in [15] and is given by

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} y(mS_s + k + j)x(mS_a + j)}{\sum_{j=0}^{L_k-1} |x(mS_a + j)| \sum_{j=0}^{L_k-1} |y(mS_s + k + j)|} \quad (22)$$

4.3.1. Computational Requirements

The simplified normalization function of equation (22) is calculated in a similar manner to equation (21), however, each summation term of the denominator initially requires only $L_{max} - P$ additions for the minimum overlap. For subsequent synthesis overlaps only one addition per summation term is required together with one multiplication of the summation terms.

P comparisons are required to determine the maximum.

4.4. AMDF

The average magnitude difference function AMDF is suggested for use in [35] and is given by

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} |y(mS_s + k + j) - x(mS_a + j)|}{L_k} \quad (23)$$

4.4.1. Computational Requirements

This function requires, on average, $L_{max} - P/2$ subtractions, $L_{max} - P/2$ additions and 1 division for each synthesis overlap/offset.

P comparisons are required to determine the minimum.

4.5. Mean Square Difference

Similar to AMDF, the mean square difference provides additional emphasis on large differences in magnitudes and is given by

$$R_m(k) = \frac{\sum_{j=0}^{L_k-1} (y(mS_s + k + j) - x(mS_a + j))^2}{L_k} \quad (23)$$

4.5.1. Computational Requirements

The mean square distance measure requires, on average, $L_{max} - P/2$ subtractions $L_{max} - P/2$ additions, $L_{max} - P/2$ multiplications and 1 division for each synthesis overlap/offset.

P comparisons are required to determine the minimum.

4.6. Mean Square Difference

The envelope matching time-scale modification algorithm [43], [34] transforms the overlapping synthesis segments into one-bit functions prior to correlation. In [34] the following variables are defined

$$x_2(j) = sign(x(mS_a + j)) = \begin{cases} 1 & if\ x(mS_a + j) \geq 0 \\ -1 & if\ x(mS_a + j) < 0 \end{cases} \quad (25)$$

$$y_2(j) = sign(y(mS_s + j)) = \begin{cases} 1 & if\ y(mS_s + j) \geq 0 \\ -1 & if\ y(mS_s + j) < 0 \end{cases} \quad (26)$$

$$x_{2,k}(j) = x_2(j + k) \quad (27)$$

$$y_{2,k}(j) = y_2(j + k) \quad (28)$$

where k is the offset described in section 2.

The similarity function given by equation (29) is then used to determine a suitable offset

$$R_m(k) = \frac{\sum_{j=0}^{L_m(k)-1} y_{2,k}(j) x_{2,k}(j)}{L_m(k)} = \frac{\beta_{z,k}}{L_k}\left(2\sum_{j=1}^{M_k+N_k-2r_k}(-1)^{j+1}C_k(j) + (-1)^{M_k+N_k}L_k\right) \quad (29)$$

where $\beta_{z,k} = y_{2,k}(0)x_{2,k}(0)$ and

$A_k = \{j : x_{2,k}(j-1)x_{2,k}(j) = -1, for\ 0 < j < L_k\}$.

Therefore, A_k is the set of locations of the zero crossing points in $x_{2,k}$. Also, M_k is the cardinality of A_k i.e. the number of zero crossings in $x_{2,k}$.

$B_k = \{j : y_{2,k}(j-1)y_{2,k}(j) = -1, for\ 0 < j < L_k\}$.

Therefore, B_k is the set of locations of the zero crossing points in $y_{2,k}$. Also, N_k is the cardinality of B_k i.e. the number of zero crossings in $y_{2,k}$.

r_k is the number of common zero crossings in $x_{2,k}$ and $y_{2,k}$ i.e. the cardinality of $A_k \cap B_k$.

$C_k = A_k \oplus B_k$, where $\oplus$ is the Exclusive OR operator

Equation (29) should be calculated for each offset, k, in the range k_{min} to k_{min}, which still represents a significant number of computations [34]. However, it is shown in [34] that equation (29) need only be evaluated for a smaller subset of all possible offsets, K_o, where

$$K_o = \{k : A_k \cap B_k \neq \phi\} \cup \{k : b_{k-1,1} = 1 \cup b_{k+1,N_k} = N-1\} \cup \{k : a_{k-1,M_k} = L_{k-1} - 1 \cap L_k < N\} \cup \{k_{\min}, k_{\max}\} \quad (30)$$

where A_k and B_k are defined above and are also represented by

$A_k = \{a_{k,1}, a_{k,2}, a_{k,3}....., a_{k,M_k}\}$

$B_k = \{b_{k,1}, b_{k,2}, b_{k,3}....., b_{k,N_k}\}$

It should be noted that in [34] a value p is also defined, however within the implementation described in section 2 this parameter does not need to be considered.

Given that $K_o = \{k_1, k_2, k_3, \ldots, k_{Q+1}\}$, it is also shown in [34] that $R_m(k_{i+1})$ can be found iteratively from

$$R_m(k_{i+1}) = \frac{L_{k_i}}{L_{k_i} - (k_{i+1} - k_i)} R_m(k_i) + (k_{i+1} - k_i) \frac{2\beta_{z,k_i}\xi_{k_i} + \beta_{z,k_i}(-1)^{M_{k_i}+N_{k_i}+1}}{L_{k_i} - (k_{i+1} - k_i)} \quad (31)$$

where $\xi_{k_i} = \sum_{j=1}^{N_{k_i}} (-1)^{g(k_i,j)}$ and $g(k_i, j)$ is the location of $b_{k_i,j}$ in the set $C_{k_i,j}$.

Equation (31) is significantly simpler than equation (29) and equation (29) needs only be evaluated for k_1 and all other elements of K_o can be found iteratively and efficiently from equation (31).

4.6.1. Computational Requirements

The steps involved in the implementation of the EM-TSM algorithm can be summarised as follows:

1. Determine the envelope function of both of the synthesis frames.
2. Determine the set K_0 that is defined in [34] and described as being:
    a) 'any lag k such that there is at least one common crossing point between $x_{2,k}$ and $y_{2,k}$'.
    b) 'any k such that $y_{2,k-1}$ has a zero-crossing point which disappears in $y_{2,k}$'.
    c) 'any k such that $x_{2,k-1}$ has a zero-crossing point that disappears in $x_{2,k}$'.
    d) k_{min} and k_{max}.
3. For the first k in the set K_0 i.e. k_1, determine $R(k)$ using equation (29)
4. For the remaining k in K_0 determine $R(k)$ using equation (31).
5. Find the value of k for which $R(k)$ is a maximum.

The computations required for each step is now expanded:

Step 1:

L_{max} comparisons for each frame. A_k and B_k can also found in parallel.

Step2:

a) Approximately $ZC_{avg}(L_{max} - P/2)$ comparisons, where ZC_{avg} is the average number of zero crossings per sample. C_k and $g(k,j)$ can also be determined in parallel.

b) 1 comparison for each overlap to determine if $b_{k-1,1} = 1$ i.e. P comparisons.

c) 1 comparison for each overlap to see if $a_{k-1,M_{k-1}} = L_{k-1}$

d) No operations are required to find k_{min} and k_{max}.

Step 3:

Equation (29) requires the following operations:

1 addition and 1 subtraction to determine M_k+N_k and $M_k+N_k - 2r_k$.

1 comparison to determine each $(-1)^{M_k+N_k}$.

$M_k+N_k - 2r_k$ comparisons to determine $(-1)^{j+1}$.

Each comparison is followed by an addition or subtraction.

1 addition of the terms within brackets.

1 shift to calculate the multiply by 2.

1 comparison to determine $\beta_{z,k} = x_{2,k}[0]y_{2,k}[0]$.

1 subtraction to determine L_k, since L_k is given by $L_{max} - k$.

1 division by L_k.

Step 4:

Equation (31) requires the following operations to be performed:

1 multiplication to determine $L_{ki}R(k_i)$.

1 comparison to determine β_{z,k_i}.

1 shift is required for the multiply by 2.

2 additions to determine $M_{k_i} + N_{k_i} + 1$.

1 compare is required to determine $(-1)^{M_{k_i}+N_{k_i}+1}$.

2 subtractions to determine $L_{k_i} - (k_{i+1} - k_i)$.

1 multiplication to determine $(k_{i+1} - k_i)\left(2\beta_{z,k_i}\xi_{k_i} + \beta_{z,k_i}(-1)^{M_{k_i}+N_{k_i}+1}\right)$.

1 addition to sum LHS and RHS.

1 division by $L_{k_i} - (k_{i+1} - k_i)$.

$g(k,j)$ can be calculated during step 2 (a) since the location of $b_{k,j}$ in the set C_k can be determined at that time.

One iteration in the calculation of ξ_{k_i} requires one comparison to determine whether an addition or subtraction is required, followed by an addition or subtraction. On average ZC_{avg}

(L_{max} –$P/2$) iterations are required to determine each ξ_{k_i} .

Step 5:

Requires $Q+1$ comparisons, where $Q+1$ is the cardinality of K_o.

On average M_k and N_k are approximately $ZC_{avg}(L_{max} - P/2)$ and r_k is approximately $ZC_{common,avg}(L_{max} - P/2)$, where $ZC_{common,avg}$ is the average number of common zero crossings per sample.

4.7. MEM-TSM

In [43] a number of refinements to the EM-TSM synchronization procedure are proposed. Having obtained the envelope matching function, the offsets that correspond to the M largest magnitudes of the EMF, given by $\{k_{c,1}, k_{c,2}, k_{c,3},...., k_{c,M}\}$, are re-evaluated using the following decimated normalized correlation function

$$R_{m,2}(k) = \frac{\sum_{j=0}^{\frac{L_k}{q}-1} y(mS_s + k + q.j)x(mS_a + q.j)}{\sqrt{\sum_{j=0}^{\frac{L_k}{q}-1} x^2(mS_a + q.j) \sum_{j=0}^{\frac{L_k}{q}-1} y^2(mS_s + k + q.j)}} \quad (32)$$

The offset k_m is chosen such that $R_{m,2}(k)$ is a maximum for $k = k_m$ where k is an element of $\{k_{c,1}, k_{c,2}, k_{c,3},...., k_{c,M}\}$. By using the multiple candidate re-examination procedure described above, the quality of the output is improved upon over the EM-TSM implementation [43].

The efficiency of the EM-TSM approach is governed by the number of zero-crossings in the overlapping regions of the synthesis frames. In [43] it is noted that high frequency components and noise introduce many zero-crossings, thus increasing the computational requirements of the approach, yet it is the low frequency components that are most important in obtaining a 'good' synthesis overlap. One method used in [43] of reducing the number of computations within an EM-TSM implementation is to apply the following rule prior to determining $R_m(k)$; if the distance between two adjacent zero-crossing points in A_k or B_k is less than a pre-defined threshold, T_1, the pair is removed from A_k or B_k, respectively.

4.7.1. Computational Requirements

The function of equation (32) is a 'decimated' version of the normalized correlation function. Since the decimated correlation function is only determined for a relatively small number of synthesis overlaps, it is no longer computationally efficient to employ an FFT in determining the numerator. Calculating the numerator, on average, requires $(L_{max} - P/2)/q$ multiplies and $(L_{max} - P/2)/q$ additions for each candidate offset that is being re-examined.

The denominator of the decimated function is found in a similar manner to that described in subsection 4.3, however the number of operations required to determine each summation term is proportional to the maximum overlap associated with each of the candidate offsets i.e. the overlap associated with the minimum candidate offset. An estimate of the number of operations required to determine the summation terms for all candidate offsets is then $L_{c,max}/q$ multiplies and $L_{c,max}/q$ additions, where $L_{c,max}$ is the overlap associated with the smallest candidate offset. Assuming that the occurrence of a candidate offset is equally likely to occur anywhere in the range 0 to P, and assuming $M << L_{max}$, where M is the number of candidates being re-examined, it can be statistically shown that $L_{c,max}$ is approximately given by $P(M/(M+1))+ L_{max} - P$. Having determined both summation terms for each candidate offset, the summation terms are multiplied and the square root of their product is determined. The square root can be determined as in subsection 4.3. Finally the denominator is divided into the numerator. M comparisons are then required to determine the maximum.

In order to reduce the number of zero-crossings the distance between consecutive pairs is first determined and then compared with the defined threshold. This operation requires one subtraction and one comparison for each pair being evaluated in each frame.

4.8. GLS-TSM

The global and local search approach (GLS-TSM) [44] consists of two stages. The first stage is a preliminary global search, and the second is a refined local search. The global search consists of a search for an offset, $k_{globalmin}$, where $k_{globalmin}$ is chosen such that the difference in the number of zero-crossing points between synthesis frames, in their common overlapping region, is minimized. $k_{globalmin}$ lies between k_{min} and k_{max}.

Having found $k_{globalmin}$ the next step is to locate the zero crossing with the maximum slope in the 'output' synthesis frame within the overlapping region identified

during the global search i.e. $y(mS_s + k_{globalmin} + j)$ for $0 \leq j \leq k_{max} - k_{globalmin}$. The zero crossing with the maximum slope, labeled $Z_{max\ slope}$, occurs at $y(mS_s + k_{max\ slope})$ and is chosen since 'It is observed that a wrong match at a zero cross point with a greater slope has a more pronounced effect than a zero cross point with a smaller slope', [44].

In [44], an eleven dimensional feature vector, f, is defined which is used to represent local information in the region of a zero crossing. If a zero crossing occurs between $x(i)$ and $x(i+1)$, the eleven feature vector components are given by:

$$f_1 = x(i) - x(i+1) \qquad f_7 = |x(i+3)|$$

$$f_2 = |x(i)| \qquad f_8 = (x(i-1) - x(i+1))/2$$

$$f_3 = |x(i+1)| \qquad f_9 = |x(i-1)|$$

$$f_4 = (x(i) - x(i+2))/2 \qquad f_{10} = (x(i-2) - (i+1))/3$$

$$f_5 = |x(i+2)| \qquad f_{11} = |x(i-1)|$$

$$f_6 = (x(i) - x(i+3))/3$$

The feature vector of Z_{max_slope} is compared to the feature vectors of zero crossings in the corresponding locality of Z_{max_slope} on the 'input' synthesis frame i.e. $x(mS_a + j)$ for $k_{max\_slope} - k_1 \leq j \leq k_{max\_slope} + k_2$. It is found that choosing k_1 and k_2 such that approximately 10 zero crossings candidates exist results in a good quality output. The feature vectors are compared using the following distance measure

$$d_i = \frac{1}{11}\sum_{j=1}^{11}\left|f_x(j) - f_{y,i}(j)\right| \tag{33}$$

where, $f_x(j)$ is the j^{th} feature vector component of Z_{max_slope}, $f_{y,i}(j)$ is the j^{th} feature vector component of the i^{th} candidate zero crossing.

The candidate zero crossing that produces the smallest distance measure is then chosen so as to align with Z_{max_slope} i.e. given that the zero crossing that produces the smallest distance measure occurs at $x(mS_a + k_{best\_candidate})$ then

$$k_m = k_{max\_slope} - k_{best\_candidate} \tag{34}$$

where k_m is the 'optimum' offset described in section 2.

4.8.1. Computational Requirements

The steps involved within the GLS-TSM algorithm can be summarized as follows:

Step 1: *For each overlap position determine the number of zero-crossings in each of the synthesis frames.*
The most efficient way to determine this is to first find the zero-crossings in the minimum overlap region and iteratively determine the zero-crossings for the remaining overlap positions. This initially requires $L_{max} - P$ compares and one addition for each zero crossing in the minimum overlap region. Then for each of the remaining P possible overlaps a comparison is required with an addition required if a zero-crossing is detected.

Step 2: *Determine the overlap position that provides the minimum difference between the number of zero crossings in the analysis frame and the number of zero crossings in the synthesis frames. This provides the global search overlap.*
Using the data determined in Step 1, this procedure then requires P comparisons.

Step 3: *Find the slope at each zero crossing of the analysis frame within the global search overlap.*
For each zero crossing a subtraction is required followed by a division, to determine the slope. Assuming that the average global search overlap is $P/2$, then $(P/2).ZC_{avg}$ subtractions and $(P/2).\ ZC_{avg}$ divisions are required.

Step 4: *Find the zero crossing that corresponds to the maximum of all the slopes calculated. This zero crossing then becomes the reference zero crossing point.*
Finding the maximum slope requires $(P/2).ZC_{avg}$ comparisons.

Step 5: *Compute the feature vector for the reference zero crossing.*
Calculating an 11 point feature vector requires: 5 subtractions, 2 shifts (for two divide by 2 operations) and 2 divisions.

Step 6: *Compute the synthesis zero-crossings in the neighbourhood of the reference zero crossings. Use the U nearest neighbours.*
This requires U x step 5 operations.

Step 7: *Find the 'distance measure' between the reference feature vector and the candidate synthesis feature vectors.*
Calculating one distance measure requires:
11 subtractions, 11 additions, 1 division.

Step 8: *Find the minimum 'distance measure' and use the corresponding synthesis zero crossing point and reference to determine the final overlap position.*
This requires U comparisons.

4.9. Peak Alignment

A peak alignment approach has been described in [31] and [15]. Here a method is briefly outlined that allows a peak alignment approach be applied to the overlap-add procedure described in section 2.

The first step is to determine the maximum, i.e a peak, in $x(mS_a + j)$ for $1 \leq j \leq P$. Given that a maximum occurs at $j = j_{max,x}$, the next step is to determine the maximum in $y(mS_s - k_{m-1} + j_{max,x} + j)$ for $1 \leq j \leq P$. Given that a maximum occurs at $j = j_{max,y}$

$$k_m = j_{max,y} - j_{max,x} \tag{35}$$

It should be noted that L_{max} must be $2P$ for the peak alignment process to operate as expected.

4.9.1. Computational Requirements

This approach requires P comparisons to determine the peak/maximum in each frame. Calculating $mS_s - k_{m-1} + j_{max,x}$ and k_m requires one addition and two subtractions.

5. COMPUATIONAL COMPARISON SUMMARY

	Compares	Additions /Subtracts	Shifts	Mults /divides	Total
Peak Alignment	$2.P$	3	0	0	**1.00**
GLS-TSM	$2.L_{max} + ZC_{avg}.(P/2) + U+P$	$2.L_{max}.ZC_{avg} + (P/2).\ ZC_{avg} + U.27 + 5$	$(1+U).2$	$(1+U).3 + (P/2).\ ZC_{avg}$	**4.04**
EM-TSM	$3.Q + 2.P + 3 + 2.L_{max} + 2.(ZC_{avg} - ZC_{common,avg})(L_{max} - P/2) + ZC_{avg}.(L_{max} - P/2)(Q+1)$	$4 + 5.Q + Q.ZC_{avg}.(L_{max} - P/2)+ 2.(ZC_{avg} - ZC_{common,avg})(L_{max} - P/2)$	$Q+1$	$3.Q+1$	**43.58**
MEM-TSM Reduced zero crossings	$ZC_{avg}.L_{max}$	$ZC_{avg}.L_{max}$	0	0	**18.12**
MEM-TSM Candidate re-examination	M	$(P(M/(M+1)) + L_{max} - P)/q + M(L_{max} - P/2)/q + 10.M$	$10.M$	$11.M + (P(M/(M+1))+ L_{max} - P)/q + M.(L_{max} - P/2)/q$	**21.67**
Unbiased Correlation	P	$2.L_{max}(3.\text{Log}_2(2.L_{max}) - 1) - 4$	0	$4.L_{max}.\text{Log}_2(2.L_{max}) + P$	**91.63**
Simplified Normalized Correlation	P	$2.L_{max}(3.\text{Log}_2(2.L_{max}) - 1) - 4 + 2.L_{max}$	0	$4.L_{max}.\text{Log}_2(2.L_{max}) + 2.P$	**94.11**
Normalised Correlation	P	$2.L_{max}(3.\text{Log}_2(2.L_{max}) - 1) - 4 + 2.L_{max} + 10.P$	$10.P$	$4.L_{max}.\text{Log}_2(2.L_{max}) + 12.P + 2.L_{max}$	**111.01**
AMDF	P	$2.P.\ (L_{max} - P/2)$	0	P	**239.50**
Mean Square Difference	P	$2.P.\ (L_{max} - P/2)$	0	$P.(L_{max} - P/2) + P$	**358.75**

Table 1 A Comparison of Synchronization Procedures

The column furthermost to the right of the table above shows a normalized comparison of the number of operations each approach requires. The comparison assumes that each operation requires the same duration to process and uses the parameter values given below. The totals are normalized by dividing the total by the number of operations required by a peak alignment approach. The totals for the two MEM-TSM rows (shown shaded) also take the number of operations required by EM-TSM, after the zero crossing reduction has been applied, into consideration.

For a sampling rate of 16kHz the following values typically apply:

Maximum period, P = 160 samples; corresponding to 10ms.

The initial (and maximum) synthesis overlap, L_{max} = 320; corresponding to 20ms.

Average zero crossings per sample, $ZC_{avg} = 0.19$.

Average common zero crossings, between synthesis frames, per sample, $ZC_{common,avg} = 0.068$.

The number of re-examined candidates in GLS-TSM, U = 10.

The number of re-examined candidates in EM-TSM, M = 8.

Average number of elements in K_0, Q = 128.

The MEM-TSM correlation decimation factor, $q = 5$.

The figures shown for the two MEM-TSM rows were obtained when the T_1 parameter is set to 6 samples, for the application of the zero crossing reduction procedure. After the zero crossing reduction procedure is applied ZC_{avg} becomes 0.072, $ZC_{common,avg}$ becomes 0.0066 and Q becomes 99.2. The parameters Q, ZC_{avg} and $ZC_{common,avg}$ were determined from the examination of 250 test signals obtained from the TIMIT speech corpus [45].

It should be noted that a further reduction in the computational complexity of synchronization procedures which employ the FFT could also be achieved through the use of techniques such as FFT pruning [46] or the Goertzel technique [47]. In addition, as noted in [34], synchronizing high frequency content of a signal is not as important as the low frequency data, therefore all of the synchronization procedures are likely to produce high quality results when applied to down sampled data. Such an approach is also suggested in [36] whereby the input (sampled at 48 kHz) was first down sampled by a factor of 6, thus providing significant computational reduction.

6. OBJECTIVE OUTPUT EVALUATION

The output quality of a time-domain time-scale modification algorithm is primarily dependent on how similar the overlapping segments of the synthesis frames are; hence the reason for making use of similarity measures in finding the optimum overlap. The same similarity measures can be (and have been in [34]) used to provide an objective evaluation of the quality of each synchronization procedure; however some difficulty lies in determining which similarity measure is perceptually 'best'. For the purpose of the evaluation presented here, it is assumed that the similarity measures are equally valid and are therefore used to assess the quality of each approach; however, in an attempt to determine which synchronization procedure is 'best', the results obtained from each output quality assessment measure are statistically normalized and the 'best' synchronization procedure is deemed to be that procedure that is associated with the maximum of the sum of the normalized measures.

In addition to the time-domain similarity measures presented earlier (unbiased correlation, normalized correlation, AMDF and mean-square difference) a frequency-domain based similarity measure is also used, which is essentially the mean square difference of the magnitude spectra of the overlapping segments. A similar measure is used in [48], which was derived from [49]. It should be noted that in [49] a time-scale modification approach is presented that iteratively attempts to minimize a similar measure though manipulation of the input signal's short-time Fourier transform. It is also worth noting that the motivation behind the development of SOLA [20] was to reduce the number of iterations required to implement the iterative procedure of [49].

Over 250 test signals obtained from the TIMIT speech corpus were used during the objective evaluation. Each test signal is time-scaled by a factor of 2 using each of the synchronization procedures described in section 3. It should be noted that any time-scale factor could be used; however a time-scale factor close to one requires a relatively small number of iterations, on the other hand a very large time-scale factor would return very similar results from successive iterations of the algorithm. For each iteration of each algorithm, the similarity measures given by equations (36-39) are applied.

$$measure_1 = \frac{\sum_{j=0}^{L_{k_m}-1} y(mS_s + k_m + j)x(mS_a + j)}{\sqrt{\sum_{j=0}^{L_{k_m}-1} x^2(mS_a + j) \sum_{j=0}^{L_{k_m}-1} y^2(mS_s + k_m + j)}} \quad (36)$$

$$measure_2 = \frac{\sum_{j=0}^{L_{k_m}-1} y(mS_s + k_m + j)x(mS_a + j)}{L_{k_m}} \quad (37)$$

$$measure_3 = \frac{\sum_{j=0}^{L_{k_m}-1} \left| y(mS_s + k_m + j) - x(mS_a + j) \right|}{L_{k\,m}} \quad (38)$$

$$measure_4 = \frac{\sum_{j=0}^{L_{k_m}-1} \left(y(mS_s + k_m + j) - x(mS_a + j) \right)^2}{L_{k_m}} \quad (39)$$

Having accumulated the measures for each synchronization procedure (and a random offset), the measures are then normalized using a standard score approach [50] e.g. given that the accumulation of $measure_1$ is given by the set $measure_{1,acc} = \{m_1, m_2, \ldots, m_9\}$, where m_w is the accumulation of $measure_1$ when applied to synchronization procedure number w, then the normalized set is given by

$$\left\{\frac{m_1 - mean(measure_{1,acc})}{stdDev(measure_{1,acc})}, \frac{m_2 - mean(measure_{1,acc})}{stdDev(measure_{1,acc})}, \ldots, \frac{m_9 - mean(measure_{1,acc})}{stdDev(measure_{1,acc})}\right\} \quad (41)$$

where *stdDev* is the standard deviation.

In addition, the sign of the normalized data set is inverted for $measure_3$ and $measure_4$ to take account of the fact that a minimization of these functions is desired.

Table 2 shows the results of the objective output quality assessment, with the furthermost right column showing the sum of the normalized measures for each of the synchronization procedures.

	$Measure_1$	$Measure_2$	$Measure_3$	$Measure_4$	Total
Mean Square Difference	0.83	0.57	0.55	0.59	2.55
AMDF	0.55	0.52	0.73	0.70	2.50
Normalized Correlation	0.44	0.47	0.80	0.63	2.35
Unbiased Correlation	0.71	0.42	0.35	0.45	1.93
Simplified Normalized Correlation	0.36	0.79	0.25	0.35	1.76
MEM-TSM (M=20, q = 5)	0.45	0.36	0.39	0.37	1.58
EM-TSM	0.26	0.22	0.29	0.20	0.98
Peak alignment	-0.33	-0.32	-0.27	-0.20	-1.12
GLS-TSM	-0.83	-0.42	-0.50	-0.46	-2.22
Random Offset	-2.46	-2.62	-2.60	-2.64	-10.33

Table 2 An Objective Output Quality Comparison

In two separate tests the objective measures were applied to the test signals with additional noise injected into the test signals and only those frames considered voiced (since appropriate overlapping of voiced regions of speech is, in general, perceptually more important than that of unvoiced or silent regions). The results of both additional tests are closely approximated by those presented in table 2.

7. DISCUSSION

The objective output quality assessment given in section 6 is useful in that it provides a quantifiable comparison of the various approaches, but it is important to view these results with some knowledge of a subjective evaluation. It should be noted that in general each of the approaches outlined in section 3 produce a reasonably high quality output and when 'inexperienced' subjects are presented with a comparison between any of the approaches they find it difficult to differentiate between them. It is only when 'experienced' subjects, i.e. subjects who are working in the audio processing realm, evaluate the various approaches that the differences become more apparent. For a small number of tests and 'experienced' subjects (in a quite office environment), it was found that the results of the objective comparison relate to that of a subjective comparison quite closely; however, extensive listening tests would be required in order to validate the objective assessment presented here with subjective tests with any statistical accuracy.

The synchronization procedures presented in section 4 can be classified into two groups i.e. similarity measures (correlation, AMDF, MSD, EM-TSM, MEM-TSM) and feature matching processes (GLS-TSM and peak alignment). Similarity measure processes could be viewed as multiple feature matching processes. From the results of the objective output quality assessment, there is a clear divide in the quality achieved between these groups, which can be attributed to the fact that only one feature is used in the synchronization process for GLS-TSM and peak alignment implementations, resulting in a higher probability of choosing incorrect or ambiguous features to align (for example, consider the ambiguous situation that would occur in figure 4 (b) or (c) if either feature matching process was used). There is also a significant difference in the results obtained for GLS-TSM and peak alignment, with results suggesting that the peak alignment process is superior to the zero-crossing feature used in GLS-TSM. Intuitively this makes sense when it is considered that there are, in general, many more candidate zero-crossings in a segment of a speech signal than 'maximum' peaks, therefore the probability of an 'incorrect' match is increased when a zero-crossing feature is used.

The time-domain approaches described in this paper time-scale all regions of the input signal by same amount; this can result in artefacts being introduced into the time-scaled output, e.g. transient skipping or repetition [33] and a 'slurred'/'drunken' sounding output [51], which tends to be most problematic for large time-scale factors. The problem of transient preservation has been addressed in [33], while [52] and [3] have taken steps in producing a more natural sounding time-scaled output by applying different levels of time-scaling to different regions of the input signal e.g. in [52] it is suggested that consonants be time-

scaled compressed more than vowels. In [53] and [54] the entire input is first segmented into 'auditory scenes' and each scene is then time-scaled individually by an appropriate amount, resulting in a further improvement in the quality of output [53].

Finally, it should be noted that the underlying requirement for high quality time-scale modification when time-domain approaches are applied, is the existence of a quasi-periodic element within the signal being time-scaled. This requirement is generally fulfilled in simple monophonic audio signals such as speech and monophonic music, but for more complex audio, such as polyphonic music, periodicity is largely lost and time-domain approaches prove inadequate. However, for small time-scaling +-15% high quality results can be obtained [53], [36]. This is due to the fact that some level of periodicity will generally exist in complex audio and also because small amounts of distortion introduced by poor synchronization of frames will generally not be perceived. For time-scaling complex audio by an amount greater than +-15%, phase vocoder [48], sinusoidal modeling [55] or time-domain/subband [56], [57] approaches should be employed.

8. CONCLUSION

This paper develops a set of guidelines for the choice of parameters used within time-domain time-scale modification algorithms within the context of a waveform editing procedure. Both computational load and output quality comparisons of a number of commonly used synchronization procedures are presented. A brief outline of each synchronization procedure is first given in section 4, followed by a thorough computational load analysis that considers the number of basic arithmetic, shift and compare operations each procedure requires. An objective assessment of the quality produced, when each synchronization procedure is employed, is given in section 6 and a brief discussion of the comparative results is presented in section 7. Results of the objective assessment indicate that the use of a mean square distance function produces the highest quality output; however more efficient implementations are capable of producing a similar quality of output with a significant reduction in computational load. The peak alignment synchronization procedure is the most efficient, requiring approximately 1% of the operations required by the commonly used normalized correlation function; however its efficiency comes at the expense of some degradation in the quality of the output.

9. REFERENCES

[1] Amir, A.; Cohen, G.; Ponceleon, D.; Blanchard, B.; Petkovic, D.; Srinivasan, S., "Using audio time scale modification for video browsing," Proc. 33rd Annual Hawaii Int. Conf. on System Sciences, pp.1117 – 1126, 2000.

[2] Erogul, O.; Karagoz, I., "Time-scale modification of speech signals for language-learning impaired children", Proc. 2nd Int. Conf. Biomedical Engineering Days, pp.33 – 35,1998.

[3] Donnellan, O.; Elmar Jung; Coyle, E , "Speech-adaptive time-scale modification for computer assisted language-learning", IEEE Int. Conf. on Advanced Learning Technologies, pp.165–169, 2003.

[4] Demol M., Struyve K., Verhelst W., Paulussen H., Desmet P.; Verhoeve P., "Efficient non-uniform time-scaling of speech with WSOLA for CALL applications", In Proc. of InSTIL/ICALL Symposium, paper 007, 2004.

[5] Hejna D. J., "Real-time time-scale modification of speech via the synchronized overlap-add algorithm", M.I.T. Masters Thesis, Department of Electrical Eng. and Computer Science, 1990.

[6] Pallone G., Boussard P., Daudet L., Guillemain P., Kronland-Martinet R., "A wavelet based method for audio-video synchronization in broadcasting applications". In proc of the DAFX, 1999.

[7] Mansour, M.F.; Tewfik, A.H., "Audio watermarking by time-scale modification", IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3, pp. 1353-1356, 2001.

[8] Arons, B. "Techniques, Perception, and Applications of Time-Compressed Speech", In Proceedings of 1992 Conference, American Voice I/O Society, pp. 169-177, 1992.

[9] Bonada, J., "Automatic Technique in Frequency Domain for Near-Lossless Time-Scale Modification of Audio", 'Proceedings of International Computer Music Conference, 2000.

[10] Tan, R.K.C.; Lin, A.H.J, "A Time-Scale Modification Algorithm Based on the Subband Time-Domain Technique for Broad-Band Signal Applications", Journal of the Audio Engineering Society, vol. 48, no. 5, pp. 437-449, 2000.

[11] Malah D., Crochiere R.E., Cox R.V., "Performance of Transform and Sub- band Coding Systems Combined with Harmonic Scaling of Speech", IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-29, No. 2, , pp. 273-283, April 1981.

[12] Wayman J.L., Wilson D.L., "Some improvements on the synchronized-overlap-add method of time scale modification for use in real-time speech compression and noise filtering", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 36 , Issue: 1, pp. 139 - 140 ,1988.

[13] Teletar, Z., Erogul, O. , "Heart sounds modification for the diagnosis of cardiac disorders", IJCI Proceedings of International Conference on Signal Processing, Vol.1, No.2, pages 101-105, 2003

[14] Rodriguez-Hernandez, M., Casajus-Quiros. F., "Improving time-scale modification of audio signals using wavelets", *IC-SPA*T, Vol. 2, pages 1573–1577, 1994.

[15] Lawlor, B. and Fagan, A.D., "A Novel High Quality Efficient Algorithm for Time-Scale Modification of Speech", 6th Conference on Speech Communication and Technology, 1999.

[16] Moulines, E., Charpentier, F., "Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones", Speech Communication, Vol. 9 (5/6), pp. 453-467, 1990.

[17] Macon, M.W.; Clements, M.A., "Speech concatenation and synthesis using an overlap-add sinusoidal model", 1996 IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume: 1, Pages:361 – 364, 1996.

[18] Verhelst W., Brouckxon H., "Rejection phenomena in inter signal voice transplantations", IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, pages 165-168, 2003.

[19] Verhelst W., Van Compernolle D., Wambacq P., "A Unified View on Synchronized Overlap-Add Methods for Prosodic Modification of Speech." In Proc. International Conference on Spoken Language Processing, vol. II, pages 63-66, 2000.

[20] Roucos S. and Wilgus A.M., "High Quality Time-Scale Modification for Speech", IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 493-496, March 1985.

[21] Lee, F., "Time compression and expansion of speech by the sampling method" Journal of the audio engineering society, pp. 738 –742, May 1972.

[22] Fairbanks, G; Everitt, W. L. and Jaeger, R. P., "Method for time or frequency compression-expansion of speech", Trans of the Inst. of Radio Eng professional group on audio,pp.7–12, 1954.

[23] Gabor, D., "Theory of Communication", Journal IEE, Vol. 93, pp. 429-457, 1946.

[24] Dudley, H., "Remaking speech", Journal of the Acoustic Society of America, Vol. 11, No. 2, pp. 169-175, 1939.

[25] Scott, R.J., "Time adjustment in speech synthesis", Journal of the Acoustical Society of America, Vol. 41, No.1, pp. 60-65, 1967

[26] Neuberg E. P., "Simple pitch-dependent algorithm for high quality speech rate changing", Jour. of the Acoustical Society of America, Vol. 63, No.2, pp.624-625, 1978.

[27] Malah, D., "Time-domain algorithms for harmonic bandwidth reduction and time scaling of speech signals", IEEE trans on acoustics, speech and signal processing, vol. ASSP-27,No.2,pp. 121–133,1979.

[28] Dorran D., Lawlor, R. and Coyle E., "Time-Scale Modification of Speech using a Synchronised and Adaptive Overlap-Add (SAOLA) Algorithm", AES 114th Convention, preprint no. 5834, 2003.

[29] Makhoul, J.; El-Jaroudi, A. "Time-scale modification in medium to low rate speech coding", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing., Vol. 11 , pp:1705 – 1708, 1986.

[30] Hardam, E., "High quality time scale modification of speech signals using fast synchronised-overlap-add algorithms", Proc. of the IEEE International conference on acoustics, speech and signal processing, pp. 409 -412, 1990.

[31] Dorran D., Lawlor, R. and Coyle E., "High Quality Time-Scale Modification of Speech using a Peak Alignment Overlap-Add Algorithm (PAOLA)", IEEE International Conf. on Acoustics, Speech and Signal Processing, Vol. 1, pp. I-700 – I-703, 2003.

[32] Goodwin, M.; Vetterli, M., "Time-frequency signal models for music analysis, transformation, and synthesis", IEEE Int. Sym. on Time-Frequency and Time-Scale Analysis, Pages:133 – 136, 1996.

[33] Lee, S., "Variable Time-Scale Modification of Speech using Transient Information", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, pp. 1319 -1322 vol.2., 1997

[34] Wong P.H.W., Au O.C., "Fast SOLA-based Time Scale Modification using Envelope Matching," KAP Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology, vol. 35, no. 1, pp. 75-90, 2003.

[35] Verhelst, W.; Roelands, M., **"**An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech**",** IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, pp. 554 -557 vol.2, 1993

[36] Laroche, J., "Autocorrelation method for high-quality time/pitch-scaling", IEEE Workshop on App's of Signal Processing to Audio and Acoustics, pp.131-134, 1993.

[37] Charpentier F. and Moulines E., "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones", Speech rcommunication, vol. 9, No.s 5/6, pp 13-19, 1990.

[38] Moulines, E. and Laroche, J., "Non-parametric techniques for pitch-scale and time-scale modification of speech", speech communication 16, pp. 175-205, 1995.

[39] Moulines, E. and Verhelst W., "Time-Domain and Frequency-Domain Techniques for Prosodic Modification of Speech", Speech coding and synthesis, 1995.

[40] Bristow-Johnson, R., "A Detailed Analysis of a Time-Domain Formant-Corrected Pitch-Shifting Algorithm", Journal of the Audio Engineering Society, Volume 43, Number 5, pp. 340-352, 1995.

[41] Mitra S.K., Kaiser J.F., "Handbook for digital signal processing", Wiley interscience, 1993.

[42] Patterson D.A., Hennessey, J.L., Computer Architecture: A Quantitative Approach. Morgan Kaufmann Publishers, Inc., 2nd ed., 1996.

[43] Wong, P.H.W.; Au, O.C., "Fast SOLA-based time scale modification using modified envelope matching", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, vol. 3, pp:III-3188 - III-3191, 2002

[44] Yim, S.; Pawate, B.I, "Computationally efficient algorithm for time scale modification (GLS-TSM)", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing , Vol: 2. pp. 1009 -1012, 1996.

[45] http://wave.ldc.upenn.edu/Catalog/docs/TIMIT.html

[46] Markel, J., "FFT pruning", IEEE Trans on Audio and Electroacoustics, vol: 19/4, pp:305 – 311, 1971

[47] Oppenheim A.V., Shafer R.W., "Digital Signal Processing", Eaglewood Cliffs,Prentice-Hall, 1975.

[48] Laroche, J.; Dolson, M., "Improved Phase Vocoder", Speech and Audio Processing, IEEE Transactions on speech and audio processing, Volume: 7 Issue: 3 , Page(s): 323 –332, May 1999.

[49] Griffen, D, W. and Lim, J. S., "Signal Estimation from modified short-time Fourier Transform", IEEE trans on acoustics, speech and signal processing, Vol. ASSP-32, No.2, pp.236-243, 1984.

[50] Coolidge F.L., "Statistics: A Gentle Introduction", Sage Publications, 2000.

[51] Di Martino, J.; Laprie, Y., "Suppression of phasiness for time-scale modifications of speech signals based on a shape invariance property", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, vol: 2 , pp:853 – 856, 2001.

[52] Covell, M.; Withgott, M.; Slaney, M., "MACH1: nonuniform time-scale modification of speech", IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Vo: 1, pp. 349 - 352 , 1998.

[53] Crokett B.G., "High Quality Multi-channel Time-Scaling and Pitch-Shifting using Auditory Scene Analysis", Audio Engineering Society Convention, preprint no. 5948, 2003.

[54] Duxbury C., Davies M., Sandler M., "Temporal Segmentation and Pre-analysis for Non-linear Time-scaling of Audio", 114th Convention of the AES, Preprint no. 5812, 2003.

[55] Smith, J. O. and Serra, X., "PARSHL: An Analysis /Synthesis Program for Non-Harmonic Sounds based on a Sinusoidal Representation", Proceedings of the International Computer Music Conference, pp. 290 – 297, 1987.

[56] Dorran D., Lawlor, R., "Time-scale modification of music using a subband approach based on the bark scale", IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics, pages 173-176, 2003.

[57] Dorran D., Lawlor R., "Time-scale modification of music using a synchronized subband/time-domain approach," IEEE International Conference on Acoustics, Speech and Signal Processing, pp. IV 225 – IV 228, 2004.