Time-scale modification for speech coding

Burazerovic, D.

Award date:
2000

TECHNISCHE UNIVERSITEIT EINDHOVEN

FACULTEIT ELEKTROTECHNIEK

LEERSTOEL Signal Processing Systems

Time-scale modification for speech coding

Dževdet Burazerović

SPS 12-00

Verslag van een afstudeerproject verricht
binnen de leerstoel Signal Processing Systems
onder leiding van ir. J.H.F. Ritzerfeld,
ir. M.R. Taori en drs. A.J. Gerrits (Philips Research)
in de periode november 1999 – augustus 2000

Eindhoven, september 2000

Time-scale modification for speech coding

Dževdet Burazerović

25-08-2000

Contents

Abstract

Time-scale modification (TSM) of speech refers to compressing or expanding the time-scale of speech, while preserving the identity of the speaker (pitch, formant structure). It can be used for various applications, such as text-to-speech synthesis, film/sound-track post-synchronisation, foreign language learning, etc.
In this study, we have investigated if TSM can be beneficial to speech coding. The central idea is to compress the time-scale of a speech signal prior to coding, enabling usage of less bits, and to expand it by a reciprocal factor after decoding, to come to the original time-scale.
For this purpose, we have built a speech coding system, incorporating a compressor, an expander, and standard speech coders. As opposed to traditional approaches, we propose the use of parametric modeling of unvoiced sounds.
For evaluation purposes, we have used 25 compression - 33 expansion. Comparison with some standard coders is presented.
We conclude that speech coders with TSM perform worse than dedicated speech coders at comparable bit-rates. However, TSM can be beneficial in providing graceful degradation at higher bit-rates.

Chapter 1

Introduction

1.1 Purpose

Time-scale modification (TSM) of speech refers to compression or expansion of the time scale of speech, while preserving the identity of the speaker (pitch, formant structure). An illustrative example is given is Figure 1.1, showing a speech excerpt and its time-scale modified versions. (The results have been obtained using the SOLA algorithm, which will be discussed in Section 2.2.) As can be imagined, TSM of speech (and audio) can be used for many applications, such as as text-to-speech synthesis, film/sound-track post-synchronization, quick replay of recorded messages, etc. Therefore, it is a subject of major theoretical and practical interest.

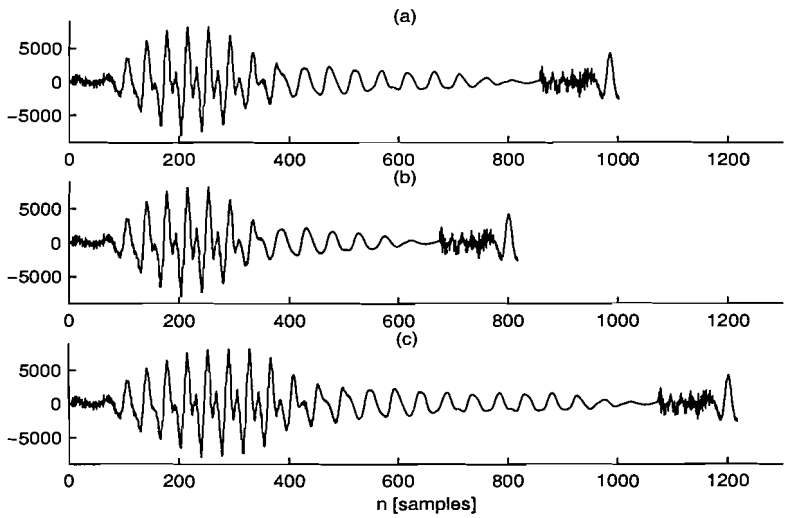

Figure 1.1: The figure show an original speech excerpt (a), its time-scale compressed (b) and time scale expanded (c) version.

The aim of this study is to investigate if TSM can be beneficial to speech coding. The central idea is to compress the time scale of a speech signal prior to coding, enabling usage of less bits, and to expand it by a reciprocal factor after decoding, to come to the

original time-scale. For example, consider a situation where speech is normally coded at 8 kbit/s. If, now, speech is first time-scale modified by say, 25 % (3/4), the amount of samples that need to be encoded will reduce by the same factor. So, the effective bit-rate is expected to go down to 6 kbit/s. The decoder than reconstructs the signal and expands its time-scale by 33 % (4/3) to come to the original time-scale.
This concept is now illustrated in Figure 1.2. All the actions referring to source and channel coding, possible bit transfer, and decoding are represented by the block denoted by *codec*. Both the compressor and the expander receive and process only speech signals.

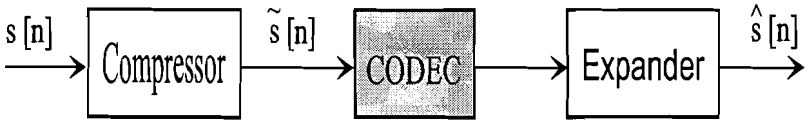

Figure 1.2: A scheme describing usage of TSM in coding applications. Source and channel encoding, bit transfer, and decoding are denoted by CODEC.

It must be pointed out that speech coding in principle allows a more general concept of TSM. So, one could imagine that no particular property of the input speech needs to be maintained during the time-scale compression, as long it can be "recovered" during the time-scale expansion. The adequacy of this approach, however, is dependent on the deployed codec. For instance, not insisting on maintenance of speech-like nature of the compressed signal is quite conceivable with PCM (Pulse Code Modulation), which through sampling and quantization equally encodes any input waveform. On the other hand, the coding approaches making use of perceptual measures and properties of human hearing (eg. masking), such as the CELP (Code Excited Linear Prediction) which is most widely used for efficient waveform coding of speech at moderate and low bit-rates [1], require valid speech signals. Hence, in order to employ an arbitrary codec, the time-scale compression should comply with the definition given at the beginning of this section.
It should be noted that in communication applications in particular, some specific actions may be needed in order to harmonize the data transfer between the compressor and codec and achieve on-line transmission. Here, we do not go in such detail but simply assume that the codec will process compressed speech the same way it would process "regular" speech.
In the sequel, we shall often refer to time-scale compression followed by reciprocal time-scale expansion as **companding** of the time scale, of simply companding.
Summarizing, the main research question could be formulated as: Can the use of TSM help in achieving a better reconstruction quality than conventional coding at comparable bit rates ? Basically, the use of TSM is an alternative in that the bit-rate is lowered by reducing the amount of samples that need to be encoded rather than by using rougher quantization or attempting to modify the coding technique.

[1] CELP has served as an effective tool for compressing speech down to around 6 kbit/s (eg. ITU-T G.732.1).

1.2 Previous work

Although most research concerning TSM seem to be aimed at the applications different than speech coding, using TSM for speech coding has also been reported. In the literature, different approaches can be found, among which several coincide with the concept described in the previous section, where TSM has been used as a pre- and post-processing [5], [6] (par. 2.7). Nevertheless, TSM has been employed as an integral part of the coder as well [3], [8], [17].
In the past, good results were claimed using several TSM methods and speech coding techniques. During the more recent years, improvements have been made to the TSM methods, as well as to the coding techniques. There, the TSM methods and coding techniques have mostly been studied independently from each other.

1.3 Setup of the study

We have seen that deployment of TSM for coding purposes, in contrast to other applications, usually involves several processing stages at the same time, namely compression, coding and expansion. Therefore, the issues of computational complexity, and especially maintenance of speech quality require special attention.
In order to investigate the potential of different TSM techniques, we shall employ a commonly adopted approach sketched in Figure 1.3. In essence, any companding system can be evaluated independently from the codec (1), as well as incorporating an *arbitrary* codec (2).

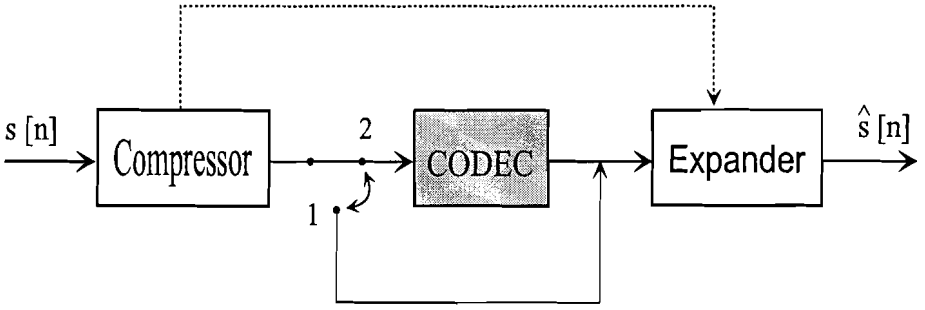

Figure 1.3: Setup of the study.

Besides offering the convenience of employing an arbitrary codec, this approach is also in accordance with a pragmatic reasoning taking into account the expected difficulties in complying with the quality and complexity requirements. So, if the quality of speech acquired along path 1 is not adequate, than it makes not much sense to evaluate path 2, which is expected only to cause further degradation. On the other hand, if path 1 can be made "perceptually lossless", then one can hope that the artifacts introduced by the codec will remain bounded.
Finally, the figure suggests that a communication of some kind between the compressor and the expander may be needed, which is generally not desirable, since it calls for an additional transmission channel.

1.4 Outline of the report

The report is organized as follows. In Chapter 2, an evaluation of different TSM methods and approaches is given, upon which a tool and a direction for further study and experimentation are selected. Some previous work related to TSM for speech coding is also reflected. As a result of the continued study, we have proposed a novel method for time-scale expansion of unvoiced speech, which is introduced in Section 3. In Chapter 4, the selected TSM approaches are combined into a TSM-based coding system, designed and evaluated according to the motivations discussed in the previous section. The report ends with Chapter 5, where some concluding remarks and recommendation for further study are given.

Chapter 2

Evaluation of TSM methods

This chapter gives an overview of TSM approaches that emerge from the literature as most popular and relevant. Several of them have formed the basis for our study of TSM, and are treated in more detail, in separate sections. Other methods are summarized through a global classification, providing also a general reference. At the end of the chapter, an evaluation of the studied methods is presented, serving as a motivation for the succeeding chapters.
In our description of the selected algorithms, we shall omit details concerning their backgrounds and derivation, for which we refer to the literature. At the same time, we shall point out those properties relevant for time-scale companding purposes discussed in the previous chapter.

2.1 Time-domain Harmonic Scaling (TDHS)

The TDHS algorithm was derived [1] as a time-domain approximation of frequency scaling of harmonic signals followed by re-sampling. In essence, it employs an overlap-add procedure, incorporating the *pitch* information to determine its parameters, specifically the window length.
In the following subsections, we explain the algorithm distinguishing between the compression and expansion case. The main aspects of its application to speech are discussed, involving non-stationarity and voicing. In addition, we propose a modified usage of TDHS, which we call "*alternate TDHS*".

2.1.1 Time-scale compression

Let s be a periodic signal having (pitch) period of N_p samples, and $\tilde{s}_c$ its time-scale compressed version. A single compression cycle can now be described by Equation:

$$\tilde{s}_c[n] = s[n] \cdot w[n] + s[n + N_p] \cdot w[n - N_c] \;\; (n = 0, \ldots, N_c - 1) \qquad (2.1)$$

Here, w can be Hanning, or triangular window function having length of N_c samples, as depicted in Figure 2.1. The triangular window, for example, is constructed using formula

2.2. (Based on identity $w_1[n] = w_2[n - N_c]$, Equation 2.1 can be re-written using two additions and one multiplication, making the computations highly efficient.)

$$w[n] = \begin{cases} w_1[n] = 1 - \dfrac{n}{N_c - 1} & (n = 0, \ldots, N_c - 1) \\ w_2[n] = 1 + \dfrac{n+1}{N_c - 1} & (n = -N_c, \ldots, -1) \end{cases} \tag{2.2}$$

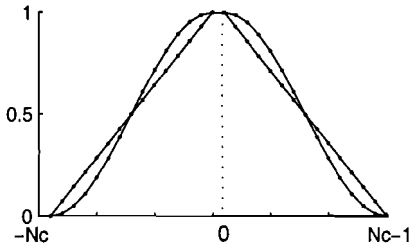

Figure 2.1: The ($2 \cdot N_c$ samples long) windows used for TDHS.

Further, the window length is determined by the pitch period and the scale factor α, as expressed through Equation 2.3.

$$N_c = round\left(\frac{N_p \cdot \alpha}{1 - \alpha}\right) \quad (\alpha < 1) \tag{2.3}$$

Now, Equation 2.1 can be interpreted as follows. An input signal s is being weighted by the trailing edge of w, and by its amplitude-complementary counterpart shifted over $N_p + N_c$ samples. The results of this weighting are added together, producing $\tilde{s}_c$. This way, $N_c + N_p$ input samples are used to generate N_c output samples. This is also illustrated in Figure 2.2.

Apparently, the two outermost samples of s are preserved as such in $\tilde{s}_c$. Namely, they are weighted by 1 (A, resp. A') and added with 0 (B', resp. B). Consequently, the described computations can be repeated with each consecutive $N_c + N_p$ input samples, while maintaining speech continuity.

So far, N_p has been treated as a constant, assuming a periodic input signal. A similar on-line compression process is feasible for *voiced speech*, which is quasi-periodic, if a measurement of its pitch is incorporated. (Several algorithms for this purpose are described in [6], Chapter 6.) Explicitly, a new pitch value N_p needs to be supplied at the beginning of each new compression cycle.

This can be clarified using Figure 2.3. Suppose at $n = 0$ (A_1), a pitch N_{p1} has been measured. From it, given a scale factor α, the corresponding window half-length N_{c1} is computed using formula 2.3. Now, the first $N_{c1} + N_{p1}$ input samples, i.e. segment $\overline{A_1 A_1'}$, can be compressed as described earlier. Then, a new pitch N_{p2} is measured at the position of A_2, defining a new window half-length N_{c2} and enabling the compression of the next $N_{c2} + N_{p2}$ samples, i.e. segment $\overline{A_2 A_2'}$. Segment $\overline{A_3 A_3'}$ is compressed next, etc.

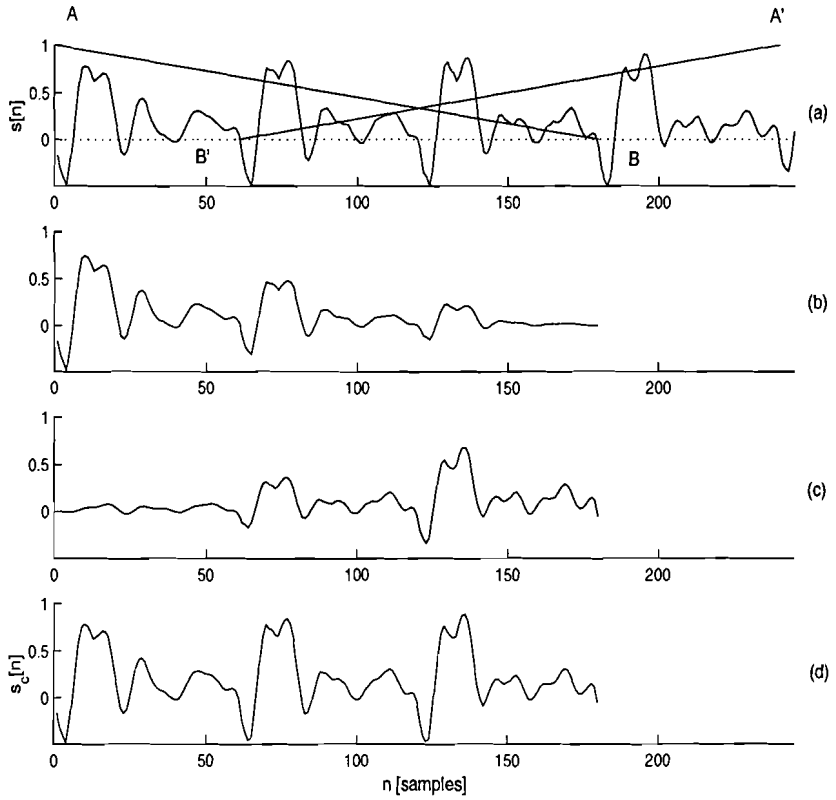

Figure 2.2: TDHS compression cycle: weighting of an original signal (a), weighting results (b, c) and the compressed signal (d) as their sum. ($N_p = 60$, $\alpha = 0.75$, $N_c = 180$.)

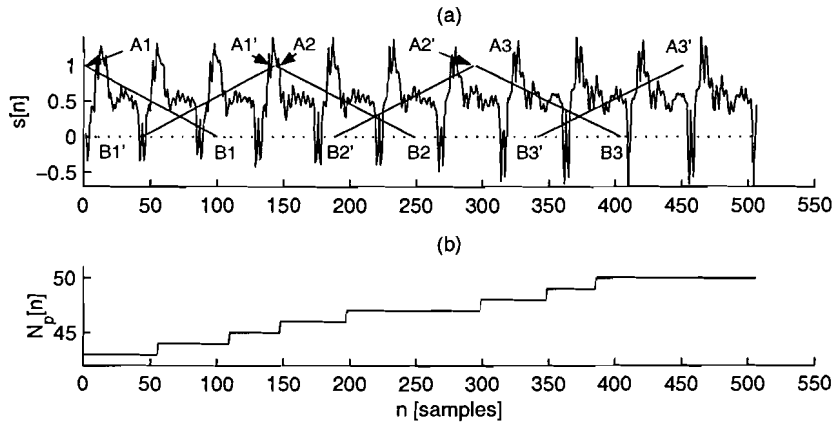

Figure 2.3: On-line TDHS compression: windowing of an input speech segment (a) and the pitch contour corresponding to this segment (b). The pitch contour is interpolated from pitch measurement that is updated after each 5ms. (The sampling frequency amounts 8 kHz.)

Note that the pitch supplied at the beginning of the i-th cycle need not necessarily be measured at that point (A_i). The measurement position could also be advanced towards the centre of the sequence, i.e. the intersections of $\overline{A_i B_i}$ and $\overline{A_i' B_i'}$, for example. (This can be viewed as a "look-ahead" measurement, since the centre points are not known in advance, but dependent on the pitch which is being measured.) This way, more representative pitch values can possibly be obtained for longer, generally less stationary segments.
Furthermore, longer windows are required in proportion to the lowering of the amount of compression. This follows directly from Equation 2.3. Using longer windows implies overlap-adding of longer segments, which are more likely to exhibit non-stationarities. Therefore, a lower compression will generally introduce larger distortions, i.e. smearing of the input waveform shape than higher compression. The smeared waveform will typically appear in the central region of the compressed sequence, since the described windowing puts highest weight on the terminating parts of the original sequence, preserving them the best. An illustrative example is given in Figure 2.4.
The other way around, larger compression implies shorter windows. Eventually, with compression larger than 50 %, some input samples will not be weighted at all ($\alpha < 0.5 \Rightarrow N_c < N_p$). Nevertheless, such skipping of samples will not affect the pitch, as is demonstrated by Figure 2.5.

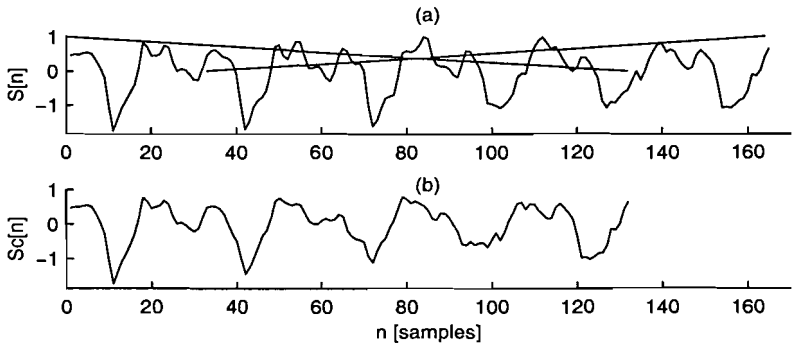

Figure 2.4: 20% compression by TDHS: original- (a) and compressed signal (b)

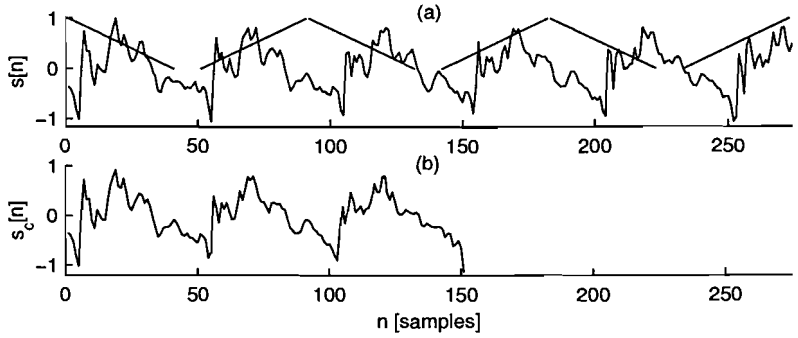

Figure 2.5: 55% compression by TDHS: original- (a) and compressed signal (b)

In practice, *unvoiced speech* can be compressed by TDHS, as well. For the case a voicing detector is incorporated (no pitch is then attributed to unvoiced parts), using a constant window during all consecutive cycles is proposed ([6], par. 2.7). We suggest another approach, which is to use a random pitch. This is motivated by the assumption that a more random character is then given to (spectral) distortions caused by the compression.

2.1.2 Time-scale expansion

Let s again be a periodic signal having pitch N_p, and let $\tilde{s}_e$ be its time-scale expanded version. A single expansion cycle can then be described by Equation 2.4. There, w is the same triangular or Hanning window from Figure 2.1, with its half-length now denoted by N_e and defined by formula 2.5.

$$\tilde{s}_e[n] = s[n] \cdot w[n] + s[n - N_p] \cdot w[n - N_e] \quad (n = 0, \ldots, N_e - 1) \tag{2.4}$$

$$N_e = round\left(\frac{N_p \cdot \alpha}{\alpha - 1}\right) \quad (\alpha > 1) \tag{2.5}$$

Apparently, $N_e + N_p$ samples of s are used to produce N_e samples of $\tilde{s}_e$. This is illustrated in Figure 2.6, having similar interpretation as Figure 2.2. The indexing according to Equation 2.5 starts with Np-th sample, whose position on the time axis corresponding to the vertical projection of A. Thus, the signal that has actually been expanded is the sequence drawn by solid line, while the remaining parts (dash-dotted) are simply copied to the output unmodified, i.e. translated.

Similar as before, the outermost samples of the input sequence are not affected by the computations, since weighted by 1 (A, A') and added with 0 (B', B). Hence, the analogy can be established with the compression case, concerning the suitability of the computations for a continuous on-line processing.

An implementation can be clarified using Figure 2.7. The expansion starts at the position of A_1. (The preceding samples, back to $n = 0$, are translated to the output first.) Let N_{p1} be the pitch supplied there. From it and a given scale factor α, the first window half-length N_{s1} is derived using Equation 2.5. Having defined the window, the above computations can be performed on the previous N_{p1} and the next N_{s1} input samples (sequences $\overline{A_1 B_1}$ and $\overline{B_1' A_1}$), yielding expansion of sequence $\overline{A_1 A_1'}$. Similarly, segment $\overline{A_2 A_2'}$ is expanded next, where the window length is computed from the pitch corresponding to A_2, etc.

Recall from last section that different implementations are conceivable, depending on the position where the pitch is being measured at the beginning of each new cycle.

Another analogy with the compression is that longer windows are needed to accomplish a lower amount of TSM. Furthermore, if the expansion is aimed to reconstruct already compressed time scale, then it requires even longer windows compared to those used for the compression itself. This can be seen by comparing expressions 2.5 and 2.3, after substituting a certain scale factor α and its reciprocal $1/\alpha$, respectively, there. The earlier mentioned inconvenience of longer windows is then further emphasized.

Observe that, in conformity with the idea of expansion, some input samples are re-used in computations. For example, looking at Figure 2.7, it can be seen that segment $\overline{B_2' B_1}$ is used for the expansion of both $\overline{A_1 A_1'}$ and $\overline{A_2 A_2'}$.

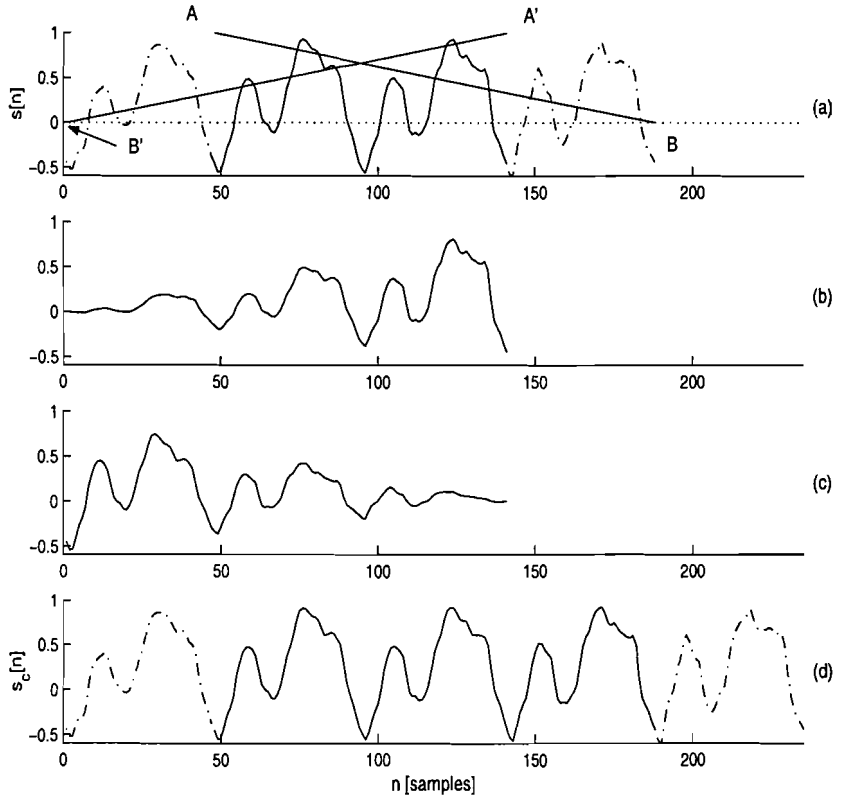

Figure 2.6: TDHS expansion cycle: weighting of an input signal (a), weighting results (b), and the expanded signal (d). Only segment $\overline{AA'}$ of s is actually expanded, by 50 % ($N_p = 47$, $N_e = 141$), while the remaining parts (dash-dotted) are translated.

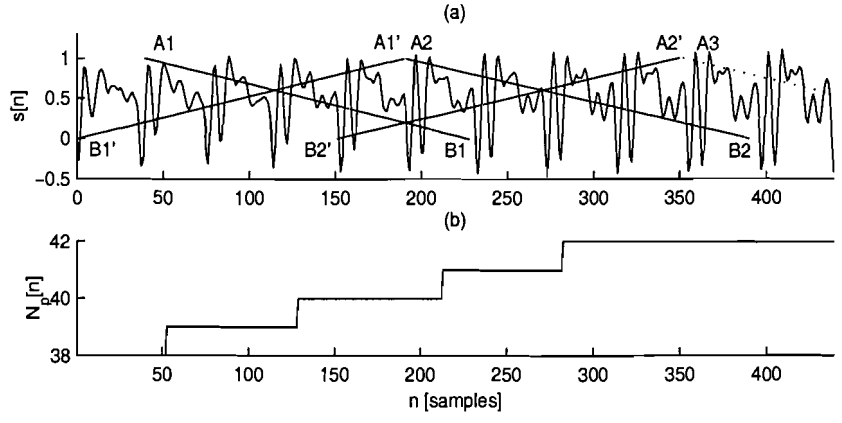

Figure 2.7: On-line TDHS expansion: weighting of a voiced speech segment (a), and the pitch contour corresponding to this segment (b). The pitch contour is interpolated from pitch measurement that is updated after each 5ms. (The sampling frequency amounts 8 kHz.)

Moreover, two modified (by weighting and addition with other weighted samples) versions of the same segment $B_2'B_1$ will be concatenated in the output signal. This will artificially increase auto-correlation over that particular region of the signal. Hence, when similar situation is repeated with other segments, long time auto-correlation will also increase. This is demonstrated by an example, in Figure 2.8.
Unfortunately, when expanding waveforms which normally exhibit low correlation, such as non-stationary or unvoiced speech, this leads to perception of an artificial *reverberation*. In the case of unvoiced speech, artificial *tonality* would perhaps be a more appropriate term. Due to the window length computation in TDHS, averaging of non-stationary waveforms is most likely to occur when expanding already compressed speech.
Again, using random window length with unvoiced speech could be convenient, now serving to possibly "randomize" the tonal noise and therefore reduce the perception of it. Still, the improvements brought by this approach are only limited. (It does not eliminate the cause, which is re-usage of samples, but only varies the length of segments which are reused.)

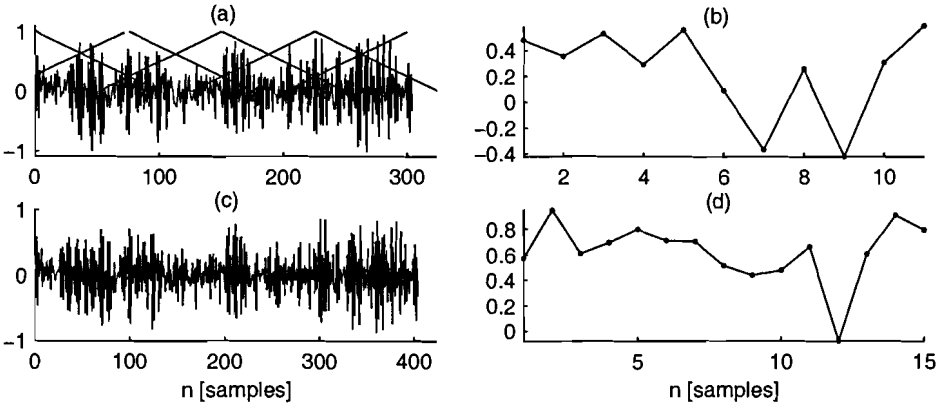

Figure 2.8: An unvoiced excerpt (a) and its 33% TDHS-expanded version (c), obtained by using a fixed pitch $N_p = 25$. For both, the corresponding signals in (b), (d) represent a normalized dot-product of each two consecutive 25 samples long segments.

2.1.3 Alternate TDHS

Here, we propose a modified usage of TDHS, capable of dealing with some previously described inconveniences of TDHS, as will be demonstrated.
The basic idea is to apply TDHS only to some input sequences, while translating the others, i.e. copying them to the output unmodified. This is enabled by the previously explained property of TDHS to preserve the speech continuity at the joints of consecutive input sequences. A special case of such processing is the one where input sequences are time-scale modified and translated alternately, hence *alternate TDHS*. There, the length of consecutive sequences is determined as before, using in the (changing) pitch information. Statistically, if the number of sequences is large, the portion of the input signal which is time-scale modified will be approximately equal to that which is translated. Then, the

overall amount of time-scale modification becomes controllable, which is demonstrated as follows.

Suppose an L_{in} samples long input speech s is submitted to alternate TDHS using a scale factor β during its time-scale modification stages. (Thus, β should replace α in the formulas from previous sections.) Now, let x and y respectively be the total number of samples of s submitted to time-scale modification and translation, assuming $y \approx x$, as explained above. Denoting the total number of samples contained in the output speech by L_{out}, we can now write:

$$L_{in} = x + y \approx 2 \cdot x \tag{2.6}$$

$$L_{out} = round(x \cdot \beta) + y \approx round((\beta + 1) \cdot x) \tag{2.7}$$

Thus, a desired time-scale factor $\alpha = L_{in}/L_{out}$, can be approximately achieved by alternate TDHS using scale factor β determined by:

$$\beta = 2 \cdot \alpha - 1 \tag{2.8}$$

For example, 50% overall expansion ($\alpha = 1.25$) can be realized by alternate TDHS that uses 100% expansion ($\beta = 2$). This is also in accordance with the common intuition that the compression loss due to translation of a certain segment can be compensated by expanding the succeeding segment using a two times larger scale factor. With compression, only moderate scale factors should be attempted. (Otherwise, β could become unacceptably small.)

Keeping in mind previous discussion concerning the inconvenience of long windows and repetition of short-time correlated segments of *non-stationary or unvoiced speech*, some potential benefits of alternate TDHS are now apparent. Firstly, through alternate TDHS, moderate scale factors can still be realized, while avoiding long windows. Secondly, the segments that are "re-used" during two consecutive iterations, i.e. translation followed by expansion and vice versa, may be given substantially different weights during these iterations. This is likely to result in "repetition" of less correlated waveforms. This can be further clarified using Figure 2.9. There, the segments drawn by solid and dotted lines are submitted to the expansion, respectively translation. For example, observe the segment having the length corresponding to the projection of $\overline{ab}$ on the time axis. In the output signal, it will first appear unmodified, due to translation of $\overline{A_1A_1'}$. Then, during the expansion of $\overline{A_2A_2'}$, it will be weighted by $\overline{ab}$ and added with the part of the segment underlying $\overline{A_2A_2'}$, which is more heavily weighted, by $\overline{A_2c}$. This also implies an improved continuation of the original waveform at the joints of translated and compressed sequences.

It should be noted that alternate TDHS shows no particular advantages for usage with voiced speech. On the contrary, a danger arises of introducing a perceivable 'speech variability', due to alteration of regular waveforms (preserved through translation) and less regular waveforms (distorted by compression). Clearly, unvoiced speech is less sensitive to such waveform discontinuity.

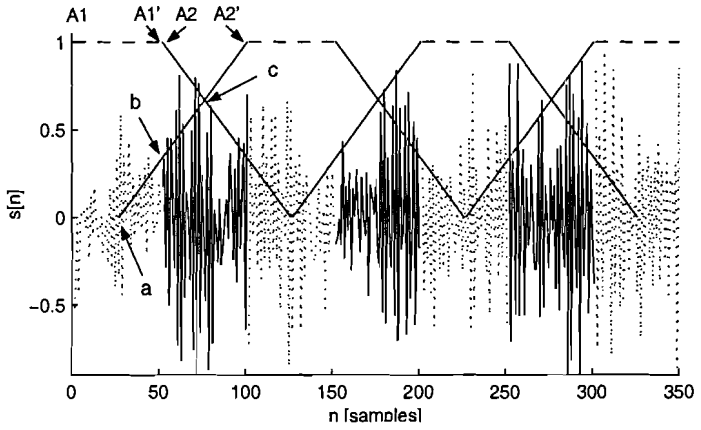

Figure 2.9: 25% expansion of unvoiced speech by alternate TDHS, using fixed pitch $N_p = 25$. The segments drawn by solid (dotted) lines are being expanded (translated).

2.2 Synchronized Overlap-Add (SOLA)

Since it was introduced [4], SOLA has evolved into a popular TSM algorithm, which has been widely used as a benchmark by several related and other approaches. Due to its waveform-based character, the algorithm is applicable to more complex speech signals, such as those produced by multiple speakers or corrupted by background noise, and is fairly applicable to music. In the course of time, several quality-improving modifications to the original version have generally been adopted ([5], [7], [10]). They are also included in this text, which is organized as follows.
In Section 2.2.1, an overlap-add procedure (OLA) is first described, serving as a base for SOLA (and later also for WSOLA). The synchronization procedure is explained in Section 2.2.2. Here, both *compression and expansion* are covered, which only differ in the choice of certain parameters. In 2.2.3, we point out some convergence properties of the algorithm. Finally, Section 2.2.4 explains a method for high quality time-scale reconstruction, employing transmission of the synchronization parameters.

2.2.1 Time-scale modification by overlap-add (OLA)

Let s be an input signal, which is being analyzed in N samples-long overlapping frames. The frames are input at a fixed rate, such that after each *analysis period* of S_a samples a new frame starts being input. Thus, an i-th *analysis frame* is defined as:

$$x_i[n] = s[i \cdot S_a + n] \;\; (n = 0, \ldots, N-1) \;\; (i = 0, \ldots, m) \;\; (S_a < N) \qquad (2.9)$$

A straightforward way to modify the time-scale of s is to output frames x_i at a different rate. Accordingly, time-scale compression or expansion is achieved by initializing the output of a new frame x_i after each *synthesis period* of S_s samples, where S_s is chosen such that $S_s < S_a$, respectively $S_s > S_a$. The amount of time-scale expansion in particular is limited by the condition $S_s < N$, which makes sure that all samples are available. Apparently, some samples originating from different frames will be output at a same

time, and should be averaged in some way. Usually, such time-overlapping signals are *cross-faded*, i.e. respectively weighted by two amplitude complementary functions, then added.
The procedure outlined above can be viewed as an overlap-add (OLA) procedure, and is illustrated in Figure 2.2.1. There, linear weighting function and, for simplicity, one-dimensional signal representation have been used.

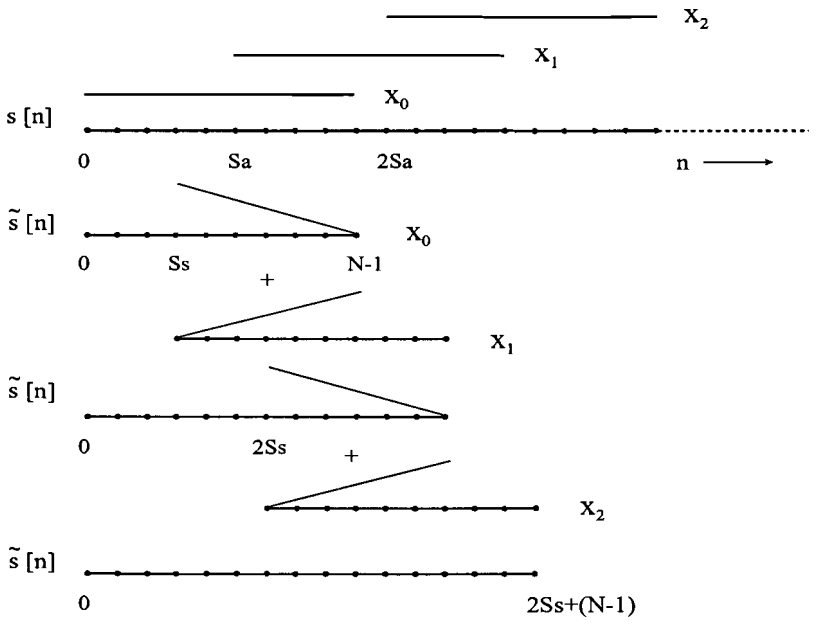

Figure 2.10: TSM by overlap-add: inputting the analysis frames x_i at rate $S_a = N/2$ and outputting them at rate $S_s = 0.75 \cdot S_a$, where the overlapping parts are cross-faded using the linear weighting function.

Now, it can easily be shown that the time-scale factor α achieved in this way will asymptotically approach the value S_s/S_a with increasing number of iterations, i.e. number of frames. Namely, after m iterations, the overall input- and output signal length, respectively denoted by l_{sm} and $l_{\tilde{s}m}$, can be computed as:

$$l_{sm} = (m-1) \cdot S_a + N \tag{2.10}$$

$$l_{\tilde{s}m} = (m-1) \cdot S_s + N \tag{2.11}$$

Hence, for a large m, this yields:

$$\alpha = \frac{l_{\tilde{s}m}}{l_{sm}} \approx \frac{S_s}{S_a} \tag{2.12}$$

Conclusively, it is evident that the waveforms that are averaged in the above described fashion are merely determined by a fixed choice of S_a and S_s. Therefore, OLA will generally fail to preserve the (non-stationary) pitch of an arbitrary input speech signal. One way to overcome this problem is to synchronize the synthesis stage of OLA, as will be explained in the next section.

2.2.2 Synchronizing OLA

Here, previously described OLA is extended by a synchronization procedure, explicitly attempting to preserve waveform shape of an input signal, therefore implicitly maintaining its pitch.

In essence, the analysis frames x_i are input as before, but output such that only similar waveforms will overlap in time, i.e. be averaged. This is achieved by initializing the output of x_i ($\forall i, i = 1, \ldots m$) no longer at pre-defined time instants iS_s, but at $iS_s + k_i$, where k_i is found such that the normalized cross-correlation given by expression 2.13 is maximal for $k = k_i$. As before, s and $\tilde{s}$ respectively denote the input signal and its time-scale modified version, while L denotes the length of the overlap corresponding to a particular k in the given range. Hence, $L = L_i$ for $k = k_i$.

$$R_i[k] = \frac{\sum_{j=0}^{L-1} \tilde{s}[iS_s + k + j] \cdot s[iS_a + j]}{\left(\sum_{j=0}^{L-1} s^2[iS_a + j] \cdot \sum_{j=0}^{L-1} \tilde{s}^2[iS_s + k + j] \right)^{1/2}} \quad (0 \leq k \leq N/2) \tag{2.13}$$

In order to conclude a reliable waveform similarity, any short overlap L_i should be disabled. A suitable threshold is then given by expression 2.14. Note that it may generally impose a smaller range for the lag k in 2.13 than the one given there.

$$L_i \geq L_{min} \approx N/8 \quad (\forall i, i = 1, \ldots, m) \tag{2.14}$$

Having found k_i, the update of the synthesis signal is performed similar as before. The update procedure is summarized by the set of equations 2.15. There, the last equation is aimed to improve speech continuity and control of the synthesis signal length. The interpretation of these equations is clarified by the illustrations in Figure 2.11, featuring again the linear weighting function for cross-fade. (Another popular choice is the Hanning window.) Explicitly, the linear weighting function is defined by Equation 2.16.

$$\tilde{s}[iS_s + k_i + j] = \begin{cases} (1 - w[j]) \cdot \tilde{s}[iS_s + k_i + j] + w[j] \cdot s[iS_a + j] & (j = \overline{0, L_i - 1}) \\ s[iS_a + j] & (j = \overline{L_i, N - 1}) \\ 0 & (j \geq N) \end{cases} \tag{2.15}$$

$$w[j] = \frac{1}{L_i - 1} \quad (0 \leq j < L_i) \tag{2.16}$$

Now, it can easily be seen that SOLA yields a scale factor which still complies with the approximation given in 2.12. Namely, since an i-th update of the synthesis signal starts with iS_s-th sample, appending at most k_i new samples (this is ensured by the last Equation from 2.15), the length $l_{\tilde{s}m}$ in Equation 2.10 will only be larger by k_m, obtained during the last iteration.

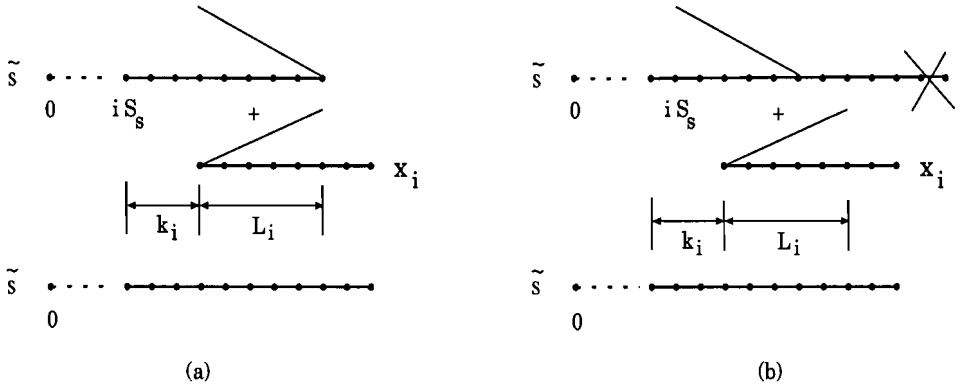

Figure 2.11: Illustration of a single SOLA update of the synthesis signal $\tilde{s}$ by a new analysis frame x_i. After k_i has been computed, x_i is aligned with the $(iS_s + k_i)$-th sample of $\tilde{s}$ and the overlapping samples of $\tilde{s}$ and x_i cross-faded (a). The last case from Equation 2.15 is illustrated in (b). It makes sure that the updated synthesis signal terminates with the "freshest" input samples.

Now, some remarks can be made about the choice of parameters specifying SOLA. With speech signals, the best results are typically associated with N that satisfies: $N > 4 \cdot N_p$, where N_p is the average pitch period [11]. Further, based on the above explanation, S_s should be such that $S_s = \alpha \cdot S_a$, where α is a desired time-scale factor. While S_a is chosen as:

$$S_a = \begin{cases} N/2 & \text{for compression} \\ N/(2\alpha) & \text{for expansion } (\alpha > 1) \end{cases} \tag{2.17}$$

From the above expression it can be seen that the analysis frames will overlap by a considerable amount during the expansion. Consequently, concatenation of short and correlated waveforms in the output signal will often result. Hence, when used for expansion of compressed waveforms, SOLA will exhibit similar reverberation problems already pointed out for TDHS.

2.2.3 Convergence of SOLA

As explained in the previous section, a number of iterations is needed to realize a certain amount of time-scale modification by SOLA. This needs to be taken into account when 'shorter' input signals are concerned. For example, such situations can occur if the algorithm is applied selectively, being interrupted and re-initialized again.
Furthermore, time-scale compression by SOLA will actually start happening only after outputting a frame x_i can no longer start at the position where this frame was 'excised' from the input signal. (By default, this is the point of maximal cross-correlation.) Moreover, only then will the waveforms that are cross-faded be non-identical. Hence, the lag k from Equation 2.13 must satisfied the requirement:

$$k + iS_s < iS_a \Leftrightarrow k < iS_a \cdot (1 - \alpha) \quad (i = 0, \ldots m) \tag{2.18}$$

From the previous equation, taking into account the range of k ($k \leq N/2$), follows that for $\alpha = 0.75$, for instance, the compression will start with 5-th frame.

Note also that k_1 ($i = 1$) will always be equal to $S_a - S_s$, for non-zero vectors, regardless the time-scale factor $\alpha = S_s/S_a$. We shall conveniently make use of this property in our realization of translation of speech transients, as will be explained in section 4.2.

2.2.4 Time-scale reconstruction using transmission of the synchronization parameters.

During SOLA, the N-samples long analysis frames x_i are excised from an input signal at times iS_a, and output at the corresponding times $k_i + iS_s$. Eventually, such modified time-scale can be restored by the opposite process, i.e. by excising N samples long frames $\hat{x}_i$ from the time-scale modified signal at times $k_i + S_s$, and outputting them at times iS_a.

This procedure can be expressed through Equation 2.2.4, where $\tilde{s}$ and $\hat{s}$ respectively denote the TSM-ed and reconstructed version of an original signal s. It is assumed here that $k_0 = 0$, in accordance with the previously introduced indexing of k, starting from $m = 1$. Apparently, $\hat{x}_i[n]$ will be assigned multiple values, i.e. samples from different frames will overlap in time, which, as before, should be averaged by cross-fade. (Rewrite Equations 2.15, using the appropriate notation.)

$$\hat{x}_i[n] = \hat{s}[n + iS_a] = \tilde{s}[n + iS_s + k_i] \;\; (i = \overline{0, m}) \;\; (n = \overline{0, N-1}) \tag{2.19}$$

By comparing the consecutive overlap-add stages of SOLA and the reconstruction procedure outlined above, it can easily be seen that $\hat{x}$ and x_i will generally not be identical. Therefore, these two processes do not exactly form a "1-1" transformation pair. However, the quality of such reconstruction is notably higher compared to merely applying SOLA that uses a reciprocal S_s/S_a ratio ([7], [10]).

It should be noted that concatenation of short-time correlated waveforms remains the basic feature of the expansion mechanism. This is also illustrated in Figure 2.12, showing a possible situation. It can be seen that some samples from x_i which at the same time form the front samples of x_{i+1} will be "repeated" later, after being averaged with other samples by cross-fade. (The 4-th sample of x_i and thus the first sample of x_{i+1} will indeed simply be repeated, since it is weighted by 1 and added with 0.)

In practice, realization of the explained time-scale reconstruction concept requires the knowledge of the synchronization parameters k_i, which brings along the expense of their transmission. The bit-rate of this transmission can be easily estimated, realizing that each SOLA cycle requires S_a new input samples, producing S_s output samples and a value k_i ($0 \leq k_i \leq N/2$) which is sent along with them. Hence, the maximal bit-rate for transmission of k_i's can be derived following the successive equations below. There, T_s represents the sampling period and $\lceil \; \rceil$ means ceiling, i.e. rounding towards the nearest-

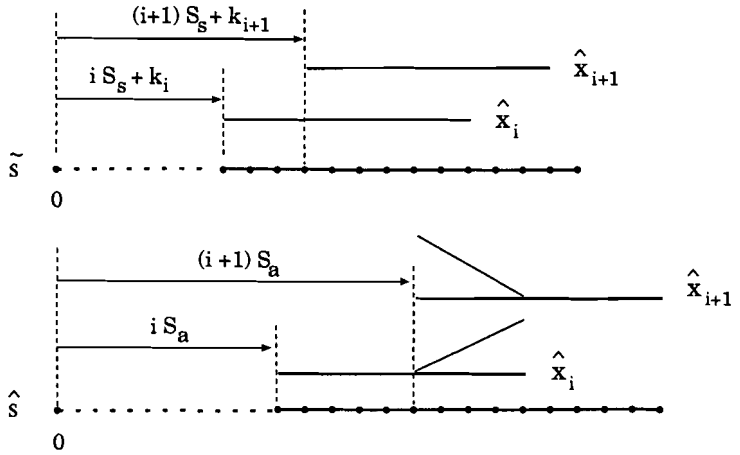

Figure 2.12: Illustration of SOLA time-scale reconstruction. The frames $\hat{x}_i$ and $\hat{x}_{i+1}$ are excised from the compressed signal $\tilde{s}$, where the k_i and k_{i+1} are known in advance. The frames are output at the original rate determined by S_a, using amplitude-complementary cross-fade to average the overlapping samples.

higher integer.

$$frame\text{-}rate = \frac{1}{S_a \cdot T_s} \frac{\text{frames}}{\text{sec}} \tag{2.20}$$

$$k\text{-}quantization = \lceil log_2(\frac{N}{2}) \rceil \frac{\text{bits}}{\text{frame}} \tag{2.21}$$

$$bit\text{-}rate = (\lceil log_2(\frac{N}{2}) \rceil) \cdot \frac{1}{S_a \cdot T_s} \frac{\text{bits}}{\text{sec}} \tag{2.22}$$

For a practical frame length of $N = 200$ samples, for instance, the maximal bit-rate amounts 560 bits/sec.

2.3 Wave Similarity Overlap-Add (WSOLA)

Similar to SOLA, WSOLA [11] is based on the OLA procedure described in Section 2.2.1. Hence, we shall describe it making use of concepts and notation already introduced there.

In contrast to SOLA, WSOLA applies a synchronization mechanism during the analysis stage of OLA, while keeping the synthesis stage as is. Specifically, the input frames are 'selected' based on waveform similarity, while they are output at a constant rate, determined by S_s. Explicitly, the analysis frames x_i are now defined by Equation 2.23.

$$x_i[n] = s[i \cdot S_a + \delta_i + n] \;\; (n = \overline{0, N-1}) \;\; (i = \overline{0, m}) \;\; \delta_i \in [-\Delta, \Delta] \tag{2.23}$$

There, δ_i is found from the given range such that it produces a most suitable x_i for an ith synthesis iteration (overlap-add), as will be explained in the following.

This new definition of x_i's in principle still allows inputting new samples at a constant rate, determined by S_a, while only making the amount of overlap between different x_i's variable. Similar to SOLA, the amount of TSM achieved after a large number of iterations (frames) can be approximated by a ratio of S_a and S_s.
The procedure of selecting the analysis frames can be clarified aided by Figure 2.13. (See also Figure 2.2.1.) Assume that x_{i-1} was the last analysis frame excised from an input signal, which we started outputting at time instant $(i-1)S_s$. To keep the synthesis frame rate constant, the output of a next analysis frame is required to start S_s samples later, i.e. at time instant iS_s. Apparently, if no TSM would be attempted, the most suitable frame to be output would be $\tilde{x}_i$, since it starts exactly S_s samples after x_{i-1}. Thus, to approximate a time-scale factor $\alpha = S_s/S_a$, while maintaining waveform continuity as much as possible, an x_i should be excised near time instant iS_a, such that it resembles $\tilde{x}_i$ the most. Equivalently, a tolerance δ_i from the given range must be found, such that a similarity measure $c(i, \delta)$ between frames x_i and $\tilde{x}_i$ is maximal for $\delta = \delta_i$. After x_i has been selected and started being output, the same procedure is repeated with x_i playing the role of x_{i-1} from the previous step.

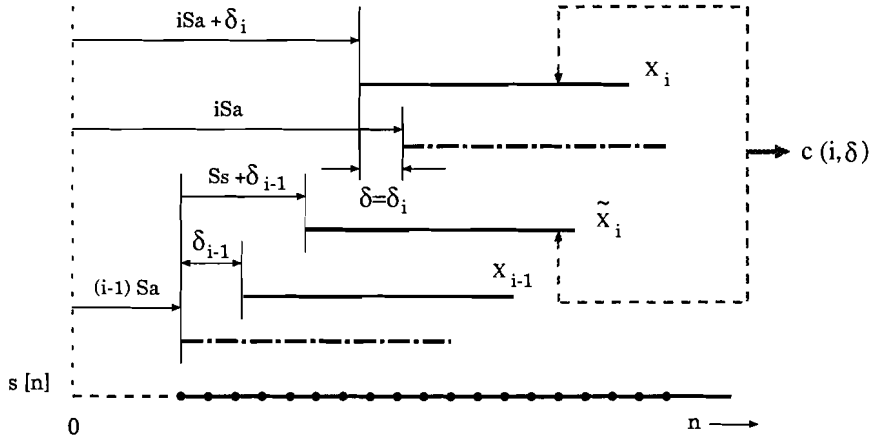

Figure 2.13: Finding an optimal analysis frame in WSOLA. In the absence of synchronization, i.e. according to OLA, the dash-dotted frames would be the analysis frames. (See also Figure 2.2.1.)

Some similarity measures that can be applied successfully in the described procedure are given by the equations below, with N again denoting the frame length. They are, respectively, *cross-correlation-*, *normalized cross-correlation* and *cross-AMDF coefficient.*

$$c_c(i,\delta) = \sum_{n=0}^{N-1} s(n + (i-1)S_a + S_s + \delta) \cdot s(n + iS_a + \delta) \tag{2.24}$$

$$c_n(i,\delta) = (c_c(i,\delta)) \cdot \left(\sum_{n=0}^{N-1} x^2(n + (i-1)S_a + \delta) \right)^{-1/2} \tag{2.25}$$

$$c_A(i,\delta) = \sum_{n=0}^{N-1} |s(n + (i-1)S_a + S_s + \delta) - s(n + iS_a + \delta)| \tag{2.26}$$

The synthesis procedure can be summarized by Equation 2.27, where w is a symmetric window. In order to ensure that the overlap-add is performed in amplitude-complementary fashion, i.e. using amplitude complementary weighting of the overlapping segments, w must satisfy Equation 2.28 (It represents the sum of all windows obtained by consecutive shifting of w by S_s samples.)

$$\tilde{s}[n] = \sum_j w[n - jS_s] \cdot s[n + jS_a - jS_s + \delta_j] \tag{2.27}$$

$$\sum_j w[n - jS_s] = 1 \tag{2.28}$$

Based on the above explanation, several analogies with respect to SOLA can be concluded. In fact, much of the discussion and formulas presented for SOLA, concerning the issues of convergence and time-scale reconstruction, can be adopted, using the appropriate notation. Still, due to the fact that the signal frames are selected at the input, WSOLA may offer some algorithmic advantages. For example, since the synthesis is performed at a constant rate, the same weighting function can be used in all cross-fade computations.

2.4 A general overview of other methods

Most TSM methods exploit different principles at the same time, which makes it difficult to categorize them in an universal way. Below, we suggest a general classification, attempting to confine this overlap between different methods.

A. Waveform approaches

   1. Time-domain, overlap-add methods
   2. STFT - based (mixed) methods
   3. Frequency-domain methods
   4. Other

B. Parametric approaches

   1. Methods based on harmonic (sinusoidal) modelling
   2. Methods based on LPC+noise modelling

In [12], a good reference is provided to many methods from the above categories. Hence, we give only a brief and global description of different approaches, including also some more recent methods.
The **time-domain methods** are usually based on the principle of performing overlap-add on short-time signals. They differ in the way these signals are synchronized. So, some methods make explicit use of the pitch information, such as the popular TD-PSOLA [12], while the others employ waveform similarity measures, like (normalized) correlation, for example. A widely popular method from the last category is SOLA, of which several

modifications have been proposed, attempting to reduce the computational cost by employing other similarity measures, like the one described in [14]. There also exist methods which exploit waveform similarity in a more or less different fashion, like PICOLA (part of MPEG4 standard) and MBROLA.

The goal of most **STFT-based methods** is to iteratively synthesize time-scaled signal whose STFT resembles the STFT of the original signal. So, during the analysis process, Fourier transformation is performed on frames of the input signal, which are excised at a certain rate. The output signal is then synthesized by overlap-adding the signals derived from the inverse Fourier transforms of the original frames, at a new rate. These signals are usually derived taking into account additional requirements, which are designed to improve the magnitude and phase continuity of the STFT. A detailed study of STFT-based TSM can be found in [2], [13].

The **frequency-domain methods** can often be approximated by overlap-add of time-domain signals. A good example is time-domain approximation of harmonic scaling (TDHS). However, some methods perform some of their operation on frequency-domain signals, applying more complicated schemes. There also exist methods which use similar approach in time- and frequency domain, such as TD-PSOLA and FD-PSOLA [12]. Usually, frequency-domain methods better serve the purpose of pitch-scale modification.

The basic idea of **parametric methods** is to approximate parts of the original waveform (frames) by sets of parameters, creating an alternative signal representation. This alternative signal can be viewed as consisting of multi-dimensional vectors (sets of parameters), where the consecutive vectors will be spaced along the time axis with the shifts which are determined by the frame rate. Typically, TSM is obtained by down- or up-sampling of this alternative multi-dimensional signal in time, then using its samples, i.e. sets of parameters, as sources for the synthesis of new frames (of the same length).

Widely studied in the literature are parametric methods for voiced speech. Commonly, voiced speech is modelled as a sum of sinusoids with slowly time-varying amplitudes and instantaneous frequencies, not necessarily assumed to be harmonically related [12]. The phase information is often omitted, and interpolated from the frequency (frequency is the derivative of the phase), which is justified by the bit-rate requirements in some applications. However, this usually creates the problem of phase incoherence, for which remedies have been proposed.

Finally, parametric modelling of unvoiced speech and some possible TSM approaches based upon it are treated in more detail in Chapter 3.

2.5 Selection

Generally, time-domain methods, parametric methods using sinusoidal modelling and STFT-based approaches are most widely studied in the literature. Among them, the time-domain methods are commonly known to produce time-scale modified speech signals of high(est) subjective quality. Therefore, our study has primarily targeted these methods.

From this group, several methods emerge as most popular. These are *SOLA, PSOLA, TDHS* and some others. From our experience, PSOLA and PICOLA (part of the MPEG-4 standard), in particular, have already been known to give a high performance when used

for time-scale compression or expansion only. However, they have been judged as less suitable for companding of the time-scale, relevant for the coding purposes of our interest, as explained in Chapter 1. At the same time, applying *TDHS* and *SOLA* for such purposes, had already been reported ([5], [6], par. 2.7). These considerations have prevailed as a motivation for choosing these two methods as a base for further study (also taking into account the time constraints bound to the project).

Our initial simulations were designed to examine the performance of the particular approaches when used for time-scale companding in absence of a speech codec. They have confirmed the expectations from previous sections. In particular, *TDHS* proved to perform better in proportion to the increasing compression (implying shorter windows). Typically, a fairly good performance results when a factor 2 (50 % compression, followed by 100 % expansion) is used. The "reverberation artefact" in voiced speech had occasionally be observed, while a notable quality degradation has been associated to unvoiced speech, where a high tonal noise is introduced. Alternate TDHS has proven (reasonably) capable of reducing this noise, while it introduced artifacts in the voiced parts. A disadvantage of TDHS remains, however, that a highly compressed speech would be an inadequate input to the speech coders which are only capable of addressing a specific range and dynamics in speech. This problem could possibly be alleviated by oversampling prior to the expansion.

In contrast, *SOLA*, when employing *transmission of k's*, has given a good performance for a range of moderate to high scale factors (up to 35%). The transmission of the synchronization parameters has proven as an efficient tool for adapting the time-scale compression to the time-scale compressed speech. A notable reduction of the "reverberation artifact" has emerged as an important improvement. Still, similar problems of tonal noise in unvoiced speech have been encountered, which are also common to all time-domain expansion methods exploiting the principle of repeating correlated short-time waveforms. In addition, artifacts have occasionally been observed with all methods, which could be well associated with degradation (smearing due to averaging) of speech transients. The transients refer to regions marking voicing transitions.

Inspired by these results, we have chosen the following approach. For compression, and for expansion of voiced speech parts, we shall use SOLA, while seeking an alternative method for the expansion of unvoiced parts. In the next chapter we present such a method. To prevent smearing of transients, we shall use translation, which has already been proposed in [15]. Since we are not targeting very high compression rates, we shall initially translate only voiced onsets, as will be explained in Section 4.3. This we justify by a common observation that voiced onsets usually exhibit more abrupt discontinuities of signal energy (amplitude) compared to voiced offsets, which makes them more "vulnerable" to smearing. Moreover, since it bases its averaging on waveform similarity, SOLA has proven capable of preserving smooth transients well (even when larger compression factors are used).

The resulting TSM-based speech coding system incorporating these principles is described and evaluated in Chapter 4.

Chapter 3

Parametric time-scale expansion of unvoiced speech

Here, we present a method for time-scale expansion of unvoiced speech, designed to overcome the problem of artificial tonality introduced by the "repetition" mechanism which is inherently present in all time-domain methods. The central idea is to lengthen the time-scale by inserting an appropriate amount of synthetic noise that reflects the spectral- and energy properties of the input sequence. The estimation of these properties is based on *LPC* (Linear Predictive Coding) and variance matching. We choose to derive the model parameters directly from the input signal, which may be an already compressed one, avoiding the necessity for their transmission. This approach relies on the reasonable assumption that only a limited distortion of the above mentioned properties of an unvoiced sequence is caused by a compression of its time-scale.

In the following subsections, the method is developed, starting from some general concepts that are incorporated in it (Section 3.1), or have served as a motivation (Section 3.2).

3.1 Parametric modelling of unvoiced speech

Linear predictive coding is a widely applied method for speech processing, employing the principle of predicting the current sample from a linear combination of previous samples. It is described by Equation 3.1, or, equivalently, by its z-transformed counterpart 3.2. In Equation 3.1, s and $\hat{s}$ respectively denote an original signal and its LPC estimate, and e the prediction error. Further, M determines the order of prediction, and a_i are the LPC coefficients. These coefficients are derived by some of the well-known algorithms ([6], 5.3), which are usually based on least-squares error (LSE) minimization, i.e. minimization of

$\sum_n e^2[n]$.

$$s[n] = \hat{s}[n] + e[n] = \sum_{i=1}^{M} a[i]s[n-i] + e[n] \tag{3.1}$$

$$H(z) = \frac{S(z)}{E(z)} = \frac{1}{1 - \sum_{i=1}^{M} a[i] \cdot z^{-i}} = \frac{1}{A(z)} \tag{3.2}$$

Using the LPC coefficients, a sequence s can be approximated by the synthesis procedure described by Equation 3.2. Explicitly, the filter $H(z)$ (often denoted as $1/A(z)$) is excited by a proper signal e, which, ideally, reflects the nature of the prediction error. In the case of *unvoiced speech*, a suitable excitation is normally distributed zero-mean noise.
Eventually, to ensure a proper amplitude level variation of the synthetic sequence, the excitation noise is multiplied by a suitable gain G. Such a gain is conveniently computed based on *variance* matching with the original sequence s, as described by Equations 3.3. Usually, the mean value $\bar{s}$ of an unvoiced sound s can be assumed to be equal to 0. But, this need not be the case for its arbitrary segment, especially if s had been submitted to some time-domain weighted averaging (for the purpose of time-scale modification) first.

$$G = \sqrt{\frac{\sigma_s^2}{\sigma_e^2}} \equiv \frac{\sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} (s[n] - \bar{s})^2}}{\sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} (e[n] - \bar{e})^2}} \qquad (\bar{s} = \frac{1}{N} \cdot \sum_{n=0}^{N-1} s[n],\ \bar{e} = 0) \tag{3.3}$$

The described way of signal estimation is only accurate for *stationary* signals. Therefore, it should only be applied to speech *frames*, which are quasi-stationary. When *LPC computation* is concerned, speech segmentation also includes windowing, which has the purpose to prevent smearing in the frequency domain. This is illustrated in Figure 3.1, featuring Hamming window, where N denotes the frame length (typically 15-20ms) and T the analysis period.

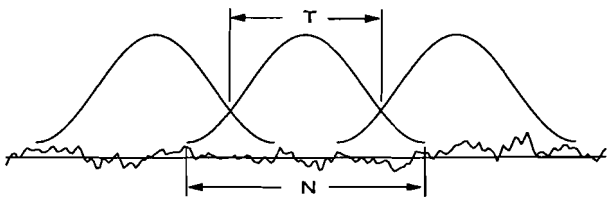

Figure 3.1: Segmentation and windowing of unvoiced speech for LPC computation.

Finally, note that the gain- and LPC computation need not necessarily be performed at the same rate.

3.2 TSM by adjusting the synthesis rate

A possible way to realize time-scale modification of unvoiced speech utilizing the previously discussed parametric modelling is to perform the synthesis at a different rate than the analysis.
In Figure 3.2, time-scale expansion that exploits this idea is illustrated. During the analysis stage, the LPC coefficients and the gain are derived from the input signal, here at a same rate. Specifically, after each period of T samples, a vector of LPC coefficients $\underline{a}$ and a gain G are computed over the length of N samples, i.e. for an N-samples long frame. In a way, this can be viewed as defining a 'temporal vector space' V, according to Equation 3.4, which is for simplicity shown as a two-dimensional signal.

$$\underline{V} = \underline{V}(\underline{a}(t), G(t)) \;\; (\underline{a} = [a_1, ..., a_M] \;,\; t = nT \;,\; n = 1, 2, \ldots) \tag{3.4}$$

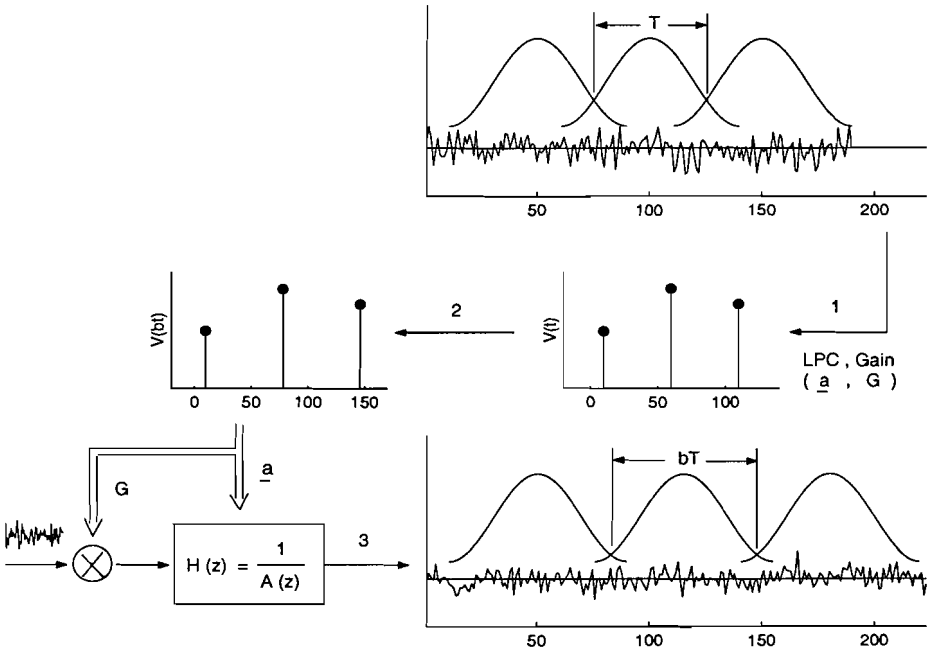

Figure 3.2: A parametric time-scale expansion of unvoiced speech by factor $b > 1$. The model parameters are derived at a rate $1/T$ (1), and used for the synthesis (3) at rate $1/bT$. The Hamming windows deployed during the synthesis are only used to illustrate the rate change. In practice, power complementary weighting would be most most appropriate.

Now, to obtain time-scale expansion by a scale factor b ($b > 1$), this vector space is simply 'down-sampled' by the same factor, prior to the synthesis. Explicitly, after each

period of bT samples, an element of V is used for the synthesis of a new N samples-long frame. Hence, compared to the analysis frames, the synthesis frames will be overlapping in time by a smaller amount. To demonstrate this, we have marked the frames by using the Hamming windows again. In practice, the overlapping parts of the synthesis frames should rather be averaged by applying the power-complementary weighting instead, deploying the appropriate windows for that purpose. Eventually, time-scale compression could be achieved in a similar way, by performing the synthesis at a faster rate than the analysis.

It must be pointed out that the output signal produced by applying this approach is an entirely synthetic signal. As a possible remedy to reduce the artifacts, which are usually perceived as an increased noisiness, a faster update of the gain could serve. A more effective approach, however, is to reduce the amount of synthetic noise in the output signal. In the case of time-scale expansion, this can be accomplished as explained in the next section.

3.3 Time-scale expansion by noise insertion

Instead of synthesizing whole frames at a certain rate, an appropriate and smaller amount of noise can be used to *lengthen* the input frames. The additional noise for each frame is obtained similar as before, namely from the models (LPC coefficients and the gain) derived for that frame. When expanding compressed sequences, in particular, the window length for LPC computation may generally extend beyond the frame length. This is principally meant to give the region of interest a sufficient weight. There, a compressed sequence which is being analyzed is assumed to have sufficiently retained the spectral and energy properties of the original sequence it has been obtained from.

Let us now explain the entire procedure of such time-scale expansion, using the illustration from Figure 3.3. First, an input unvoiced sequence $s[n]$ is submitted to segmentation into frames. Each of the L-samples long input frames $\overline{A_i A_{i+1}}$ will be expanded to a desired length of L_E samples ($L_E = \alpha \cdot L$, where $\alpha > 1$ is the scale factor). In accordance with the earlier explanation, the LPC analysis will be performed on the corresponding, longer frames $\overline{B_i B_{i+1}}$, which, for that purpose, are windowed.

The time-scale expanded version of one particular frame $\overline{A_i A_{i+1}}$ (denoted by s_i) is then obtained as follows. A L_E samples long, zero-mean and normally-distributed ($\sigma_e = 1$) noise sequence is shaped by the filter $1/A(z)$, defined by the LPC coefficients derived from $\overline{B_i B_{i+1}}$. Such shaped noise sequence is then given gain and mean values which are equal to those of frame $\overline{A_i A_{i+1}}$. Computation of these parameters is represented by blocks "G" and "$\frac{1}{L} \cdot \sum$". Next, frame $\overline{A_i A_{i+1}}$ is split into two halves, namely $\overline{A_i C_i}$ and $\overline{C_i A_{i+1}}$, and the additional noise is inserted in between them. This added noise is excised from the middle of the previously synthesized noise sequence of length L_E. Practically, these actions can be achieved by proper windowing and zero-padding, giving each sequence the same length of L_E samples, then simply adding them all together.

In addition, the windows drawn by dashed lines suggest that averaging (cross-fade) can be performed around the joints of the region where the noise is being inserted. Still, due to the noise-like character of all involved signals, possible (perceptual) benefits of such

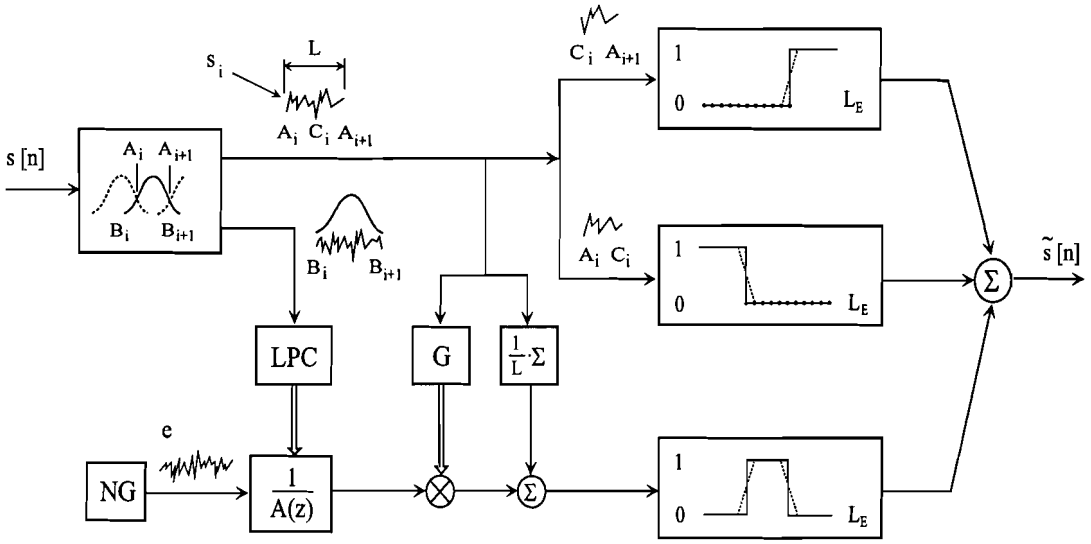

Figure 3.3: Time-scale expansion of unvoiced speech by adding appropriately modelled synthetic noise. The noise is inserted in the middle of the input frame, while its left- and the right half are preserved

'smoothing' in the transition regions remain bounded.
In Figure 3.4, the approach explained above is demonstrated by an example. First, TDHS compression has been applied to an original unvoiced sequence $s[n]$, producing $s_c[n]$ as result. The original time-scale has then been re-instated by applying expansion to $s_c[n]$. The noise insertion is made apparent by zooming in on two particular frames.
Finally, some remarks can be made concerning the above described way of noise insertion. Apparently, it is in accordance with the usual way of performing LPC analysis, employing the Hamming window. So, since the central part of the frame is given the highest weight, inserting the noise in the middle seems logical. However, if the input frame marks a region close to an acoustical event, like a voicing transition, then inserting the noise in a different way may be more desirable. For example, if the frame consists of unvoiced speech gradually transforming into a more 'voiced-like' speech, then insertion of synthetic noise closer to the beginning of the frame (where the most noise-like speech is located) would be most appropriate. An asymmetrical window putting the most weight on the left part of the frame could then be suitably used for the purpose of LPC analysis.

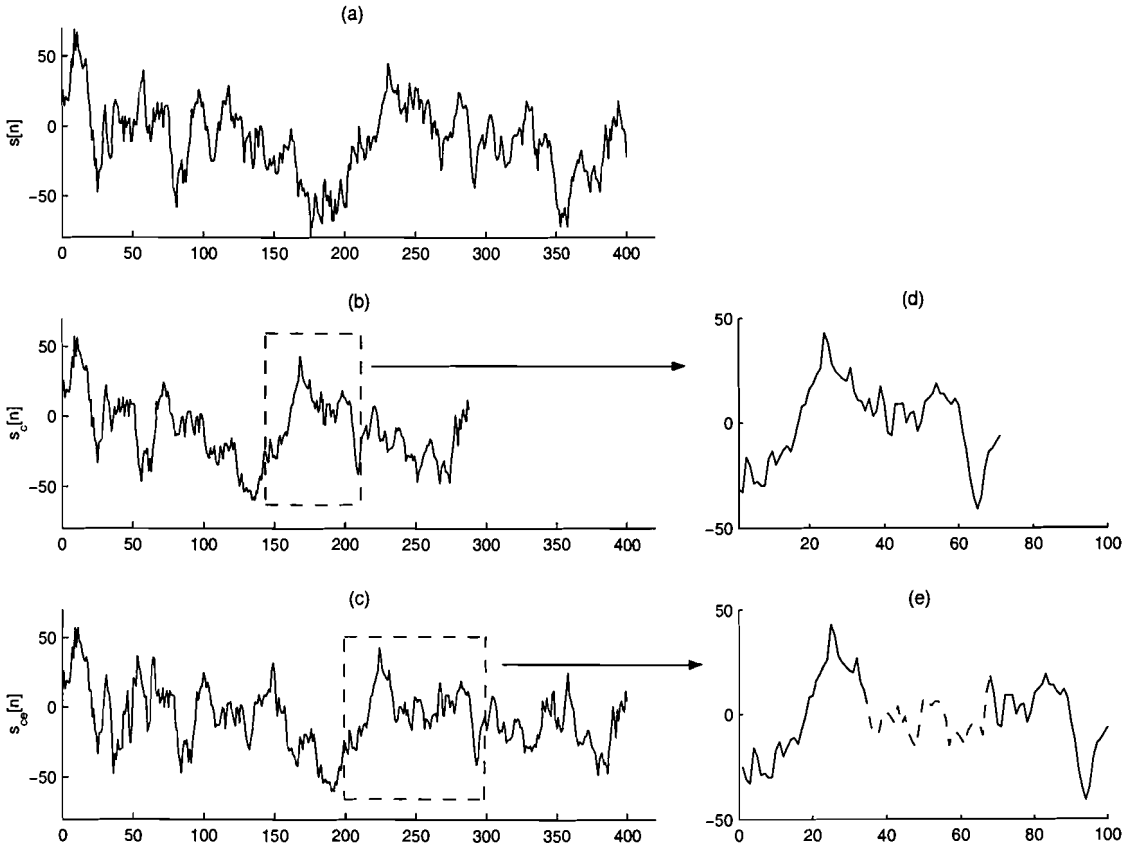

Figure 3.4: An example of time-scale companded unvoiced speech, where the noise-insertion method has been used for the purpose of time-scale expansion, and TDHS for the purpose of time-scale compression. $s[n]$ (a), $s_c[n]$ (b) and $s_{ce}[n]$ (c) represent the original-, compressed, and the output speech, respectively. The noise insertion has ben applied to consecutive 72 samples long frames of $s_c[n]$ (using 150 samples long Hamming windows for the LPC analysis). One such frame is shown in (d), and its expanded version in (e), where the inserted noise can be distinguished as the signal part drawn by the dashed line.

Chapter 4

A speech coding system incorporating TSM

In this chapter, we describe a TSM-based speech coding system designed to evaluate previously explained concepts and motivations. The system comprises of a (tunable) compressor and a corresponding expander, allowing an arbitrary speech codec to be placed in between them. The time-scale companding is realized combining SOLA, parametric expansion of unvoiced speech and the additional concept of translating *voiced onsets*. Although a rather practical frame-based processing approach is employed, no special attempt has been made at this point to address all the aspects of achieving an optimal (real-time) realization.
In the following sections, details concerning the system setup and realization of its TSM stages are given. At the end of the chapter, a performance evaluation is presented, including a comparison with some standard speech coders.

4.1 Description of the system

In figure 4.1, a global set-up of the system is shown. The signal flow can be described as follows. The incoming speech is submitted to buffering and segmentation into frames, to suit the succeeding processing stages. Namely, by performing a voicing analysis on the buffered speech (inside the block denoted by 'V/UV') and shifting the consecutive frames inside the buffer, a flow of the voicing information is created, which is exploited to classify speech parts and handle them accordingly. Specifically, voiced onsets are translated, while all other speech is compressed using SOLA. The out-coming frames are then passed to the codec (A), or bypass the codec (B) directly to the expander. Simultaneously, the synchronization parameters are transmitted through a side channel. They are used to select and perform a certain expansion method. That is, voiced speech is expanded using SOLA frame shifts k_i, as explained in Section 2.2.4, while the unvoiced speech is expanded using the parametric method from 3.3. Note that the translated speech segments are used to realize the expansion, instead of simply being copied to the output. This will be clarified in Section 4.3. Through suitable buffering and manipulation of all received data, a synchronized processing results, where each incoming frame of the original speech

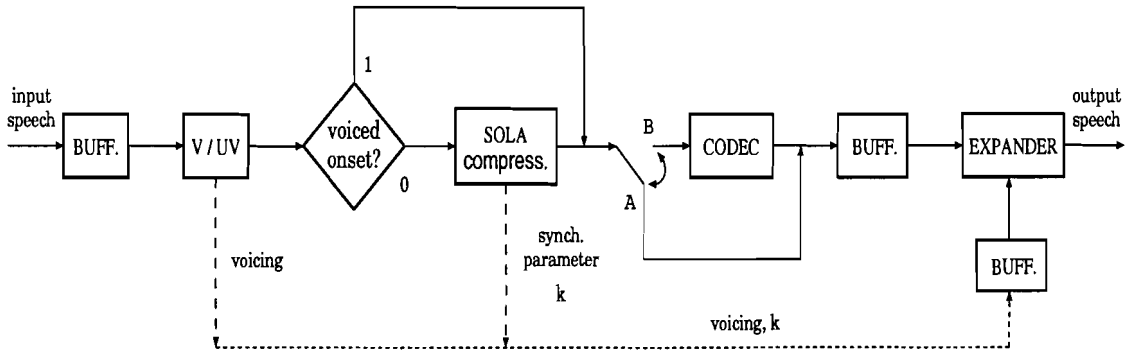

Figure 4.1: Speech coding system incorporating TSM. All speech is compressed using SOLA, except voiced onsets, which are translated. To synchronize the companding additional information is transmitted through a side-channel.

will produce a frame at the output (after an initial delay).
As already discussed in Section 1.1, at this point we assume that the data transfer between the compressor and codec does not require additional synchronization. (In practice, such synchronization may not be trivial, especially if a constant bit-rate is desired, because the compressed and translated frames will have different lengths.)
It must also be pointed out that a voiced onset is simply detected as any transition from unvoiced-like to voiced-like speech. The implications of such detection will be discussed in Section 4.3.
Finally, note that the voicing analysis could in principle be performed on the compressed speech, as well, what would eliminate the need for transmitting the voicing information. However, such speech would be rather inadequate for that purpose, because relatively long analysis frames must usually be analyzed in order to obtain reliable voicing decisions.

4.2 The compressor

Let us first explain the management of the input speech buffer, which is sketched in figure 4.2. The speech contained in the buffer at a certain time is represented by segment $\overline{0A_4}$. The segment $\overline{0M}$, underlying the Hamming window, is submitted to voicing analysis, providing a voicing decision which is associated to V samples in the centre. The window is only used for illustration, and does not suggest the necessity for weighting of the speech. (The details of the algorithm which we deploy for this purpose can be found in [16].) Hence, the acquired voicing decision is attributed to S_a samples long segment $\overline{A_1A_2}$, where $V \leq S_a$ and $|S_a - V| \ll S_a$. Further, the speech is segmented in S_a samples long frames $\overline{A_iA_{i+1}}$ $(i = 0, \ldots, 3)$, enabling a convenient realization of SOLA and buffer management. Specifically, $\overline{A_0A_2}$ and $\overline{A_1A_3}$ will play the role of two consecutive SOLA analysis frames x_i and x_{i+1}, while the buffer will be updated by left-shifting of frames $\overline{A_iA_{i+1}}$ $(i = 0, 1, 2)$ and putting new samples at the 'emptied' position of $\overline{A_3A_4}$.
A more exact definition of the buffer length and its management can be given using a suitable notation. So, let the speech in the buffer be denoted by b, the signal containing

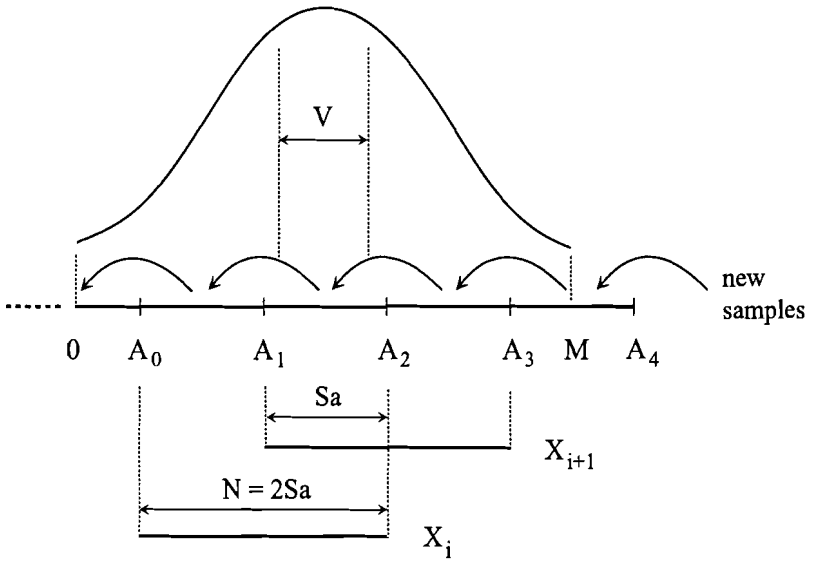

Figure 4.2: The buffer holding the input speech is updated by left-shifting of the S_a samples long frames. The voicing analysis is performed on $\overline{0M}$ and the result of it associated to $\overline{A_1A_2}$. Frames x_i and x_{i+1} are used as consecutive SOLA analysis frames.

the "freshest" (not yet read) input speech samples by s, and the length of a segment, say $\overline{ab}$ by $d(\overline{ab})$. Assuming $d(\overline{OM}) = L \leq 5 \cdot S_a$ and $d(\overline{OA_0}) = d(\overline{A_3M}) = l$, we can write:

$$3 \cdot S_a + 2 \cdot l = L \Rightarrow l = \frac{L - 3 \cdot S_a}{2} \tag{4.1}$$

with typical values $L = 400$ and $S_a = 100$ ($V = 80$) samples. A single update by a new frame is now summarized by the equations given below, where $\lfloor\ \rfloor$ means "flooring", i.e. rounding towards zero.

$$b[n] = b[n + S_a] \quad (n = 0, \ldots l - 1) \tag{4.2}$$

$$b[m + l + (j-1) \cdot S_a] = (1 - \lfloor\frac{j}{4}\rfloor) \cdot b[m + l + j \cdot S_a] + (\lfloor\frac{j}{4}\rfloor) \cdot s[m] \quad (m = 0, \ldots S_a - 1) \quad (j = 1, \ldots 4) \tag{4.3}$$

Keeping in mind the above explained, the compression can easily be described using figure 4.3, where four initial iterations are illustrated. The flow of the input- and output speech can be respectively followed on the right- and left side of the figure, where some familiar features of SOLA are apparent. Among the input frames, voiced ones are marked by "1" and unvoiced by "0".
Initially, the buffer contains a zero signal. Then, a first frame $d(\overline{A_3A_4})$ is read, in this case announcing a voiced segment. Note that the voicing of this frame will be known only after it has arrived at the position of $\overline{A_1A_2}$, in accordance with the earlier described way of performing the voicing analysis. Thus, the algorithmical delay amounts $3S_a$ samples. On the left side, the continuously changing gray-painted frame, hence *synthesis frame*, represent

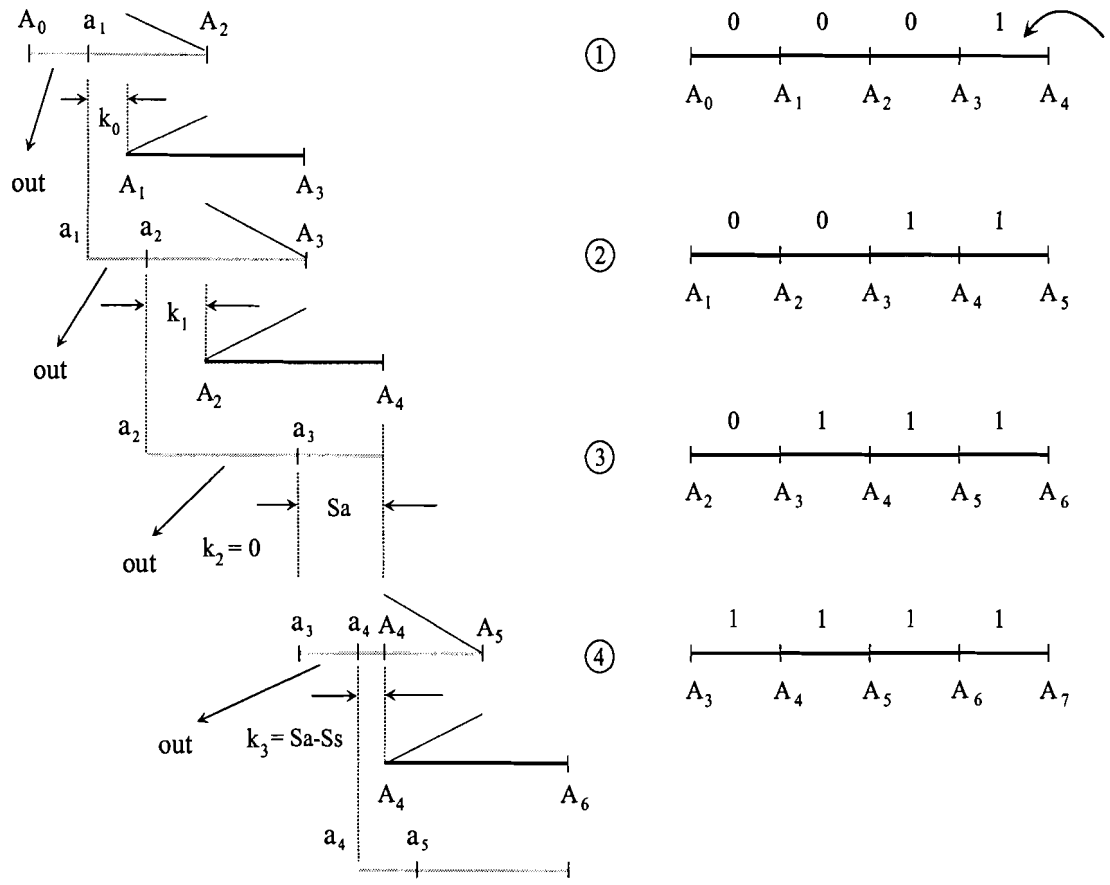

Figure 4.3: The flow of the input- (right) and output speech (left) in the compressor. The voiced and unvoiced frames are denoted by 1, respectively 0. The gray painted frames hold the remaining (not yet output) synthesized speech at the beginning of a new iteration.

the front samples of the buffer holding the output (synthesis) speech at a particular time. (As will become clear, the minimal length of this buffer is $(k_i)_{max} + 2S_a = 3S_a$ samples.) In accordance with SOLA, this frame is updated by overlap-add with the consecutive analysis frames, at the rate determined by S_s ($S_s < S_a$). So, after first two iterations, the S_s samples long frames $\overline{A_0 a_1}$ and $\overline{a_1 a_2}$ will consecutively have been output, as they become obsolete for new updates, respectively by the analysis frames $\overline{A_1 A_3}$ and $\overline{A_2 A_4}$. This SOLA compression will continue as long as the present voicing decision has not changed from 0 to 1, which here happens in step 3. At that point, the whole synthesis frame will be output, except for its last S_a samples, to which last S_a samples from the current analysis frame are appended. This can be viewed as re-initialization of the synthesis frame, now becoming $\overline{a_3 A_5}$. With it, a new SOLA compression cycle starts in step 4, etc.

It can be seen that, while maintaining speech continuity, much of frame $\overline{a_3 A_4}$ will be translated, as well as several input frames succeeding it, thanks to SOLA's slow convergence. (Recall discussion from Section 2.2.3.) These parts exactly correspond to the region which is most likely to contain a voiced onset. We shall revisit the topic of translation of voiced onsets in more detail in the following section.

It can now be concluded that after each iteration the compressor will output an "information triplet", consisting of a speech frame, SOLA k and a voicing decision corresponding to the front frame in the buffer. Since no cross-correlation is computed during the translation, $k_i = 0$ will be attributed to each translated frame. So, by denoting speech frames by their length, the triplets produced in this case are $(S_s, k_0, 0)$, $(S_s, k_1, 0)$, $(S_a + k_1, 0, 0)$ and $(S_s, k_3, 1)$. Note that the transmission of (most) k's acquired during the compression of unvoiced speech is superfluous, because (most) unvoiced frames will be expanded using the parametric method.
As will be seen in Section 4.3, the expander will have to keep the track of the synchronization parameters in order to identify the incoming frames and handle them appropriately.

4.3 Translation of voiced onsets

In this section, we comment on some impacts of translating voiced onsets, and our realization of that translation.
The principal consequence of translation is that it "disturbs" a continuous time-scale compression. We have seen that all compressed frames have the equal length of S_s samples, while the length of translated frames is variable. This could introduce difficulties in maintaining a constant bit-rate when the time-scale compression is followed by the coding. At this stage, we choose to compromise the requirement of achieving a constant bit rate, in favour of achieving a better quality.
With respect to the quality, one could also argue that preserving a segment of the speech through translation could introduce discontinuities if the connecting segments on its both sides are distorted. In our previously described realization, we make use of some properties which offer protection against such discontinuities. The first is the ability of our voicing detector to detect voiced onsets early, which implies that the translated segment will start with a part of the unvoiced speech preceding the onset. The second is SOLA's slow convergence for moderate compression rates, which ensures that the terminating part of the translated speech will include some of the voiced speech succeeding the onset. It must be pointed out here that we also conveniently make use of the fact that the duration of a voiced onset does not exceed the length of several (S_a samples long) input frames, for typical values of S_a ($\geq$ 80 samples).
Figure 4.4 shows a typical example of the above mentioned "early detection" of a voiced onset. (In fact, the voicing detector allows tuning, which can make it more "conservative" towards voiced or unvoiced speech.) It also demonstrates an imperfection of detecting a voiced onset as any transition from unvoiced-like to voiced-like speech. The first detected onset is barely worthy of translation, since it does not mark a significant transient which could not be handled by SOLA. Now, in order to compensate the compression loss caused by occasional translation of such onsets, we simply use a slightly higher compression factor. (More discriminating methods for detecting transients are conceivable, such as those exploiting auto-correlation and cepstral distance, which are described in [15].)

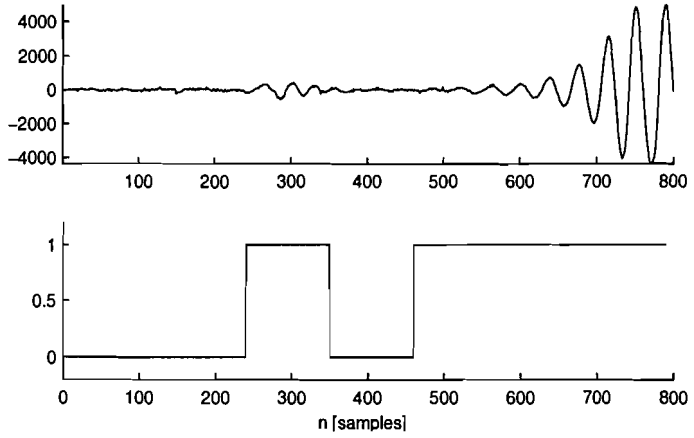

Figure 4.4: A speech signal and the corresponding voicing contour (voiced = 1). In this example, two voiced onsets are detected as transitions from unvoiced to voiced(-like) speech.

4.4 The expander

We have seen that during the compression each incoming S_a samples long frame will produce a S_s or $S_a + k_{i-1}$ ($k_i \leq S_a$) samples long frame at the output. Hence, in order to reinstate the original time-scale, the speech coming out of the expander should comprise of S_a samples long frames, or frames having different lengths but producing the same total length of $m \cdot S_a$, with m being the number of iterations. At this stage, we present a realization which is capable of only *approximating* the desired length. This is the result of a pragmatic choice, allowing us to simplify the operations and avoid introducing further algorithmical delay. Nevertheless, this approximation will prove to be adequate for simulation purposes, as will be seen in the next section.
In the following, we shall assume to have disposal over several separate buffers, all of which will be updated by simple shifting of samples. For the sake of illustration, we shall be showing the complete "information triplets" as produced by the compressor, including the k's acquired during compression of unvoiced sounds, most of which are actually obsolete.
This is also illustrated in figure 4.5, where an initial state is shown. The buffer for incoming speech is represented by segment $\overline{A_0 M}$, which is $4S_a$ samples long. For the sake of illustration, it is assumed the expansion directly follows the compression described in figure 4.3. Two additional buffers $\overline{\xi\lambda}$ and Y will serve, respectively, to provide the input information for the LPC analysis and to facilitate expansion of voiced parts. Another two buffers are deployed to hold the synchronization parameters, namely the voicing decisions and k's. The flow of these parameters will be used as a criterion to identify the incoming speech frames and handle them appropriately. From now on, we shall refer to positions 0, 1 and 2 as **past**, **present** and **future**, respectively.
During the expansion, some typical actions will be performed on the "present" frame, invoked by particular states of the buffers containing the synchronization parameters. In the following, this is thoroughly clarified through examples.

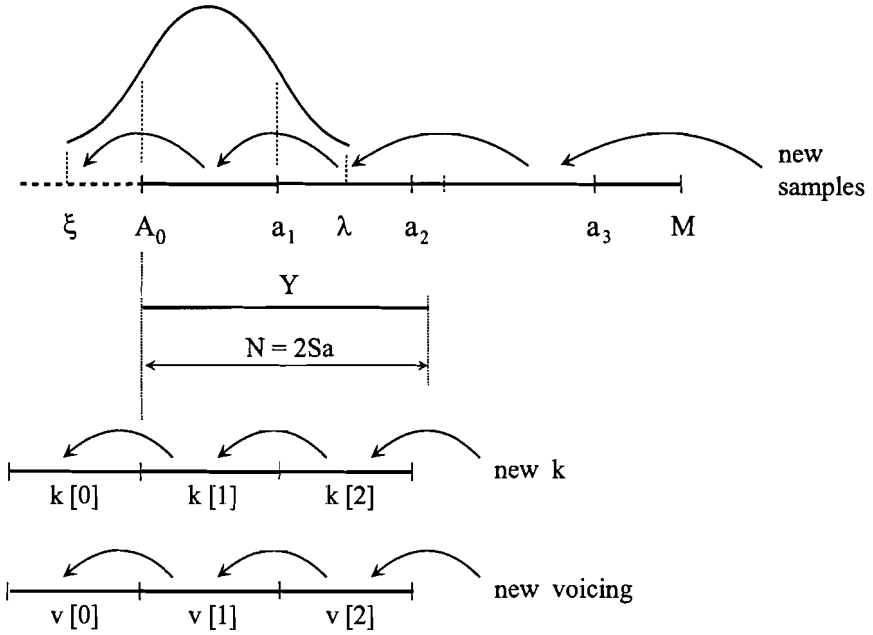

Figure 4.5: Illustration of different buffers during the initial stage of expansion, which follows directly the compression illustrated in figure 4.3.

i. Unvoiced expansion

The parametric expansion method described in Section 3.3 is exclusively deployed in the situation where all three frames of interest are unvoiced, as shown in figure 4.6. (This implies, $d(\overline{a_0a_1}) = S_s$, $d(\overline{a_1a_2}) = S_s$ and $d(\overline{a_2a_3}) = S_a$ or $S_a + k[1]$.) Later, an additional requirement will also be introduced and explained, stating that these frames should not form an immediate continuation of a voiced offset (transition from voiced to unvoiced speech).

Hence, the present frame $\overline{a_1a_2}$ is extended to the length of S_a samples and output, which is followed by left-shifting the buffer contents by S_s samples, making $\overline{a_2a_3}$ new present frame and updating the contents of the "LPC buffer" $\overline{\xi\lambda}$. (Typically, $d(\overline{\xi\lambda}) \approx 2S_s$.).

ii. Voiced expansion

A possible voicing state invoking this expansion method is illustrated in figure 4.7. Let us first assume that the compressed signal *starts* with $\overline{a_1a_2}$, i.e. that $\overline{a_0a_1}$, $v[0]$ and $k[0]$ are empty. Then, Y and X exactly represent the first two frames of the time-scale "reconstruction" process described in Section 2.2.4. It was said there that $2S_a$ samples long frames $\hat{x}_i$, with in this case $Y = \hat{x}_0$, $X = \hat{x}_1$, need to be excised from the compressed signal at position $iS_s + k_i$ and "put back" at the original positions iS_a, while cross-fading the overlapping samples. Apparently, first S_a samples of Y are not used during the overlap-add, so they are output. This can be viewed as expansion of S_s samples long frame $\overline{a_1a_2}$, which is then replaced by its successor $\overline{a_2a_3}$ by the usual left-shifting. It is now clear that all consecutive S_s samples long frames can be expanded in the analogue way, i.e. by outputting first S_a samples from buffer Y, where the rest of this buffer is continuously updated through overlap-add with X obtained for a certain present k, i.e. $k[1]$. Explicitely, X will contain $2S_a$ samples from the input buffer, starting with $S_s + k[1]$-th sample.

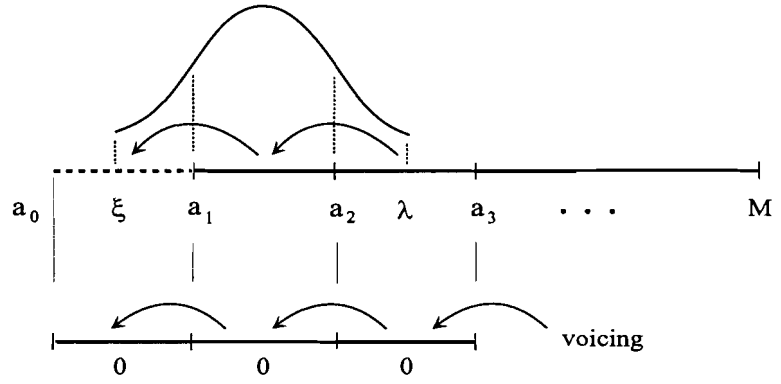

Figure 4.6: A present unvoiced frame is expanded using the parametric method only if both past- and future frame are unvoiced as well.

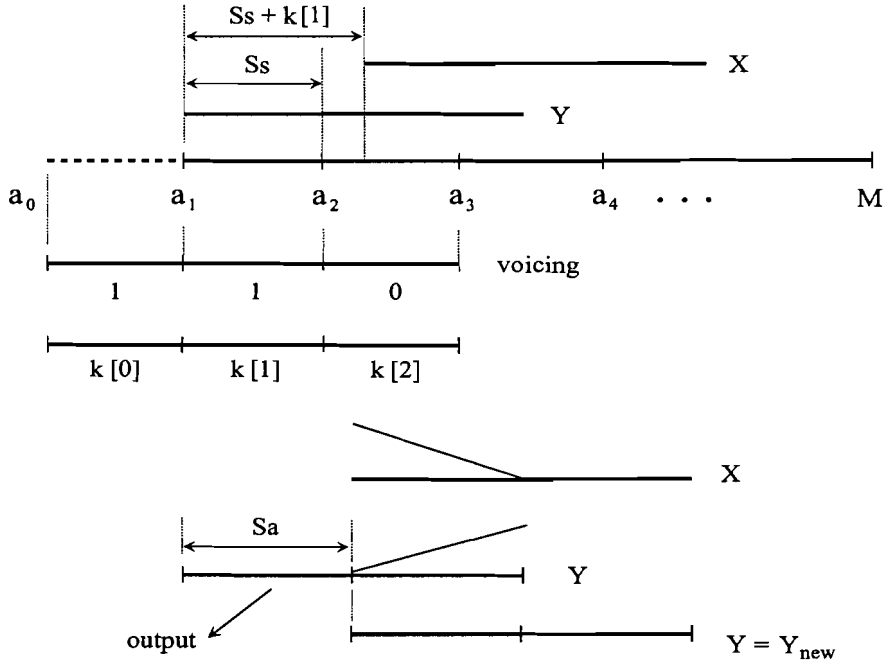

Figure 4.7: During voiced expansion, the present S_s samples long frame is expanded by outputting front S_a samples from $2S_a$ samples long buffer Y. This buffer is continuously updated by overlap-add with X, representing $2S_a$ samples from the input speech buffer, starting at the position determined by the present k.

iii. Translation

We shall use this term to refer to all situations where the present frame, or a part of it, is output as is or skipped, i.e. shifted but not output. We shall motivate and clarify these action using again figure 4.7. There, it can be seen that at the time the unvoiced frame $\overline{a_2a_3}$ has become the present frame, its front $S_a - S_s$ samples will already have been output during the previous iteration. Namely, these samples are included in the front S_a samples of Y, which have been output during the expansion of $\overline{a_2a_3}$. Consequently, expanding a present unvoiced frame that follows a past voiced frame using the parametric method would disturb speech continuity. Therefore, we first decide to maintain voiced expansion during such voiced *offsets*. In other words, the voiced expansion is prolonged to the first unvoiced frame succeeding a voiced frame. (This will not activate the "tonality problem", which is primarily caused when "repetition" of SOLA expansion extends over a relatively longer unvoiced segment.)

However, it is clear that the above outlined problem will now only be postponed and will re-appear with the future frame $\overline{a_3a_4}$. Keeping in mind the way voicing expansion is performed, i.e. the way Y is updated, a total of k_i ($0 \leq k \leq S_a$) samples may have already been output (modified by cross-fade) before they have arrived at the front of the buffer.

At this stage, we resort to a simple remedy for this problem. Firstly, each present k_i samples that have been used in the past will be skipped. This now implies a deviation from the principle exploited so far, where for each incoming S_s samples S_a samples are output. In order to compensate "the shortage" of samples", we shall use the "surplus" of samples contained in the translated $S_a + k_j$ samples long frames produced by the compressor, If such a frame does not directly follow a voiced offset (if a voiced onset does not appear shortly after a voiced offset) then none of its samples will have been used in the previous iterations, and it can be output as a whole. Hence, the "shortage" of k_i samples following a voiced offset will be counterbalanced by a "surplus" of at most k_j samples preceeding the next voiced onset.

Since both k_j and k_i are obtained during compression of unvoiced speech, therefore having a random-like character, their counterbalance will not be exact for a particular j and i. As a consequence, a slight mismatch between the duration of the original and the corresponding companded *unvoiced* sounds will generally result, which is expected to be not perceivable. At the same time, speech continuity is assured. (One could argue that a rigorous maintenance of the continuity of unvoiced, thus noise-like speech does not need to be strongly requested.)

It should be noted that the mismatch problem could easily be tackled even without introducing additional delay and processing, by choosing a same k for all unvoiced frames during the compression. Possible quality degradation due to this action is expected to remain bounded, since waveform similarity, based on which k is computed, is not an essential similarity measure for unvoiced speech.

It must be pointed out that all the buffers need to be consistently updated, in order to ensure speech continuity when switching between different actions. For the purpose of this switching and identification of incoming frames, a decision mechanism has been established, based on inspecting the states of voicing- and "k-buffer". Without going into all the details, it can be summarized through the table given below, where the previously described actions are abbreviated. To signal "re-usage" of samples, i.e. occurrence of a

voiced offset in the past, an additional predicate named "offset" is introduced. It can be defined by looking one step further into the past of the voicing buffer, as true if $v[0] = 1 \vee v[-1] = 1$ and false in all other cases ($\vee$ denotes logical "or"). Note that through suitable manipulation, no explicit memory location for $v[-1]$ is needed.

Table 4.1 Selecting actions of the expander

$v[0]$	$v[1]$	$v[2]$	offset	$k[0] > S_s$	ACTION
0	0	0	0	-	UV
0	0	0	1	0	UV
0	0	0	1	1	T
0	0	1	-	-	T
0	1	1	-	-	V
1	0	0	-	-	V
1	0	1	-	-	T
1	1	0	-	-	V
1	1	1	-	-	V

In this text, we shall refrain from presenting the exact mathematical and algorithmical formulations of the above explained concepts.

4.5 Performance evaluation

The system performance has been evaluated through waveform inspection and informal listening tests, carried out in the presence of three experienced listeners. The simulations have been performed on speech recordings of both male and female speakers (in several languages), typically ranging in duration from few to 15-20 seconds.
By comparing the original and the corresponding compressed waveforms, proper detection and translation of voiced onsets have been confirmed. Due to translation, a compression factor of 27-28 % was typically needed ($S_a = 100$, $S_s = 72$) in order to achieve overall compression of 25 %.
The mismatch between the time scale of the original and time scale of the reconstructed unvoiced sounds due to inadequate expansion, has appeared to be only minor and inaudible. Based on this observation, trustworthy listening tests could be conducted, which, in conformity with the motivations from Chapter 1, were organized to evaluate:

A. time-scale companding in absence of codec, through:

   - comparison with the original speech
   - comparison with TDHS, all-SOLA, PICOLA

B. time-scale companding incorporating codec, through

   - comparison with the original speech
   - comparison with some standard speech codecs at comparable (moderate to low) bit-rates

Revisiting the comparison with TDHS and PICOLA has been motivated by the desire to obtain a more complete impression of the performance of our system, without focusing on particular problems of SOLA that we have tried to fix (tonality in unvoiced speech, and smearing of voiced onsets).
As a result of the tests from the *first group*, a solid impression of a good performance of our system has been established. Quality degradation with respect to the original speech was perceivable, but tolerable. The comparisons with companding systems employing PICOLA, "all-SOLA" and TDHS generally provided results in favour of our approach. Using SOLA and translation of voiced onsets has proven capable of reducing the "reverberation artefact" in voiced speech, which is encountered in TDHS and PICOLA (less compared to TDHS). At the same time, using the parametric method for time-scale expansion of unvoiced speech has removed the tonal which is present there when only SOLA is used, at the cost of somewhat increased perception of noisiness in stressed (energy-rich) unvoiced sounds.
The tests from the *second group* have revealed a considerable quality degradation, which increases in proportion to the lowering of the bit-rate. This is in accordance with the common intuition that the system performance will suffer more if "worse" codecs (operating at lower bit-rates) are used. Unfortunately, the time-scale companding incorporating a codec operating at a higher bit rate also performed worse than a dedicated "stand-alone" codec operating at a lower bit rate.
The results of these tests are summarized in the table below. First, several speech codecs that have been tried are listed. These are standardized codecs dedicated to the given bit-rates. As a reference, another standard codec has been used, namely AMR (adaptive multi-rate), allowing several bit-rates. The codecs and the bit-rates have been chosen so to ensure a fair comparison for a typical tuning of our companding system. Specifically, 25 % compression rate has been used, while the cost of the transmission of synchronization parameters is estimated to 500 bit/s. The experiments have primarily targeted moderate to low bit-rates. In the table, they are expressed in kbit/s. For example, $9.15 + 0.5$ kbit/s is computed as $0.75 \cdot 12.2 + 0.5$ kbit/s.

A. EFR at 12.2 kbit/s

B. G729 at 8 kbit/s

C. G723 at 6.3 kbit/

Table 4.2 Evaluation of TSM-based speech coding systems

SYSTEM (br)	REFERENCE (br)	COMMENT:
A (9.15+0.5)	AMR (10.2)	Difference perceivable, but not annoying
B (6+0.5)	AMR (6.70)	Difference perceivable, slightly annoying
C (4.73+0.5)	AMR (5.15)	Difference perceivable, and annoying

Note that the above mentioned 0.5 kbit/s has been computed utilizing relations 2.20 given in Section 2.2.4 and a common observation that nearly one third of speech consists of unvoiced sounds (for which k's do not need to be transmitted). So, $0.7 \cdot 560$ bit/s are attributed to transmission of k and additional 80 bit/s to transmission of the voicing information (one bit per frame), the sum of which is "ceiled" to 500 bit/s.

The findings presented in Table 4.2 can now be explained through some reasonable assumptions and speculations. Firstly, it seems logical that the codecs operating at lower bitrates will introduce more degradation to the waveform (shape) of the compressed speech. Such distorted speech is then expected to be a less adequate input for the expander, since it is "tuned" directly to the compressor (through transmission of k's) and does not explicitly take into account the influence of the codec. Further, one can imagine that the expansion "repeats" the distortions caused by the coding. It is even possible that in this way distortions of the waveform which are inaudible in the decoded compressed speech become audible after expansion.

Chapter 5

Concluding remarks

Presently, several techniques and approaches exist that can successfully (e.g. giving good quality) be employed for compressing or expanding the time-scale of speech. When used for coding purposes, where the time-scale compression is followed by time-scale expansion (time-scale companding), their performance degrades considerably. The best performance is generally obtained from time-domain methods, among which SOLA is widely used.

Our experiments showed that most of the problems accompanying time-scale companding occur during the unvoiced segments and voiced onsets that are present in a speech signal. In the output signal, the unvoiced sounds take on a tonal character, while less gradual and smooth voiced onsets are often smeared, especially when larger scale factors are used.

The tonality in unvoiced sounds is introduced by the "repetition" mechanism which is inherently present in all time-domain algorithms. To overcome this problem, we propose using separate methods for expanding voiced and unvoiced speech. We have developed an improved method for expansion of unvoiced speech, which is based on inserting an appropriately shaped noise sequence into the compressed unvoiced sequences.

To avoid smearing of voiced onsets, we exclude them from TSM and translate them instead. (Translation of speech transients was already reported in [15].)

Combining these concepts with SOLA, we have realized time-scale companding system which outperforms the traditional realizations that use a similar algorithm for both compression and expansion. For evaluation purposes, we have used 25 % compression - 33 % expansion. The companded speech has been judged to have good subjective quality, exhibiting a tolerable degradation compared to the original speech.

As expected, introducing a speech codec between the TSM stages causes quality degradation, being more noticeable in proportion to the lowering of the bit-rate of the codec. When a particular codec and TSM are combined to produce a certain bit-rate, the resulting system performs worse than dedicated speech coders operating at a comparable bit-rate. At lower bit-rates, quality degradation is unacceptable. However, TSM can be beneficial in providing graceful degradation at higher bit-rates.

Inspired by the results of this study, we suggest that future research for improvements should distinguish two cases. With high to moderate bit-rate coders, further improvements through the conventional approach, i.e. where the TSM stages can be designed independent from the coder, seem feasible. With low bit-rates, on the other hand, an alternative approach is indispensable, where the influence of the coder would be accounted for during the design of the TSM stages.
In our system in particular, which complies with the former of the above mentioned concepts, several improvements are conceivable. So, refinements of the proposed expansion method for unvoiced speech through deploying alternative ways of noise insertion and gain computation could be attempted. Furthermore, it would be of great practical value to eliminate the need for transmission of SOLA synchronization parameters k, what would enable more flexible expansion, less dependent on the reliability of this transmission. To accomplish this, k's should be derived from the received signal, which would also be advantageous in terms of taking directly into account the influence of the codec.
An alternative approach to TSM is suggested in figure 5.1. It exploits analysis-by-synthesis, where the difference between the original and the output speech is used as a measure based on which all the processing stages can be adjusted. At this point, however, it is not clear which error measure should be used (SNR is not expected to be adequate), nor which companding and coding techniques.

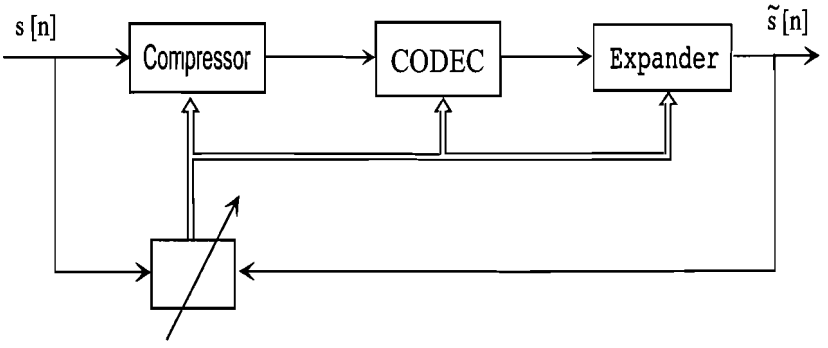

Figure 5.1: A general TSM-based coding scheme employing analysis-by-synthesis. All the processing stages are adjusted based on comparison between the original and output speech

Bibliography

[1] D. Malah, "Time-Domain Algorithms for Harmonic Band-Width Reduction and Time Scaling of Speech Signals", IEEE Transactions on ASSP, Vol. 27, No. 2, p.123-33, 1979

[2] M. R. Portnoff, "Time-Scale Modification of Speech Based on Short-Time Fourier Analysis", IEEE Transactions on ASSP, Vol. 29, No. 3, p.374-90, 1981

[3] R. V. Cox, R. E. Crochiere, J. D. Johnston, "Real-Time Implementation of Time-Domain Harmonic Scaling of Speech for Rate Modification and Coding", IEEE Transactions on ASSP, Vol. 31, No. 1, p.258-71, 1983

[4] S. Roucos, A. Wilgus, "High Quality Time-Scale Modification for Speech", Proc. of the IEEE ICASSP, Tampa, FL, USA, 26-29 March 1985, Vol. 2, p.493-6

[5] J. Makhoul, A. El-Jaroudi, "Time-Scale Modification in Medium to Low Rate Speech Coding", Proc. of ICASSP, Tokyo, Japan, 7-11 April 1986, Vol. 3, p.1705-8

[6] P. E. Papamichalis, "Practical Approaches to Speech Coding", Prentice-Hall,Inc., Engelwood Cliffs, New Jersey, 1987

[7] J. L. Wayman, D. L. Wilson, "Some Improvements on the Method of Time-Scale-Modification for Use in Real-Time Speech Compression and Noise Filtering", IEEE Transactions on ASSP, Vol. 36, No. 1, p.139-40, 1988

[8] F. Amano, K. Iseda, K. Okazaki, S. Unagami, "An 8 kbit/s TC-MQ (Time-domain Compression ADPCM-MQ) Speech Codec", Proc. of the IEEE ICASSP, New York, USA, 11-14 April 1988, p.259-62 vol.1.

[9] A. V. Oppenheim, R.W. Schafer, "Discrete-Time Signal Processing", Prentice-Hall, Inc., Engelwood Cliffs, New Jersey, 1989

[10] E. Hardam, "High Quality Time-Scale Modification of Speech Signals Using Fast Synchronized-Overlap-Add Algorithms", Proc. of the IEEE ICASSP, Albuquerque, NM, USA, 3-6 April 1990, Vol. 1, p.409-12

[11] W. Verhelst, M. Roelands, "An Overlap-Add Technique Based on Waveform Similarity (WSOLA) for High Quality Time-Scale Modification of Speech", Proc. of the IEEE ICASSP, Minneapolis, MN, USA, 27-30 April 1993, Vol. 2, p.554-7

[12] E. Moulines, J. Laroche, "Non-Parametric Techniques for Pitch-Scale and Time-Scale Modification of Speech", In: Speech Communications (Netherlands), Vol. 16, No. 2, p.175-205, 1995

[13] R. Veldhuis, He. Haiyan, "Time-Scale and Pitch-Scale Modifications of Speech Signals and Resynthesis from the Discrete Short-Time Fourier Transform", In: Speech Communications, Vol. 18, No. 3, p.257-79, 1996

[14] S. Yim, B. I. Pawate, "Computationally efficient algorithm for Time-Scale Modification (GLS-TSM)", Proc. of the IEEE ICASSP, Atlanta, GA, USA, 7-10 May 1996, Vol. 2, p.1009-12

[15] M. Sungjoo-Lee, Hee-Dong-Kim, Hyung-Soon-Kim, "Variable Time-Scale Modification of Speech Using Transient Information", Proc. of the IEEE ICASSP, Munich, Germany, 21-24 April 1997, p.1319-22

[16] R. Taori, R. Sluijter, and A. Gerrits, "A Versatile Low Bit Rate Speech Coding System", Philips Research Technical Note 157/97, April 1997.

[17] L. Chunyan, V. Cuperman, A. Gersho, "Robust Closed-Loop Pitch Estimation for Harmonic Coders by Time-Scale Modification", Proc. of the IEEE ICASSP, Phoenix, AZ, USA, 15-19 March 1999, Vol. 1, p.257-60