6

Time-segment processing

P. Dutilleux, G. De Poli, A. von dem Knesebeck and U. Zölzer

6.1 Introduction

In this chapter we discuss several time-domain algorithms which are a combination of smaller processing blocks like amplitude/phase modulators, filters and delay lines. These effects mainly influence the pitch and the time duration of the audio signal. We will first introduce some basic effects like variable speed replay and pitch-controlled resampling. They are all based on delay-line modulation and amplitude modulation. Then we will discuss two approaches for time stretching (time scaling) of audio signals. They are based on an analysis stage, where the input signal is divided into segments (blocks) of fixed or variable length, and a synthesis stage where the blocks of the analysis stage are recombined by an overlap and add procedure. These time-stretching techniques perform time scaling without modifying the pitch of the signal. The fourth section focuses on pitch shifting, and introduces three techniques: block processing based on time stretching and resampling, delay line modulation and pitch-synchronous block processing. Block processing based on delay line modulation performs pitch shifting by scaling the spectral envelope of each block. Pitch-synchronous block processing performs pitch shifting by resampling the spectral envelope of each block and thus preserving the spectral envelope. The last section on time shuffling and granulation presents a more creative use of time-segment processing. Short segments of the input signal are freely assembled and placed along time in the output signal. In this case the input sound can be much less recognizable in the output. The wide choice of strategies for segment organization implies a sound composition attitude from the user.

DAFX: Digital Audio Effects, Second Edition. Edited by Udo Zölzer.
 Published 2011 by John Wiley & Sons, Ltd. ISBN: 978-0-470-66599-2

6.2 Variable speed replay

Introduction

Analog audio tape recorders allow replay with a wide range of tape speeds. Particularly in fast forward/backward transfer mode, a monitoring of the audio signal is possible which can be used to locate a sound. During faster playback the pitch of the sound is raised and during slower playback the pitch is lowered. With this technique the duration of the sound is lengthened if the tape is slowed down, and shortened if the tape speed is increased. Figure 6.1 illustrates a sound segment which is lengthened and shortened, and the corresponding spectra.

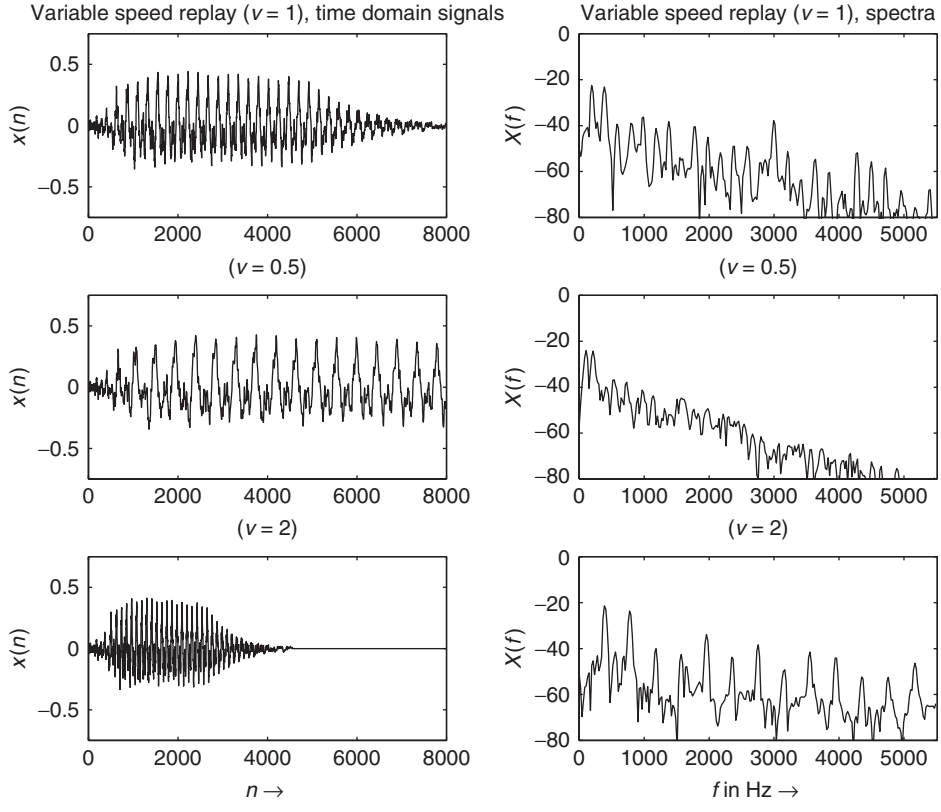

Figure 6.1 Pitch shifting: Variable speed replay leads to time compression/expansion and compression and expansion of the spectral envelope.

Signal processing

The phrase "variable speed replay" denotes that what has happened initially during the time period $nT_{s,in}$ is now happening during

$$nT_{s,replay} = nT_{s,in}/v \tag{6.1}$$

at the relative speed v, where $T_{s,in}$ and $T_{s,replay}$ are the initial and replay sampling periods. Time expansion corresponds to $v < 1$. A straightforward method of implementing the variable speed replay is hence to modify the sampling frequency while playing back the sound according to

$$f_{s,replay} = f_{s,in} \cdot v \tag{6.2}$$

where $f_{s,in}$ and $f_{s,replay}$ are the initial and replay sampling frequencies. One should distinguish whether the output should be digital or may be analog. If the output is analog, then a very effective method is to modify the sampling frequency of the output DAC. The spectrum of the signal is scaled by v and the analog reconstruction filter should be tuned in order to remove the spectral images after the conversion [Gas87, Mas98].

If a digital output is required, then a sampling frequency conversion has to be performed between the desired replay frequency $f_{s,replay}$ and the output sampling frequency $f_{s,out}$, which is usually equal to $f_{s,in}$.

If $v < 1$ (time expansion) then $f_{s,in} > f_{s,replay} < f_{s,out}$ and more output samples are needed than available from the input signal. The output signal is an interpolated (over-sampled) version by a factor $1/v$ of the input signal. If $v > 1$ (time compression) then $f_{s,in} < f_{s,replay} > f_{s,out}$ and less output samples than available in the input signal are necessary. The input signal is decimated by a factor v. Before decimation, the bandwidth of the input signal has to be reduced to $f_{s,replay}/2$ by a digital lowpass filter [McN84]. The quality of the sampling rate conversion depends very much on the interpolation filter used. A very popular method is the linear interpolation between two adjacent samples. A review of interpolation methods can be found in [Mas98, CR83].

A discrete-time implementation can be achieved by increasing/decreasing the transfer rate of a recorded digital audio signal to the DA converter, thus changing the output sampling frequency compared to the recording sampling frequency. If the output signal has to be in digital format again, we have to resample the varispeed analog signal with the corresponding sampling frequency. A discrete-time implementation without a DA conversion and new AD conversion was proposed in [McN84] and is shown in Figure 6.2. It makes use of multirate signal-processing techniques and performs an approximation of the DA/AD conversion approach. A further signal-processing algorithm to achieve the acoustical result of a variable speed replay is delay line modulation with a constant pitch change, which will be discussed in Section 6.4.3.

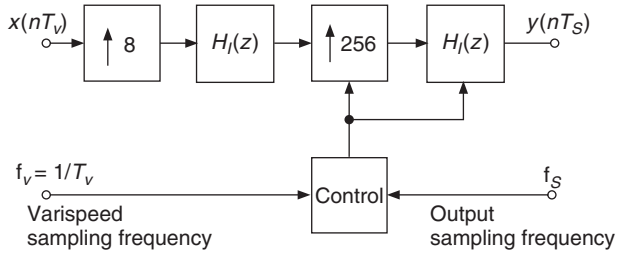

Figure 6.2 Variable speed replay scheme.

Musical applications and control

As well as for analog tape-based audio editing, variable-speed replay is very popular in digital audio editing systems. See [m-Wis94c, ID 2.9 and 2.10] for a straightforward demonstration of the effect on a voice signal.

The effect of tape-speed transposition has been used by Les Paul in the piece called "Whispering" in 1951 [Lee72]. This method is very often used in electro-acoustic music when the pitch of concrete sounds cannot be controlled at the time of recording. P. Schaeffer designed the *Phonogène chromatique* to transpose a sound to any one of the 12 degrees of the equal tempered scale. The device was based on a tape recorder with 12 capstans and pinch rollers. The operation of the pinch rollers could be controlled by a piano-like keyboard. An additional gear extended the range of operation to two octaves [Mol60, p. 71]; [Roa96, p. 119]; [Ges00]. Jacques Poullin developed another version, the *Phonogène à coulisse*, which allowed continuous speed modifications. A pair of cones, with a friction wheel in between, constitutes a variable-ratio mechanical link between the motor and the capstan of the tape player. The position of the friction wheel, and hence the replay speed, is controlled by a mechanical lever. Stockhausen, in "Hymnen," transposed orchestral sounds to give them an overwhelming and apocalyptic character [Chi82, p. 53].

In computer music too, variable-speed replay provides an effective transposition scheme. J.-C. Risset says: by "mixing a sound with transpositions of itself with a minute frequency difference (say, a twentieth of a Hertz), one can turn steady periodic tones into a pattern where the harmonics of the tone wax and wave at different rates, proportional to the rank of the harmonic" [Ris98, p. 255]; [m-INA3, Sud]. In "The Gates of H.," Ludger Brümmer exploits the fact that variable speed replay modifies both the pitch and the duration of a sample [m-Bru93, 14'40''–17'25'']. Seven copies of the same phrase, played simultaneously at speeds 7.56, 4.49, 2.24, 1.41, 0.94, 0.67, 0.42, 0.31 are overlapped. The resulting sound begins with a complex structure and an extended spectrum. As the piece continues, the faster copies vanish and the slower versions emerge one after the other. The sound structure simplifies and it evolves towards the very low registers.

The character of the transposed sounds is modified because all the features of the spectrum are simultaneously scaled. The formants are scaled up, leading to a "Mickey Mouse effect," or down, as if the sounds were produced by oversized objects. The time structure is modified as well. The transients are spread or contracted. A vibrato in the initial sound will lose its character and will appear as a slower or faster modulation. The sounds can also be played at negative speeds. A speed -1 yields a sound with the same average spectrum, although sounding very different. Think about speech or percussive sounds played backwards. Other transposition schemes that are free from these drawbacks are achieved by more sophisticated methods described in further sections of this book.

A particular application was desired by the composer Kiyoshi Furukawa. He wanted a sampler for which the speed would be controlled by the amplitude of an acoustical instrument. A sound is stored in a sampler and is played as a loop. In the meantime, the RMS amplitude of an incoming controlling signal is computed and time averaged with independent attack and decay time constants. This amplitude is converted to decibels and scaled before driving the speed of the sampler. The parameters have to be tuned in such a way that the speed remains within a valid range and the speed variations are intimately related to the loudness and the dynamics of the instrument (see Figure 6.3).

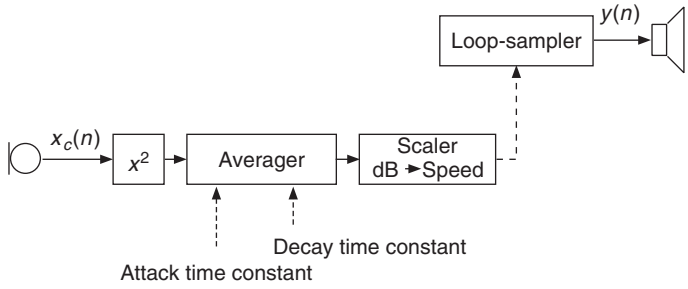

Figure 6.3 A loop-sampler controlled by an acoustical signal.

This effect is controlled by a clarinet in "Swim, swan" and by a viola in "Den Ungeborenen Göttern" [m-Fur93, m-Fur97]. The pitch of the acoustical instrument selects words out of a predefined set, whereas the loudness controls the replay speed of these words.

6.3 Time stretching

Introduction

In order to understand the issue of time stretching, let us take the example of a signal whose duration does not fit the time slot that is allocated to its application. Think about a speaker that has already recorded 33 seconds of speech, but whose contribution to a commercial may not be longer than 30 seconds. If he does not want to record his text again, the sound engineer may artificially contract his speech by 10%. With the term "time stretching" we mean the contraction or expansion of the duration of an audio signal. We have studied in 6.2 a method that alters the duration of a sound, the variable speed replay, but it has the drawback of simultaneously transposing the sound. The *Harmonizer* could be used to transpose the sound in the opposite direction and the combination of both methods leads to a time-stretching algorithm.

The main task of time-stretching algorithms is to shorten or lengthen a sound file of M samples to a new particular length $M' = \alpha \cdot M$, where α is the scaling factor. For performing time stretching algorithms the sound file has to be available in a stored form on a storage medium like a sampler, DAT or a hard disc. Time stretching of a sequence of audio samples is demonstrated in Figure 6.4.

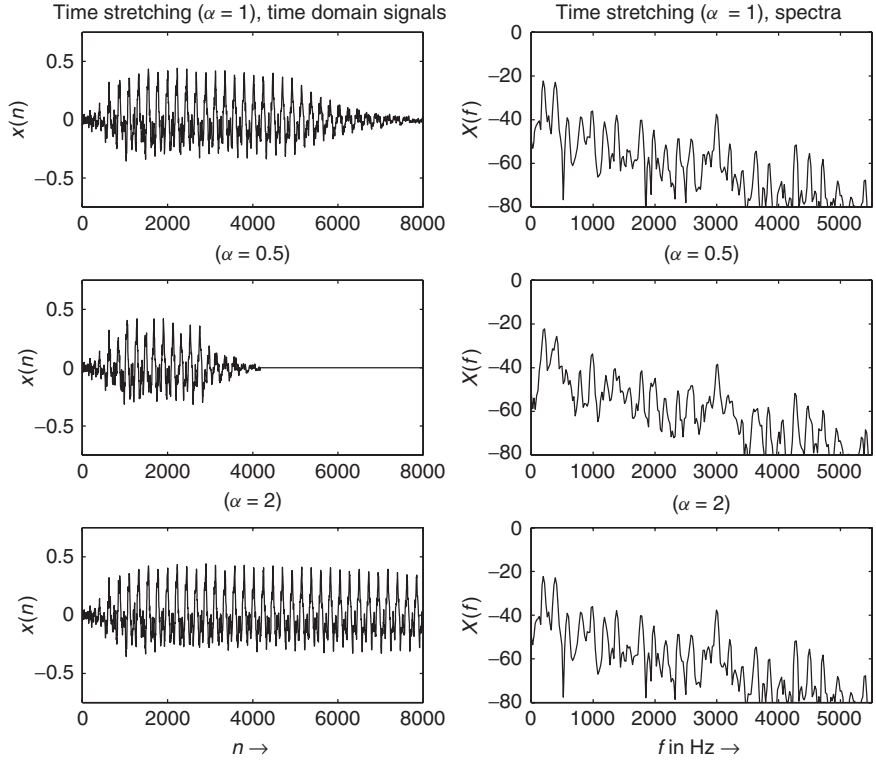

Figure 6.4 Time stretching with scaling factor $\alpha = 0.5, 2$.

The original signal is shown in the upper plot. The middle plot shows a sequence which is shortened by a scaling factor $\alpha = 0.5$ and the lower plot shows stretching by a scaling factor $\alpha = 2$.

Signal processing

The intended time scaling does not correspond to the mathematical time scaling as realized by vary-ispeed. We rather require a scaling of the perceived timing attributes, such as speaking rate, without affecting the perceived frequency attributes, such as pitch. We could say that we want the time-scaled version of an acoustic signal to be perceived as the same sequence of acoustic events as the original signal being reproduced according to a scaled time pattern. The time-stretching algorithms should not affect the pitch or the frequency contents of the processed signals. This is demonstrated by the corresponding spectra (first 2000 samples) of the discrete-time signals in Figure 6.4. For comparison, the traditional technique for time stretching based on variable speed replay introduces a pitch shift (see Section 6.2 and Figure 6.1). The basic idea of time stretching by time-segment processing is to divide the input sound into segments. Then if the sound is to be lengthened, some segments are repeated, while if the sound is to be shortened, some segments are discarded. A possible problem is amplitude and phase discontinuity at the boundaries of the segments. Amplitude discontinuities are avoided by partially overlapping the blocks, while phase discontinuities are avoided by a proper time alignment of the blocks. Two different strategies will be presented in Sections 6.3.2 and 6.3.3.

Applications

Special machines such as the *Phonogène universel* of Pierre Schaeffer or the Tempophon used by Herbert Eimerts allowed alteration of the time duration as well as the pitch of sounds. The *Phonogène* found many applications in *musique concrète* as a "time regulator." In his composition "Epitaph für Aikichi Kuboyama," Herbert Eimerts uses the Tempophon in order to iterate spoken word fragments. The device allowed the scanning of syllables, vowels and plosives and could make them shorter, longer or iterate them at will [Hal95, p. 13]; [m-Eim62].

As mentioned in the Introduction, the stretching of signals can be used to match their duration to an assigned time slot. In Techno music, different pieces of music are played one after the other as a continuous stream. This stream is supposed to have only very smooth tempo or *bpm* (beat per minute) transitions, although the musical excerpts usually do not have the same tempo. In order to adjust the tempi to each other, the disc jockey modifies the replay speeds at the transition from one excerpt to the other. This method leads to temporary pitch modifications which could be objectionable. The use of time-stretching methods could eliminate this problem.

After a brief presentation of the technology of the Phonogène, the following sections discuss two signal-processing techniques which perform time stretching without pitch modifications.

6.3.1 Historical methods – Phonogène

Fairbanks, Everitt and Jaeger report in 1954 on a modified tape recorder for time or frequency compression-expansion of speech [Lee72, Lar98]. Springer develops a similar machine [Spr55, Spr59] and Pierre Schaeffer praises a machine called *Phonogène universel* that was designed as a combination of the aforementioned *Phonogène chromatique* and *Phonogène à coulisse*, with the rotating head drum of Springer [Mol60, p. 71–76]; [Sch73, p. 47–48]; [m-Sch98, CD2, ID. 50–52]; [Pou54, PS57, Ges00].

The modified tape recorder has several playback heads mounted on a rotating head drum. The absolute speed of the tape at the capstan determines the duration of the sound, whereas the relative speed of the heads to that of the tape determines the amount of transposition. By electrical summation of the outputs of the different heads, a continous sound is delivered (Figure 6.5). Moles

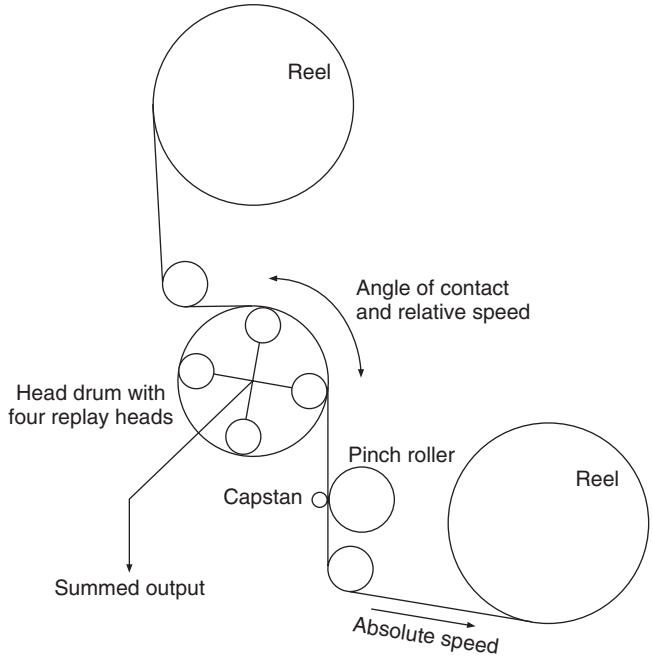

Figure 6.5 Tape-based time compression – expansion system, according to [Mol60].

reports a typical operating range of +10% to −40% [Mol60, p. 76]. The Springer machine was also known as *Laufzeitregler* or *Zeitdehner* [Car92, p. 479–480]; [End97].

6.3.2 Synchronous overlap and add (SOLA)

A simple algorithm for time stretching based on correlation techniques is proposed in [RW85, MEJ86]. Figure 6.6 illustrates the algorithm. The input signal $x(n)$ is divided into overlapping blocks of a fixed length N (see Figure 6.6a). S_a is the analysis step size, which determines the progression of the successive blocks along the input signal. In a second step these overlapping blocks are shifted according to the time-scaling factor α leading to the synthesis step size $S_s = \alpha \cdot S_a$ (see Figure 6.6b). Then the similarities in an interval of length L within the area of the overlap are searched for a discrete-time lag k_m of maximum similarity. The shift of two succeeding blocks is corrected by this time lag to synchronize the blocks and reduce artifacts (see Figure 6.6c). Finally at this point of maximum similarity the overlapping blocks are weighted by a fade-in and fade-out function and summed sample by sample (see Figure 6.6d).

Algorithm description:

(1) Segmentation of the input signal into overlapping blocks of length N with time shift of S_a samples.

(2) Repositioning of blocks with time shift $S_s = \alpha \cdot S_a$ using scaling factor α.

(3) Computation of the cross-correlation

$$r_{x_{L1}x_{L2}}(k) = \begin{cases} \frac{1}{L}\sum_{n=0}^{L-k-1} x_{L1}(n) \cdot x_{L2}(n+k) & , \quad 0 \le k \le L-1 \\ r_{x_{L2}x_{L1}}(-k) & , \quad -L+1 \le k < 0 \end{cases} \tag{6.3}$$

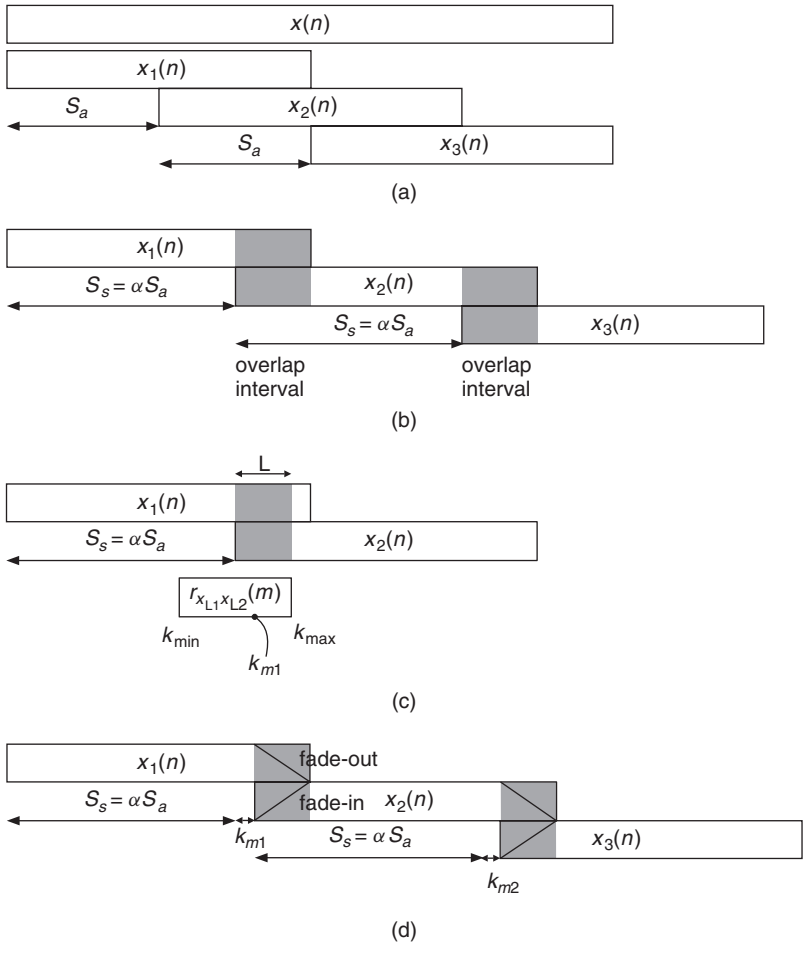

Figure 6.6 (a) Segmentation of the input signal, (b) repositioning of the blocks, (c) computation of the cross-correlation and synchronization of the blocks, (d) fading and overlap-add.

between $x_{L1}(n)$ and $x_{L2}(n)$, which are the segments of $x_1(n)$ and $x_2(n)$ in the overlap interval of length L.

(4) Extracting the discrete-time lag k_m where the cross-correlation $r_{x_{L1}x_{L2}}(k_m) = r_{\max}$ has its maximum value.

(5) Using this discrete-time lag k_m, adjust the time shift to synchronize blocks $x_1(n)$ and $x_2(n)$ for maximum similarity.

(6) *Fade-out* $x_1(n)$ and *fade-in* $x_2(n)$. Overlap-add of $x_1(n)$ and $x_2(n)$ for new output signal.

The SOLA implementation leads to time scaling with little complexity, where the parameters S_a, N, L are independent of the pitch period of the input signal.

There are a number of similar approaches derived from the SOLA algorithm. The SOLAFS, as proposed in [HM91], realizes the time-scale modification with a fixed synthesis step size, while the analysis step size is varied to achieve maximum similarity between the output and the new

window in the overlap region. A similar approach is proposed in [VR93] in the form of a waveform similarity overlap-add algorithm (WSOLA).

The following M-file 6.1 demonstrates the implementation of the SOLA time-scaling algorithm:

M-file 6.1 (TimeScaleSOLA.m)

```
% TimeScaleSOLA.m
% Authors: U. Zölzer, G. De Poli, P. Dutilleux
% Time Scaling with Synchronized Overlap and Add
%
% Parameters:
%
% analysis hop size     Sa = 256 (default parameter)
% block length          N  = 2048 (default parameter)
% time scaling factor   0.25 <= alpha <= 2
% overlap interval      L  = 256*alpha/2

clear all,close all

[DAFx_in,Fs]    =       wavread('x1.wav');
DAFx_in         =       signal';

% Parameters:
Sa        =     256;    % Sa must be less than N
N     = 2048;
alpha = 1;      % 0.25 <= alpha <= 2
Ss        = round(Sa*alpha);
L           = 128;    % L must be chosen to be less than N-Ss

% Segmentation into blocks of length N every Sa samples
% leads to M segments
M     =    ceil(length(DAFx_in)/Sa);

DAFx_in(M*Sa+N)=0;
Overlap  =  DAFx_in(1:N);

% **** Main TimeScaleSOLA loop ****
for ni=1:M-1
  grain=DAFx_in(ni*Sa+1:N+ni*Sa);
  XCORRsegment=xcorr(grain(1:L),Overlap(1,ni*Ss:ni*Ss+(L-1)));
  [xmax(ni),km(ni)]=max(XCORRsegment);

  fadeout=1:(-1/(length(Overlap)-(ni*Ss-(L-1)+km(ni)-1))):0;
  fadein=0:(1/(length(Overlap)-(ni*Ss-(L-1)+km(ni)-1))):1;
  Tail=Overlap(1,(ni*Ss-(L-1))+ ...
           km(ni)-1:length(Overlap)).*fadeout;
  Begin=grain(1:length(fadein)).*fadein;
  Add=Tail+Begin;
  Overlap=[Overlap(1,1:ni*Ss-L+km(ni)-1) ...
           Add grain(length(fadein)+1:N)];
end;
% **** end TimeScaleSOLA loop ****
% Output in WAV file
sound(Overlap,44100);
wavwrite(Overlap,Fs,'x1_time_stretch');
```

6.3.3 Pitch-synchronous overlap and add (PSOLA)

A variation of the SOLA algorithm for time stretching is the pitch synchronous overlap and add (PSOLA) algorithm proposed by Moulines *et al*. [HMC89, MC90] especially for voice processing. It is based on the hypothesis that the input sound is characterized by a pitch, as, for example, the human voice and monophonic musical instruments.

In this case PSOLA can exploit knowledge of the pitch to correctly synchronize the time segments, avoiding pitch discontinuities. When we perform time stretching of an input sound, the time variation of the pitch period $P(t)$ should be stretched accordingly. If $\tilde{t} = \alpha t$ describes the time-scaling function or time-warping function that maps the time t of the input signal into the time $\tilde{t}$ of the output signal, the local pitch period of the output signal $\tilde{P}(\tilde{t})$ will be defined by $\tilde{P}(\tilde{t}) = \tilde{P}(\alpha t) = P(t)$. More generally, when the scaling factor α is not constant, a nonlinear time-scaling function can be defined as $\tilde{t} = \mathcal{T}(t) = \int_0^t \alpha(\tau)\mathrm{d}\tau$ and used instead of $\tilde{t} = \alpha t$.

The algorithm is composed of two phases: the first phase analyses and segments the input sound (see Figure 6.7), and the second phase synthesizes a time-stretched version by overlapping and adding time segments extracted by the analysis algorithm.

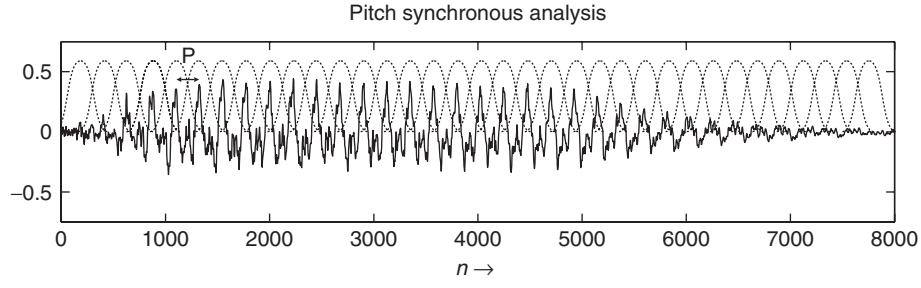

Figure 6.7 PSOLA: Pitch analysis and block windows.

Analysis algorithm (see Figure 6.8):

(1) Determination of the pitch period $P(t)$ of the input signal and of time instants (pitch marks) t_i. These pitch marks are in correspondence with the maximum amplitude or glottal pulses at a pitch-synchronous rate during the periodic part of the sound and at a constant rate during the unvoiced portions. In practice $P(t)$ is considered constant $P(t) = P(t_i) = t_{i+1} - t_i$ on the time interval (t_i, t_{i+1}).

(2) Extraction of a segment centered at every pitch mark t_i by using a Hanning window with length $L_i = 2P(t_i)$ (two pitch periods) to ensure fade-in and fade-out.

Synthesis algorithm (see Figure 6.9): for every synthesis pitch mark $\tilde{t}_k$

(1) Choice of the corresponding analysis segment i (identified by the time mark t_i) minimizing the time distance $|\alpha t_i - \tilde{t}_k|$.

(2) Overlap and add the selected segment. Notice that some input segments will be repeated for $\alpha > 1$ (time expansion) or discarded when $\alpha < 1$ (time compression).

(3) Determination of the time instant $\tilde{t}_{k+1}$ where the next synthesis segment will be centered, in order to preserve the local pitch, by the relation

$$\tilde{t}_{k+1} = \tilde{t}_k + \tilde{P}(\tilde{t}_k) = \tilde{t}_k + P(t_i).$$

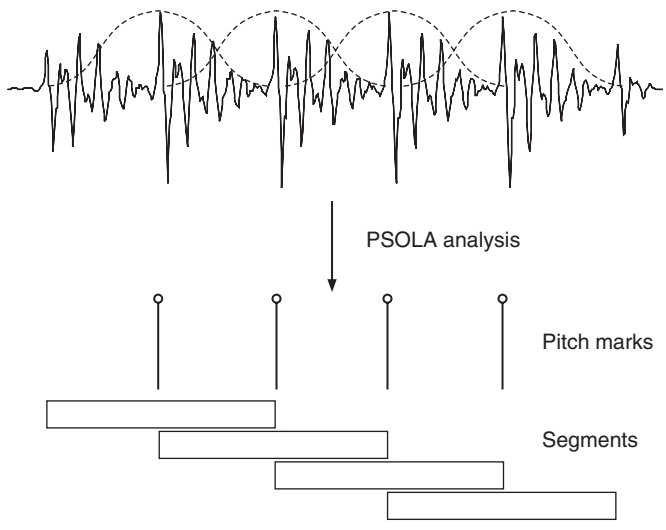

Figure 6.8 PSOLA pitch analysis.

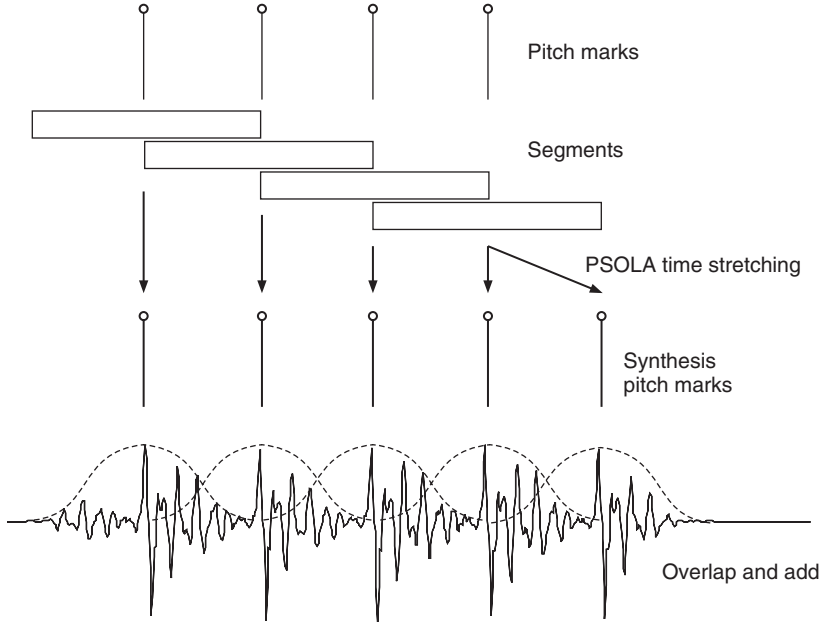

Figure 6.9 PSOLA synthesis for time stretching.

The basic PSOLA synthesis algorithm can be implemented in MATLAB® by the following M-file 6.2:

M-file 6.2 (psola.m)

```
function out=psola(in,m,alpha,beta)
% Authors: G. De Poli, U. Zölzer, P. Dutilleux
%     in    input signal
%     m     pitch marks
%     alpha time stretching factor
%     beta  pitch shifting factor

P = diff(m);   %compute pitch periods

if m(1)<=P(1), %remove first pitch mark
   m=m(2:length(m));
   P=P(2:length(P));
end

if m(length(m))+P(length(P))>length(in) %remove last pitch mark
    m=m(1:length(m)-1);
    else
    P=[P P(length(P))];
end

Lout=ceil(length(in)*alpha);
out=zeros(1,Lout); %output signal

tk = P(1)+1;       %output pitch mark

while round(tk)<Lout
  [minimum i] = min( abs(alpha*m - tk) ); %find analysis segment
  pit=P(i);
  gr = in(m(i)-pit:m(i)+pit) .* hanning(2*pit+1);
  iniGr=round(tk)-pit;
  endGr=round(tk)+pit;
  if endGr>Lout, break; end
  out(iniGr:endGr) = out(iniGr:endGr)+gr; %overlap new segment
  tk=tk+pit/beta;
end %while
```

Stretching factors typically range from $\alpha = 0.25$ to 2 for speech. Audible buzziness appears in unvoiced sound when larger values are applied, due to the regular repetition of identical input segments. In order to prevent the algorithm from introducing such an artificial short-term correlation in the synthesis signal, it is advisable to reverse the time axis of every repeated version of an unvoiced segment. With such an artifice, speech can be slowed down by a factor of four, even though some tonal effect is encountered in voiced fricatives, which combine voiced and unvoiced frequency regions and thus cannot be reversed in time.

It is possible to further exploit the analysis phase. In fact, uniformly applied time stretching can produce some artifacts on the non periodic parts of the sound. For example a plosive consonant may be repeated if the synthesis algorithm selects twice the time segment which contains the consonant. The analysis can then be extended in order to detect the presence of fast transitions

[PR99, KZZ10]. During synthesis, the time scale will not be modified at these points, thus the segments will not be repeated. This approach can be generalized for non-speech sounds where a large time-scale change during transitions (e.g., attacks) would dramatically change the timbre identity. Also in this case it is possible to limit time stretching during transitions and apply it mainly to the steady-state portion of the input sound. This technique is usually applied to digital musical instruments based on wavetable synthesis. On the other hand, the deformation of transient parts can be considered an interesting timbre transformation and can be appreciated as a musically creative audio effect.

Pitch marks

It should be noted that the determination of the pitch and of the position of pitch marks is not a trivial problem and could be difficult to implement robustly. The pitch periods can be determined with one of the pitch extraction algorithms presented in Chapter 9.2. Once the pitch periods are known, the position of the pitch marks can be determined. The sound quality of the modification results of the PSOLA algorithm essentially depends on the positioning of the pitch marks, since the pitch marks provide the centers of the segmentation windows of the PSOLA.

Obviously the pitch marks can only be positioned corresponding to the pitch period for voiced signal parts. As mentioned before, one way to handle unvoiced parts is to keep the "pitch period" at a constant value until the next voiced part. A method which determines the pitch marks of a complete voiced frame is proposed in [LJ04, MVV06]. The global maximum peak of the waveform is searched and the first pitch mark t_i is located at this position. Then the pitch marks to the right of t_i are positioned using the pitch period as a first estimate of the pitch mark locations, followed by a refinement of the location by searching the maximum within a certain region. The new pitch mark is determined as $t_{i+1} = max\left([t_i + \delta P_0, t_i + (2-\delta)\, P_0]\right)$, where P_0 is the pitch period and δ is a factor in the range of 0.5 to 0.9. This procedure is repeated until the right end of the frame is reached. Then the pitch marks to the left of t_i are positioned respectively by proceeding with the maximum search in the region $t_{i-1} = max\left([t_i - \delta P_0, t_i - (2-\delta)\, P_0]\right)$, until the left end of the frame is reached. This method is basically a peak-search approach.

A similar function to position the pitch marks can be implemented in a frame-based manner with overlapping frames of fixed length. With knowledge of the pitch period of the current frame and the position of the last pitch mark in the previous frame, the new pitch mark position can be first estimated and then refined. This enables the real-time usage of the peak-picking approach.

For steady-state periodic signals this approach yields good results. But in cases where the signal has rich harmonic content, the maximum peak may jump from one period to another. This leads to an erroneous positioning of the pitch mark, meaning the pitch mark distance of two succeeding pitch marks does not correspond to the true pitch period. The result are artifacts in the resynthesized signal. Therefore the pitch marks should be carefully positioned and rapid pitch mark jumps should be avoided. Since it is desireable to place the window in such a way that it contains a high amount of signal energy, using a center of energy approach [KZZ10] instead of peak picking yields a more robust pitch mark placement. In cases where the fundamental pitch period changes within a frame, the center of energy approach allows the pitch marks to follow the period change in a continuous manner, rather than jumping to the next maximum peak. There are methods to determine the pitch mark positions using spectral information. A method based on a weighted sum of frequency component group delays, which also ensures that the markers are positioned close to the local energy maximum, is described in [PR99].

A basic pitch mark placement algorithm is implemented in **MATLAB** by the following M-file 6.3. The pitch marks are placed in a frame-based manner at a constant rate corresponding to the fundamental pitch period of the current frame. In case of unvoiced frames the pitch period of the preceding voiced frame is used.

M-file 6.3 (findpitchmarks.m)

```
function [m] = findPitchMarks(x,Fs,F0,hop,frameLen)
% Author: A. von dem Knesebeck
% x         input signal
% Fs        sampling frequency
% F0        fundamental frequencies
% hop       hop size of F0 detection
% frameLen  length of frame

% Initialization
m           = 0;    % vector of pitch mark positions
P0          = zeros (1,length(F0));
index       = 1;
local_m     = [];   % local pitch marker position

% processing frames i
for i = 1:length(F0);
% set pitch periods of unvoiced frames
    if (i==1 && F0(i)==0); F0(i) = 120;     % 120Hz in case no preceding pitch
    elseif (F0(i)==0);     F0(i) = F0(i-1);
    end
    P0(i) = round(Fs/F0(i));                % fundamental period of frame i
    frameRange  = (1:frameLen) + (i-1)*hop; % hopping frame
    last_m = index;                         % last found pitch mark

% beginning periods of 1st frame
    j = 1;  %period number
    if i==1
        % define limits of searchFrame
        searchUpLim = 1 * P0(i);
        searchRange = (1 : searchUpLim);
        [pk,loc]  = max(x(searchRange));
        local_m(j)  = round(loc);

% beginning periods of 2nd - end frame
    else
        searchUpLim = searchUpLim + P0(i);
        local_m(j)  = last_m + P0(i);
    end % beginning periods of 1st - end frame

% remaining periods of 1st - end frame
    index = local_m(1);
    j = 2;  % grain/period number
    while( searchUpLim + P0(i) <= frameRange(end))
        % define range in which a marker is to be found
        searchUpLim = searchUpLim + P0(i);
        local_m(j)  = local_m(j-1) + P0(i);
        index = local_m(j);
        j = j+1;
    end %while frame end not reached
    m = [m local_m];
end % processing frames i

% finishing calculated pitch marks
m = sort(m);
m = unique(m);
m = m(2:end);
```

6.4 Pitch shifting

Introduction

Transposition is one of the basic tools of musicians. When we think about providing this effect by signal-processing means, we need to think about the various aspects of it. For a musician, transposing means repeating a melody after pitch shifting it by a fixed interval. Each time the performer transposes the melody, he makes use of a different register of his instrument. By doing so, not only is the pitch of the sound modified, but also the timbre is affected.

In the realm of DAFX, it is a matter of choice to transpose without taking into account the timbre modification or whether the characteristic timbre of the instrument has to be maintained in each of its registers. The first method could be called "variable timbre transposition," whereas the second approach would be called "constant timbre transposition." To get an insight into the problem we have to consider the physical origins of the audio signal.

The timbre of a sound heavily depends on the organization of its spectrum. A model can be derived from the study of the singing voice. The pitch of a singing voice is determined by the vocal chords and it can be correlated with the set of frequencies available in the spectrum. The timbre of the voice is mainly determined by the vocal cavities. Their effect is to emphasize some parts of the spectrum, which are called formants. A signal model can be derived where an excitation part is modified by a resonance part. In the case of the voice, the excitation is provided by the vocal chords, hence it is related to the frequencies of the spectrum, whereas the resonances correspond to the formants. When a singer transposes a tune, he has, to some extent, the possibility of modifying the pitch and the formants independently. In a careful signal-processing implementation of this effect, each of these two aspects should be considered.

If only the spectrum of the excitation is stretched or contracted, a pitch transposition up or down, with a constant timbre, is achieved. If only the resonances are stretched or contracted, then the pitch remains the same, but the timbre is varied. Harmonic singing relies on this effect. If both excitation and resonance are deliberately and independently altered, then we enter the domain of effects that can be perceived as unnatural, but that might have a vast musical potential.

The separation of a sound into its excitation and resonance part is a complex process that will be addressed in Chapter 8. We will present here methods which simultaneously alter both aspects, such as the harmonizer or pitch shifting by delay-line modulation in Section 6.4.3. A more refined method based on PSOLA, which allows pitch shifting with formant preservation, will be discussed in Section 6.4.4. For more advanced pitch-shifting methods we refer the reader to Chapters 7–11.

Musical applications

Typical applications of pitch shifting in pop music are the correction of the intonation of instruments or singers, as well as the production of an effect similar to a chorus. When the voice of a singer is mixed with copies of itself that are slightly transposed, a subtle effect appears that gives the impression that one is listening to a choir instead of a single singer.

The harmonizer can also produce surprising effects, such as a man speaking with a tiny high-pitched voice or a female with a gritty low-pitched one. Extreme sounds can be produced such as the deep snare drum sound on David Bowie's "Let's Dance" record [Whi99]. It has also been used for scrambling and unscrambling speech [GRH73]. In combination with a delay line and with feedback of the transposed sound to the input, a kind of spiral can be produced where the sound is always transposed higher or lower at each iteration.

A subtle effect, similar to phasing, can be achieved with a set of harmonizers [Dut88] coupled in parallel and mixed to the input sound, as shown in Figure 6.10. The transposition ratio of the n^{th} harmonizer should be set to $1 + nr$ where r is of the order of 1/3000. If f_0 is the pitch of the sound, the outputs of the n^{th} harmonizer will provide a pitch of $f_0 + n\Delta f$, where $\Delta f = rf_0$. If Δf is small enough (a few 1/100 Hz) the interferences between the various outputs of the harmonizers

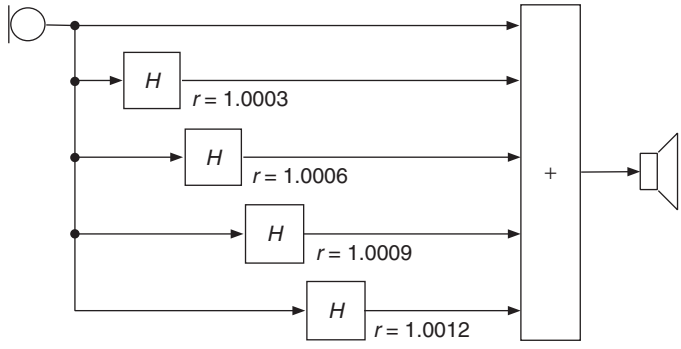

Figure 6.10 A set of harmonizers that produce a phasing-like effect. It is particularly effective for low-pitched (typically 100 Hz) signals of long duration.

will be clearly audible. When applied, for example, to a low-pitched tuba sound, one harmonic after the other will be emphasized. Flanging and chorus effects can also be achieved by setting the pitch control for a very slight amount of transposition (say, 1/10 to 1/5 of a semitone) and adding regeneration [And95, p. 53]. It appears here that tuning an audio effect is very dependent on the sound being processed. It frequently happens that the tuning has to be adjusted for each new sound or each new pitch.

Hans Peter Haller describes in [Hal95, pp. 51–55] some applications of the harmonizer for the production of musical works from Luigi Nono and André Richard.

6.4.1 Historical methods – Harmonizer

The tape-based machines described in 6.3.1 were also able to modify the pitch of sounds while keeping their initial duration. The *Phonogène universel* was bulky and could not find a broad diffusion, but in the middle of the 1970s, a digital device appeared that was called a *Harmonizer*. It implemented in the digital domain a process similar to that of the *Phonogène universel*. From there on the effect became very popular. Since *Harmonizer* is a trade mark of the Eventide company, other companies offer similar devices under names such as *pitch transposer* or *pitch shifter*.

The main limitation of the use of the harmonizer is the characteristic quality that it gives to the processed sounds. Moles states that the operating range of the *Phonogène universel*, used as a pitch regulator, was at least -4 to $+3$ semitones [Mol60, p. 74]. Geslin estimates that the machines available in the late 60s also found application in *musique concrète* at much larger transposition ratios [Ges00].

The digital implementations in the form of the harmonizer might allow for a better quality, but there are still severe limitations. For transpositions of the order of a semitone, almost no objectionable alteration of the sounds can be heard. As the transposition ratio grows larger, in the practical range of plus or minus two octaves, the timbre of the output sound obtains a character that is specific to the harmonizer.

This modification can be heard both in the frequency domain and in the time domain and is due to the modulation of the signal by the chopping window. The spectrum of the input signal is indeed convolved with that of the window. The time-domain modulation can be characterized by its rate and by the spectrum of the window, which is dependent on its shape and its size. The longer the window, the lower the rate and hence the narrower the spectrum of the window and the less disturbing the modulation. The effect of a trapezoidal window will be stronger than that of a smoother one, such as the raised cosine window.

On the other hand, a larger window tends to deliver, through the overlap-add process, audible iterated copies of the input signals. For the transposition of percussive sounds, it is necessary to reduce the size of the window. Furthermore, to accurately replay transients and not smooth them out, the window should have sharp transitions. We see that a trade-off between audible spectral modulation and iterated transients has to be found for each type of sound. Musicians using the computer as a musical instrument might exploit these peculiarities in the algorithm to give their sound a unique flavor.

6.4.2 Pitch shifting by time stretching and resampling

The variable speed replay discussed in Section 6.2 leads to a compression or expansion of the duration of a sound and to a pitch shift. This is accomplished by resampling in the time domain. Figure 6.1 illustrates the discrete-time signals and the corresponding spectra. The spectrum of the sound is compressed or expanded over the frequency axis. The harmonic relations

$$f_i = i \cdot f_{\text{fundamental}} \tag{6.4}$$

of the sound are not altered, but are scaled according to

$$f_i^{\text{new}} = \alpha \cdot f_i^{\text{old}}. \tag{6.5}$$

The amplitudes of the harmonics remain the same $a_i^{\text{new}} = a_i^{\text{old}}$. In order to rescale the pitch-shifted sound towards the original length, a further time-stretching algorithm can be applied to the sound. The result of pitch shifting followed by a time-stretching algorithm is illustrated in Figure 6.11.

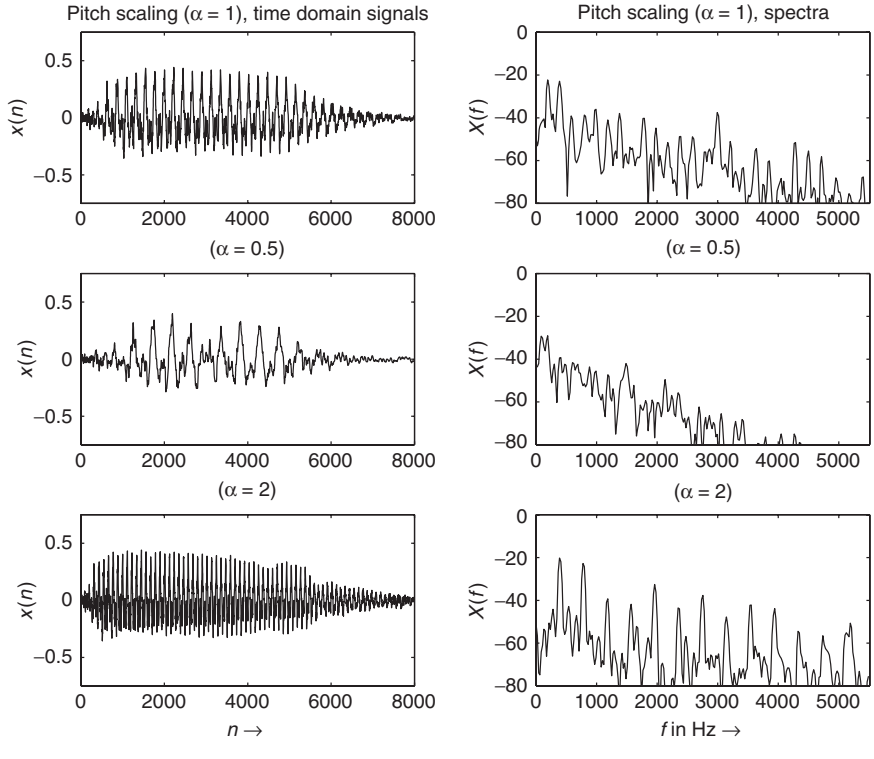

Figure 6.11 Pitch shifting followed by time correction.

The order of pitch shifting and time scaling can be changed, as shown in Figure 6.12. First, a time-scaling algorithm expands the input signal from length N_1 to length N_2. Then a resampling operation with the inverse ratio N_1/N_2 performs pitch shifting and a reduction of length N_2 back to length N_1. The following M-file 6.4 demonstrates the implementation of the SOLA time-scaling and pitch-scaling algorithm:

M-file 6.4 (PitchScaleSOLA.m)

```
% PitchScaleSOLA.m
% Authors: G. De Poli, U. Zölzer, P. Dutilleux
% Parameters:
%    analysis hop size     Sa = 256 (default parmater)
%    block length          N  = 2048 (default parameter)
%    pitch scaling factor  0.25 <= alpha <= 2
%    overlap interval      L  = 256*alpha/2
clear all,close all
[signal,Fs]     =       wavread('x1.wav');
DAFx_in         =       signal';

Sa=256;N=2048;            % time scaling parameters
M=ceil(length(DAFx_in)/Sa);

n1=512;n2=256;            % pitch scaling n1/n2
Ss=round(Sa*n1/n2);
L=256*(n1/n2)/2;

DAFx_in(M*Sa+N)=0;
Overlap=DAFx_in(1:N);

% ****** Time Stretching with alpha=n2/n1******
..... % include main loop TimeScaleSOLA.m
% ****** End Time Stretching ******

% ****** Pitch shifting with alpha=n1/n2 ******
lfen=2048;lfen2=lfen/2;
w1=hanningz(lfen);w2=w1;

% for linear interpolation of a grain of length lx to length lfen
lx=floor(lfen*n1/n2);
x=1+(0:lfen-1)'*lx/lfen;
ix=floor(x);ix1=ix+1;
dx=x-ix;dx1=1-dx;
%
lmax=max(lfen,lx);
Overlap=Overlap';
DAFx_out=zeros(length(DAFx_in),1);

pin=0;pout=0;
pend=length(Overlap)-lmax;
%  Pitch shifting by resampling a grain of length lx to length lfen
while pin<pend
  grain2=(Overlap(pin+ix).*dx1+Overlap(pin+ix1).*dx).* w1;
  DAFx_out(pout+1:pout+lfen)=DAFx_out(pout+1:pout+lfen)+grain2;
  pin=pin+n1;pout=pout+n2;
end;
```

Figure 6.12 Pitch shifting by time scaling and resampling.

6.4.3 Pitch shifting by delay-line modulation

Pitch shifting or pitch transposing based on block processing is described in several publications. In [BB89] a pitch shifter based on an overlap-add scheme with two time-varying delay lines is proposed (see Figure 6.13). A cross-fade block combines the outputs of the two delay lines according to a cross-fade function. The signal is divided in small chunks. The chunks are read faster to produce higher pitches or slower to produce lower pitches. In order to produce a continuous signal output, two chunks are read simultaneously with a time delay equal to one half of the block length. A cross-fade is made from one chunk to the other at each end of a chunk [WG94, pp. 257–259].

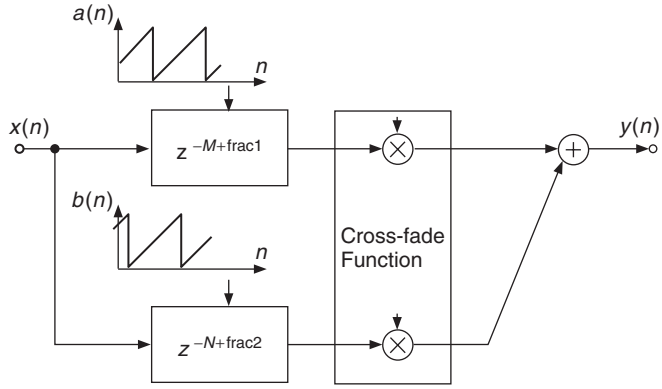

Figure 6.13 Pitch shifting.

The length of the delay lines is modulated by a sawtooth-type function. A similar approach is proposed in [Dat87], where the same configuration is used for time compression and expansion. A periodicity detection algorithm is used for calculating the cross-fade function in order to avoid cancellations during the cross-fades.

An enhanced method for transposing audio signals is presented in [DZ99]. The method is based on an overlap-add scheme and does not need any fundamental frequency estimation. The difference from other applications is the way the blocks are modulated and combined to the output signal. The enhanced transposing system is based on an overlap-add scheme with three parallel time-varying delay lines (see Figure 6.15).

Figure 6.14 illustrates how the input signal is divided into blocks, which are resampled (phase modulation with a ramp-type signal), amplitude modulated and summed yielding an output signal of the same length as the input signal. Adjacent blocks overlap with 2/3 of the block length.

The modulation signals form a system of three 120°-phase shifted raised cosine functions. The sum of these functions is constant for all arguments. Figure 6.15 also shows the topology of the pitch transposer. Since a complete cosine is used for modulation, the perceived sound quality of the processed signal is much better than in simple twofold overlap-add applications using several windows. The amplitude modulation only produces sum and difference frequencies with the base

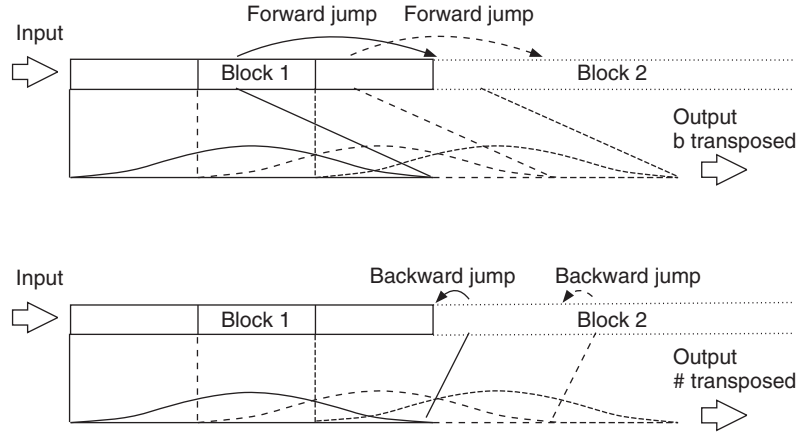

Figure 6.14 Enhanced pitch transposer: block processing, time shifting and overlap-add.

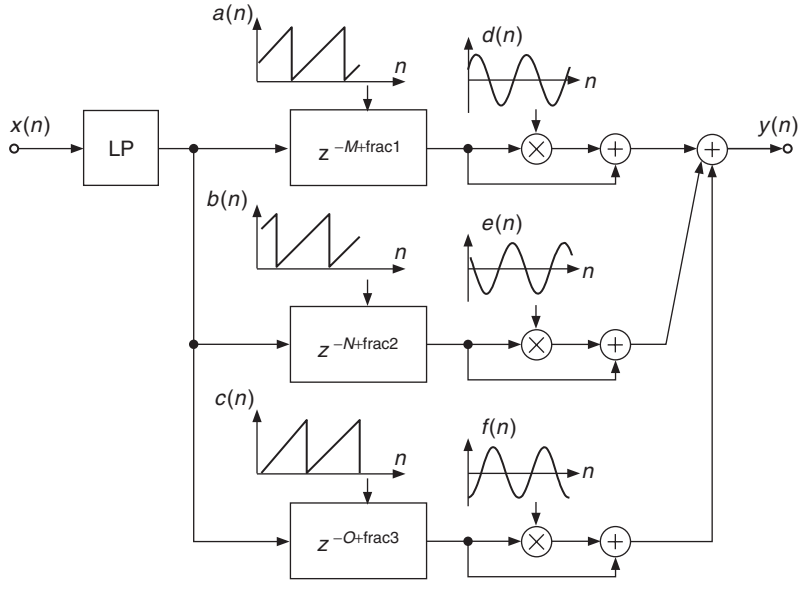

Figure 6.15 Enhanced pitch transposer: block diagram.

frequency of the modulation signal, which can be very low (6–10 Hz). Harmonics are not present in the modulation signal and hence cannot form sum or difference frequencies of higher order. The perceived artifacts are phasing-like effects and are less annoying than local discontinuities of other applications based on twofold overlap-add methods.

If we want to change the pitch of a signal controlled by another signal or signal envelope, we can also make use of delay-line modulation. The effect can be achieved by performing a phase modulation of the recorded signal according to $y(n) = x(n - D(n))$. The modulating factor $D(n) = M + \text{DEPTH} \cdot x_{\text{mod}}(n)$ is now dependent on a modulating signal $x_{\text{mod}}(n)$. With this approach the pitch of the input signal $x(n)$ is changed according to the envelope of the modulating signal (see Figure 6.16).

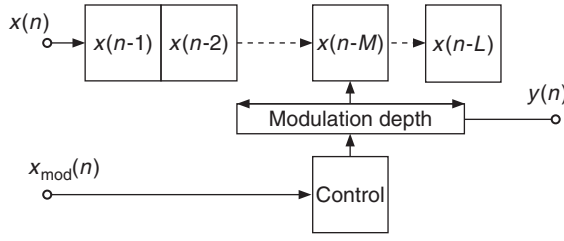

Figure 6.16 Pitch controlled by envelope of signal $x_{\mathrm{mod}}(n)$.

6.4.4 Pitch shifting by PSOLA and formant preservation

This technique is the dual operation to resampling in the time domain, but in this case a resampling of the short-time spectral envelope is performed. The short-term spectral envelope describes a frequency curve going through all amplitudes of the harmonics. This is demonstrated in Figure 6.17, where the spectral envelope is shown. The harmonics are again scaled according to $f_i^{\mathrm{new}} = \beta \cdot f_i^{\mathrm{old}}$, but the amplitudes of the harmonics $a_i^{\mathrm{new}} = \mathrm{env}(f_i^{\mathrm{new}}) \neq a_i^{\mathrm{old}}$ are now determined by sampling the spectral envelope. Some deviations of the amplitudes from the precise envelope can be noticed. This depends on the chosen pitch-shifting algorithm.

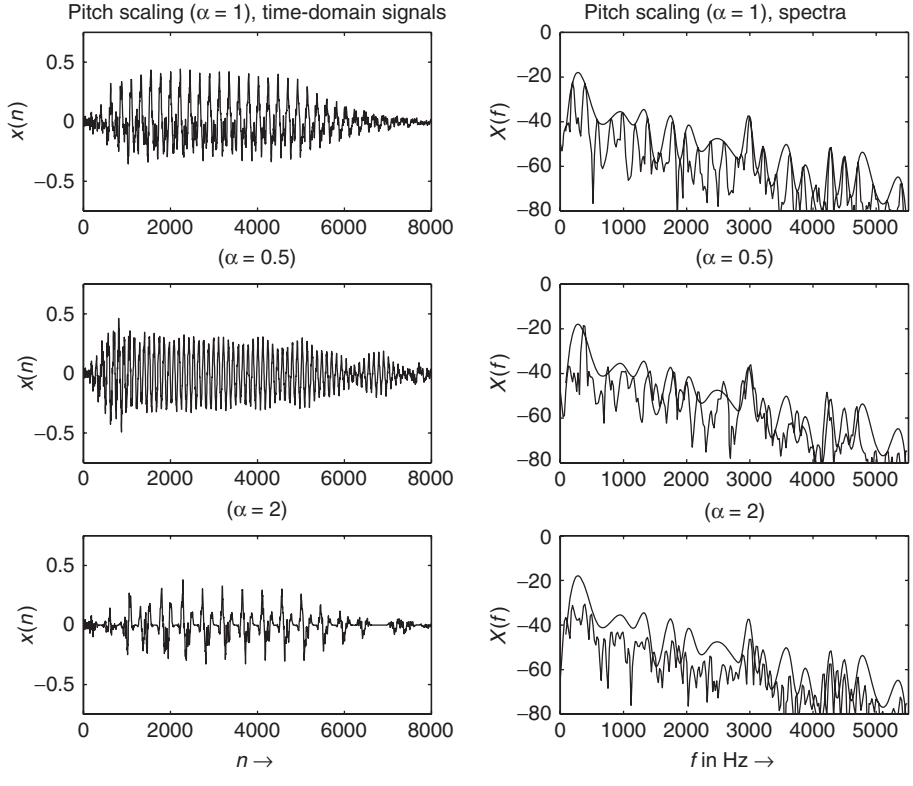

Figure 6.17 Pitch shifting by the PSOLA method: frequency resampling the spectral envelope.

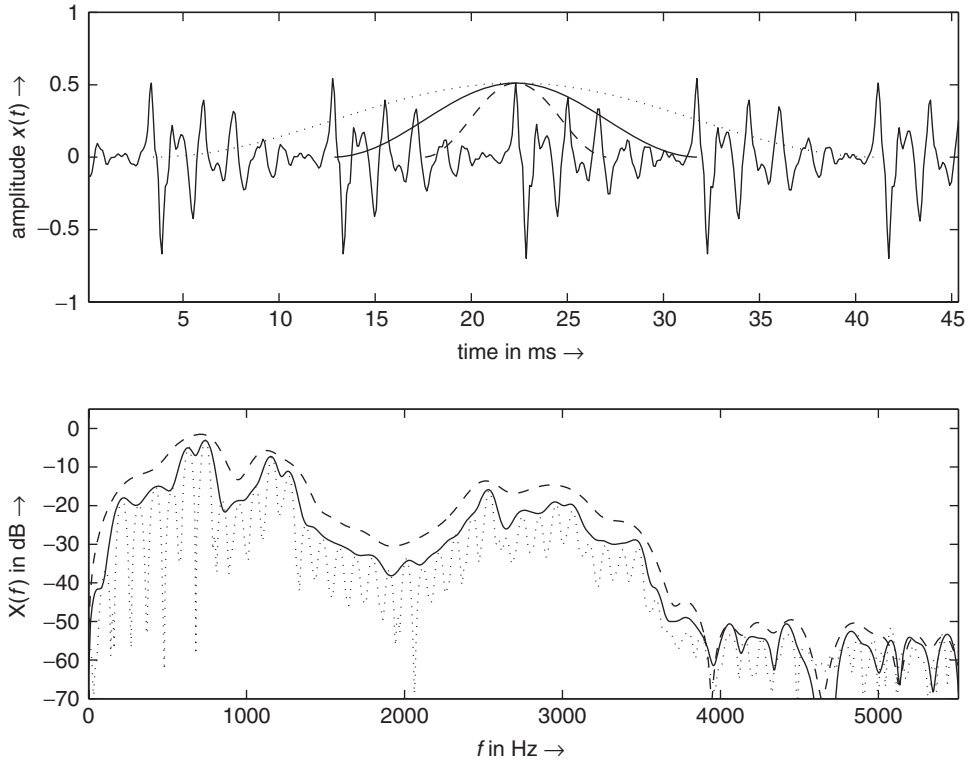

Figure 6.18 Spectrum of segments extracted from a vowel /a/ by using a Hanning window 4 (dotted line), 2 (solid line) and 1 (dashed line) pitch periods long, respectively. It can be noticed that the solid line approximates the local spectral envelope.

The PSOLA algorithm can be conveniently used for pitch shifting a voice sound maintaining the formant position, and thus the vowel identity [ML95, BJ95]. The basic idea consists of time stretching the position of pitch marks, while the segment waveform is not changed. The underlining signal model of speech production is a pulse train filtered by a time-varying filter corresponding to the vocal tract. The input segment corresponds to the filter impulse response and determines the formant position. Thus, it should not be modified. Conversely, the pitch mark distance determines the speech period, and thus should be modified accordingly. The aim of PSOLA analysis is to extract the local filter impulse response. As can be seen in Figure 6.18, the spectrum of a segment extracted using a Hanning window with a length of two periods approximates the local spectral envelope. Longer windows tend to resolve the fine line structure of the spectrum, while shorter windows tend to blur the formant structure of the spectrum. Thus if we do not stretch the segment, the formant position is maintained. The operation of overlapping the segments at the new pitch mark position will resample the spectral envelope at the desired pitch frequency. When we desire a pitch shift by a factor β, defined as the ratio of the local synthesis pitch frequency to the original one $\beta = \tilde{f}_0(\tilde{t})/f_0(t)$, the new pitch period will be given by $\tilde{P}(\tilde{t}) = P(t)/\beta$, where in this case $\tilde{t} = t$ because time is not stretched.

The analysis algorithm is the same as that previously seen for PSOLA time stretching in Section 6.3.3 (see Figure 6.8). The synthesis algorithm is modified (see Figure 6.19) according to the following steps:

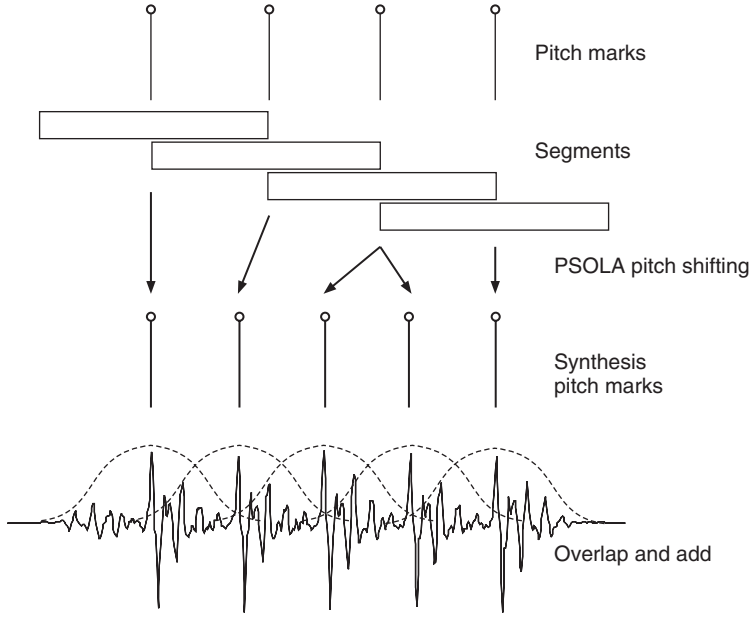

Figure 6.19 PSOLA: synthesis algorithm for pitch shifting.

- For every synthesis pitch mark $\tilde{t}_k$:

  (1) Choice of the corresponding analysis segment i (identified by the time mark t_i) minimizing the time distance $|t_i - \tilde{t}_k|$.

  (2) Overlap and add the selected segment. Notice that some input segments will be repeated for $\beta > 1$ (higher pitch) or discarded when $\beta < 1$ (lower pitch).

  (3) Determination of the time instant $\tilde{t}_{k+1}$ where the next synthesis segment will be centered, in order to preserve the local pitch, by the relation

$$\tilde{t}_{k+1} = \tilde{t}_k + \tilde{P}(\tilde{t}_k) = \tilde{t}_k + P(t_i)/\beta.$$

- For large pitch shifts, it is advisable to compensate the amplitude variation, introduced by the greater or lesser overlapping of segments, by multiplying the output signal by $1/\beta$.

It is possible to combine time stretching by a factor α with pitch shifting. In this case, for every synthesis pitch mark $\tilde{t}_k$ the first step of the synthesis algorithm above presented will be modified by the choice of the corresponding analysis segment i (identified by the time mark t_i), minimizing the time distance $|\alpha t_i - \tilde{t}_k|$.

The PSOLA algorithm is very effective for speech processing and is computationally very efficient, once the sound has been analyzed, so it is widely used for speech synthesis from a database of diphones, for prosody modification, for automatic answering machines etc. For wide variation of the pitch it presents some artifacts. On the other hand, the necessity of a preliminary analysis stage for obtaining a pitch contour makes the real-time implementation of an input-signal modification difficult. Also the estimation of glottal pulses can be difficult. A solution is to place the pitch marks at a pitch synchronous rate, regardless of the true position of the glottal pulses. The resulting synthesis quality will be only slightly decreased (see, for example, Figure 6.20).

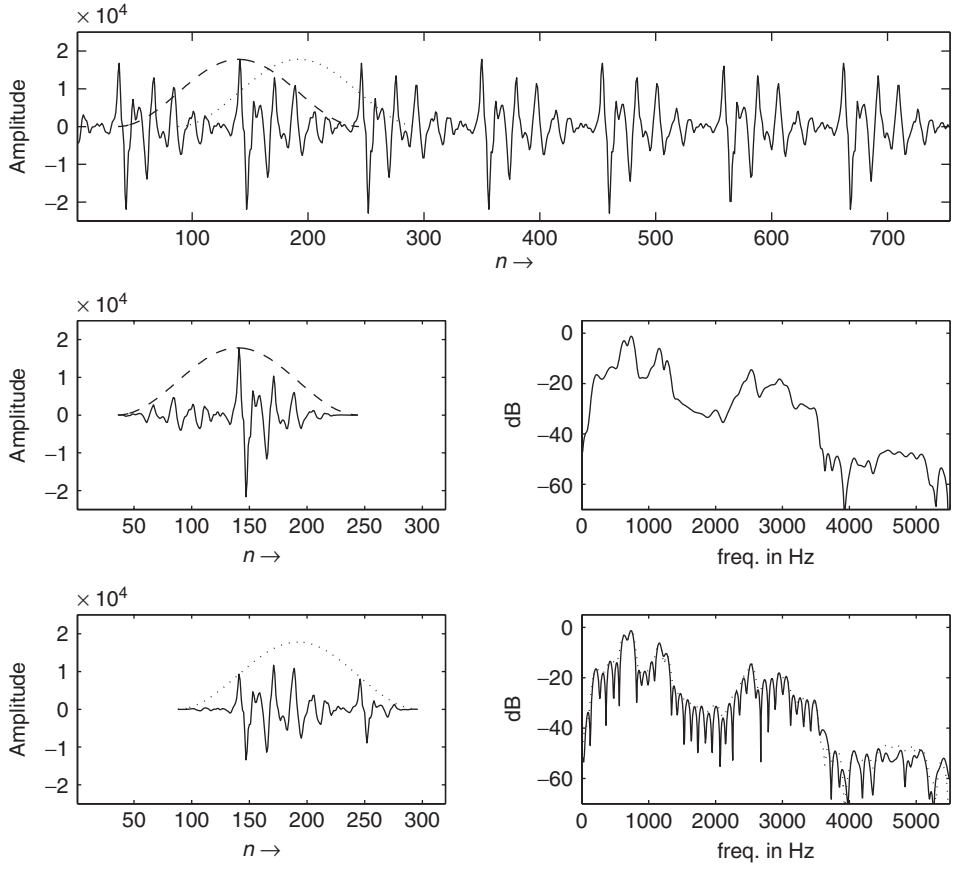

Figure 6.20 Comparison of a segment extracted in the correspondence with glottal pulse with one extracted between pitch pulses.

A further effect that can be obtained by a variation of PSOLA is linear scaling of formant frequencies (see Figure 6.21). In fact, we saw that the time scale of a signal corresponds to an inverse frequency scale. Thus when we perform time scaling of the impulse response of a filter, we inversely scale the frequency of formants. In PSOLA terms, this corresponds to time scaling the selected input segments before overlap and adding the synthesis step, without any change in the pitch marks calculation. To increase the frequencies of formants by a factor γ, every segment should be shortened by a factor $1/\gamma$ by resampling. For example, the average formant frequencies of female adults are about 16% higher than those of male adults, and children's formants are about 20% higher than female formants. Notice that care should be taken when the frequencies increase in order to avoid foldover. Ideally band-limited resampling should be used.

The PSOLA pitch shifter can be used to synthesize multiple voices from one real singer to create a *virtual choir* effect [SPLR02].

The following M-file 6.5 shows the implementation of the basic PSOLA synthesis algorithm. It is based on the PSOLA time-stretching algorithm shown in Section 6.3.3.

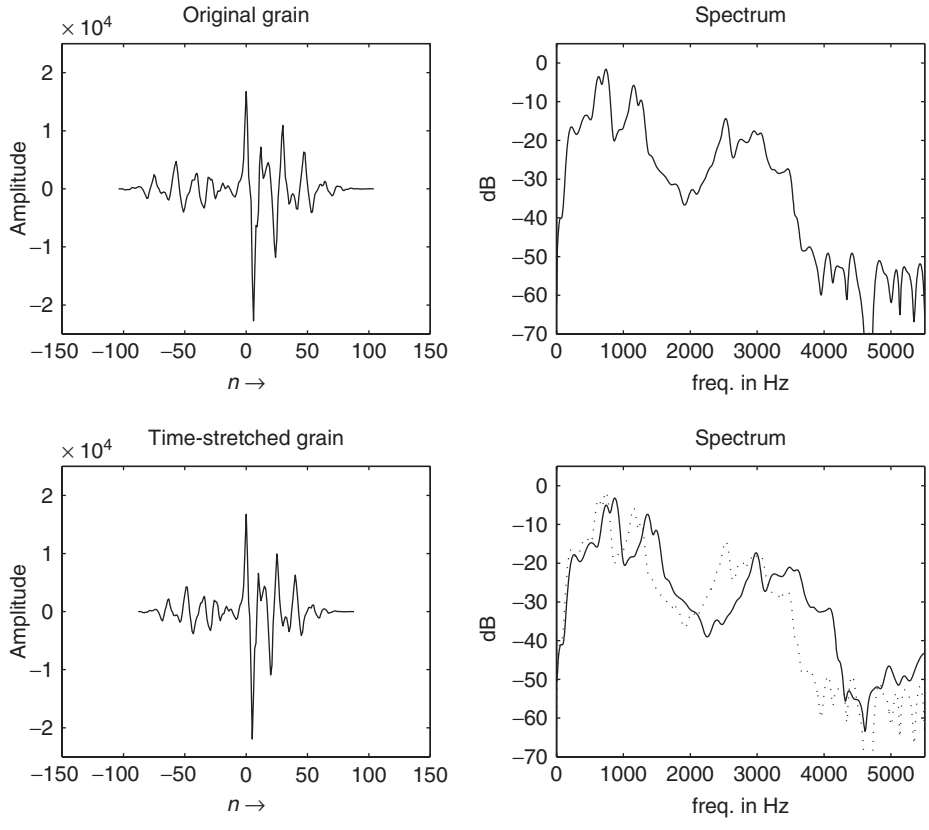

Figure 6.21 PSOLA: variation of PSOLA as linear formant scaling.

M-file 6.5 (psolaf.m)

```
function out=psolaF(in,m,alpha,beta,gamma)
% Authors: G. De Poli, U. Zölzer, P. Dutilleux
%      . . .
%      gamma newFormantFreq/oldFormantFreq
%      . . .
%      the internal loop as
tk = P(1)+1;                    %output pitch mark
while round(tk)<Lout
 [minimum i]=min(abs(alpha*m-tk) );    % find analysis segment
 pit=P(i);pitStr=floor(pit/gamma);
 gr=in(m(i)-pit:m(i)+pit).*hanning(2*pit+1);
 gr=interp1(-pit:1:pit,gr,-pitStr*gamma:gamma:pit);% stretch segm.
 iniGr=round(tk)-pitStr;endGr=round(tk)+pitStr;
 if endGr>Lout, break; end
 out(iniGr:endGr)=out(iniGr:endGr)+gr; % overlap new segment
 tk=tk+pit/beta;
end % end of while
```

6.5 Time shuffling and granulation

6.5.1 Time shuffling

Introduction

Musique concrète has made intensive use of splicing of tiny elements of magnetic tape. When mastered well, this assembly of hundreds of fragments of several tens of milliseconds allows an amalgamation of heterogeneous sound materials, at the limit of the time discrimination threshold. This manual operation, called micro-splicing, is very time-consuming. Bernard Parmegiani suggested in 1980 at the Groupe de Recherches Musicales (GRM) that this could be done by computers. An initial version of the software was produced in the early 80s. After being rewritten, improved and ported several times, it was eventually made available on personal computers in the form of a program called *brassage* in French that will be translated here as *time shuffling* [Ges98, Ges00].

Signal processing

Let us describe here an elementary algorithm for time shuffling that is based on the superposition of two time segments that are picked randomly from the input signal (see Figure 6.22):

(1) Let $x(n)$ and $y(n)$ be the input and output signals.

(2) Specify the duration d of the fragments and the duration $D \geq d$ of the time period $[n - D, n]$ from which the time segments will be selected.

(3) Store the incoming signal $x(n)$ in a delay line of length D.

(4) Choose at random the delay time τ_1 with $d \leq \tau_1 \leq D$.

(5) Select the signal segment x_{1d} of duration d beginning at $x(n - \tau_1)$.

(6) Follow the same procedure (steps 4 and 5) for a second time segment x_{2d}.

(7) Read x_{1d} and x_{2d} and apply an amplitude envelope W to each of them in order to smooth out the discontinuities at the borders.

(8) When the reading of x_{1d} or x_{2d} is finished, iterate the procedure for each of them.

(9) Compute the output as the overlap add of the sequence of x_{1d} and x_{2d} with a time shift of $d/2$.

Musical applications and control

The version described above introduces local disturbances into the signal's actual timing, while preserving the overall continuity of its time sequence. Many further refinements of this algorithm are possible. A random amplitude coefficient could be applied to each of the input segments in order to modify the density of the sound material. The shape of the envelope could be modified in order to retain more of the input time structure or, on the other hand, to smooth it out and blend different events with each other. The replay speed of the segments could be varied in order to produce transposition or glissandi.

At a time when computer tools were not yet available, Bernard Parmegiani magnificently illustrated the technique of tape-based micro-splicing in works such as "Violostries" (1964) or "Dedans-Dehors" (1977) [m-Par64, m-Par77]. The elementary algorithm presented above can be operated in real time, but other off-line versions have also been implemented which offer many more features. They have the ability to merge fragments of any size, sampled from a random field, and of any dimension; from a few samples to several minutes. Thus, apart from generating

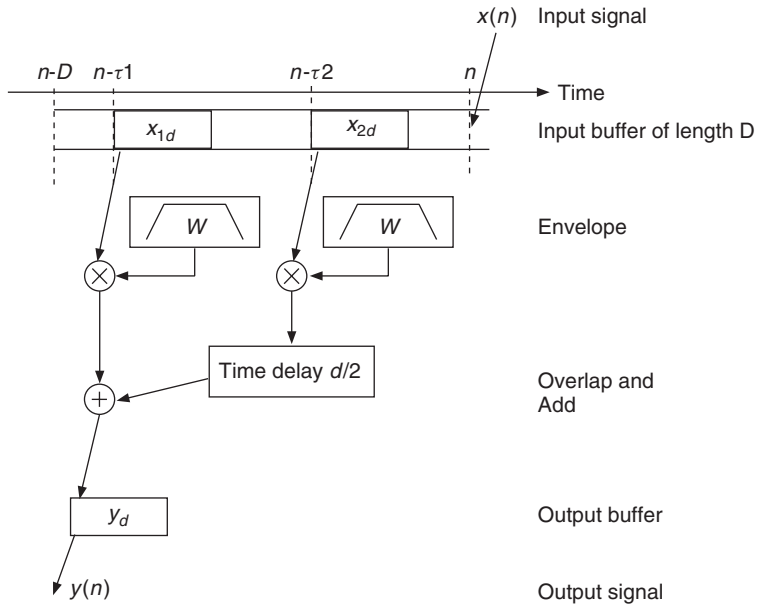

Figure 6.22 Time shuffling: two input segments, selected at random from the past input signal, are overlap-added to produce an output time segment. When one of the input segments is finished, a new one is selected.

fusion phenomena, for which the algorithm was conceived, the software was able to produce cross-fading of textured sound and other sustained chords, infinitely small variations in signal stability, interpolation of fragments with silence or sounds of other types [Ges98]. Jean-Claude Risset used this effect to perform sonic developments from short sounds, such as stones and metal chimes [m-INA3, Sud-I, 3'44'' to 4'38'']; [Ris98, Ges00] and to produce a "stuttering" piano, further processed by ring modulation [m-INA3, Sud-I, 4'30'', 5'45'']. Starting from "found objects" such as bird songs, he rearranged them in a compositional manner to obtain first a pointillistic rendering, then a stretto-like episode [m-INA3, Sud-I, 1'42'' to 2'49''].

6.5.2 Granulation

Introduction

In the previous sections about pitch shifting and time stretching we have proposed algorithms that have limitations as far as their initial purpose is concerned. Beyond a limited range of modification of pitch or of time duration, severe artifacts appear. The time-shuffling method considers these artifacts from an artistic point of view and takes them for granted. Out of the possibilities offered by the methods and by their limitations, it aims to create new sound structures. Whereas the time-shuffling effect exploits the possibilities of a given software arrangement, which could be considered here as a "musical instrument," the idea of building a complex sound out of a large set of elementary sounds could find a larger framework.

The physicist Dennis Gabor proposed in 1947 the idea of the quantum of sound, an indivisible unit of information from the psychoacoustical point of view. According to his theory, a granular representation could describe any sound. Granular synthesis was first suggested as a computer music technique for producing complex sounds by Iannis Xenakis (1971) and Curtis Roads (1978). This technique builds up acoustic events from thousands of sound grains. A sound grain lasts a

brief moment (typically 1 to 100 ms), which approaches the minimum perceivable event time for duration, frequency and amplitude discrimination [Roa96, Roa98, Tru00a].

The granulation effect is an application of granular synthesis where the material out of which the grains are formed is an input signal. Barry Truax has developed this technique [Tru88, Tru94] by first real-time implementation and using it extensively in his compositional pieces.

Signal processing

Let $x(n)$ and $y(n)$ be the input and output signals. The grains $g_k(i)$ are extracted from the input signal with the help of a window function $w_k(i)$ of length L_k by

$$g_k(i) = x(i + i_k)w_k(i), \tag{6.6}$$

with $i = 0, \ldots, L_{k-1}$. The time instant i_k indicates the point where the segment is extracted; the length L_k determines the amount of signal extracted; the window waveform $w_k(i)$ should ensure fade-in and fade-out at the border of the grain and affects the frequency content of the grain. Long grains tend to maintain the timbre identity of the portion of the input signal, while short ones acquire a pulse-like quality. When the grain is long, the window has a flat top and is used only to fade-in and fade-out the borders of the segment.

The following M-files 6.6 and 6.7 show the extraction of short and long grains:

M-file 6.6 (grainSh.m)

```
function y = grainSh(x,init,L)
% Authors: G. De Poli
% extract a short grain
% x    input signal
% init first sample
% L    grain length (in samples)
y=x(init:init+L-1).*hanning(L)';
```

M-file 6.7 (grainLn.m)

```
function y = grainLn(x,iniz,L,Lw)
% Authors: G. De Poli
% extract a long grain
% x    input signal
% init first sample
% L    grain length (in samples)
% Lw   length fade-in and fade-out (in samples)
if length(x) <= iniz+L , error('length(x) too short.'),  end
y = x(iniz:iniz+L-1);                       % extract segment
w = hanning(2*Lw+1)';
y(1:Lw)      = y(1:Lw).*w(1:Lw);            % fade-in
y(L-Lw+1:L) = y(L-Lw+1:L).*w(Lw+2:2*Lw+1); % fade-out
```

The synthesis formula is given by

$$y(n) = \sum_k a_k g_k(n - n_k), \tag{6.7}$$

where a_k is an eventual amplitude coefficient and n_k is the time instant where the grain is placed in the output signal. Notice that the grains can overlap. To overlap a grain g_k (`grain`) at instant $n_k =$ (`iniOLA`) with amplitude `ak`, the following **MATLAB** instructions can be used

```
endOLA = iniOLA+length(grain)-1;
y(iniOLA:endOLA) = y(iniOLA:endOLA) + ak * grain;
```

An example of granulation with random values of the parameters grain initial point and length, output point and amplitude is shown in Figure 6.23. The M-file 6.8 shows the implementation of the granulation algorithm.

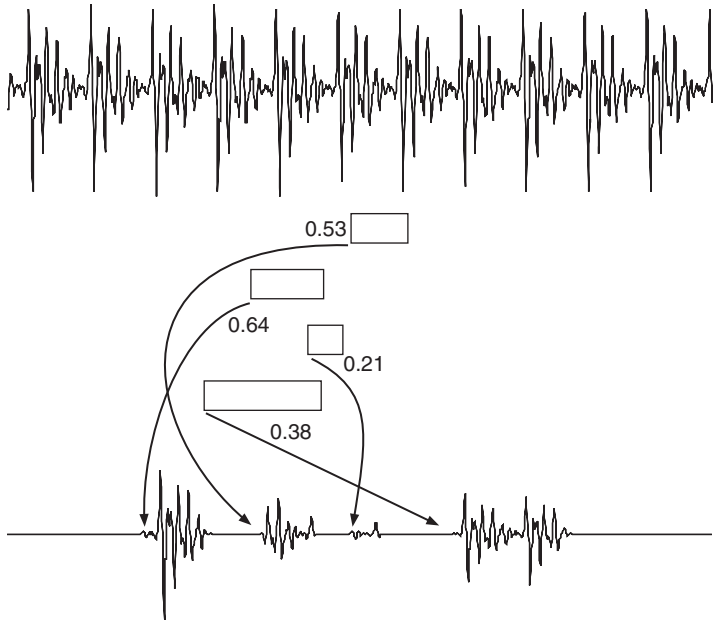

Figure 6.23 Example of granulation.

M-file 6.8 (`granulation.m`)

```
% granulation.m
% Authors: G. De Poli
f=fopen('a_male.m11');
x=fread(f,'int16')';
fclose(f);

Ly=length(x);  y=zeros(1,Ly);                %output signal
% Constants
nEv=4; maxL=200;  minL=50;  Lw=20;
% Initializations
L = round((maxL-minL)*rand(1,nEv))+minL;   %grain length
initIn = ceil((Ly-maxL)*rand(1,nEv));      %init grain
initOut= ceil((Ly-maxL)*rand(1,nEv));      %init out grain
a = rand(1,nEv);                           %ampl. grain
endOut=initOut+L-1;
% Synthesis
```

```
for k=1:nEv,
  grain=grainLn(x,initIn(k),L(k),Lw);
  y(initOut(k):endOut(k))=y(initOut(k):endOut(k))+a(k)*grain;
end
```

This technique is quite general and can be employed to obtain very different sound effects. The result is greatly influenced by the criterion used to choose the instants n_k. If these points are regularly spaced in time and the grain waveform does not change too much, the technique can be interpreted as a filtered pulse train, i.e., it produces a periodic sound whose spectral envelope is determined by the grain waveform interpreted as an impulse response. An example is the PSOLA algorithm shown in the previous Sections 6.3.3 and 6.4.4. When the distance between two subsequent grains is much greater than L_k, the sound will result in grains separated by interruptions or silences, with a specific character. When many short grains overlap (i.e., the distance is less than L_k), a sound texture effect is obtained.

The strategies for choosing the synthesis instants can be grouped into two rather simplified categories: synchronous, mostly based on deterministic functions, and asynchronous, based on stochastic functions. Grains can be organized in streams. There are two main control variables: the delay between grains for a single stream, and the degree of synchronicity among grains in different streams. Given that the local spectrum affects the global sound structure, it is possible to use input sounds that can be parsed in grains without altering the complex characteristics of the original sound, as water drops for stream-like sounds.

It is further possible to modify the grain waveform with a time transformation, such as modulation for frequency shifting or time stretching for frequency scaling [DP91]. The main parameters of granulation are: grain duration, selection order from input sound, amplitude of grains, temporal pattern in synthesis and grain density (i.e., grains per second). Density is a primary parameter, as it determines the overall texture, whether sparse or continuous. Notice that it is possible to extract grains from different sound files to create hybrid textures, e.g., evolving from one texture to another.

Musical applications

Examples of the effect can be found in [m-Wis94c]. Barry Truax has used the technique of granulation to process sampled sound as compositional material. In "The Wings of Nike" (1987) he has processed only short "phonemic" fragments, but longer sequences of environmental sound have been used in pieces such as "Pacific" (1990). In each of these works, the granulated material is time stretched by various amounts and thereby produces a number of perceptual changes that seem to originate from within the sound [Tru00b, m-Tru95].

In "Le Tombeau de Maurice," Ludger Brümmer uses the granulation technique in order to perform timbral, rhythmic as well as harmonic modifications [m-Bru97]. A transition from the original sound color of an orchestral sample towards noise pulses is achieved by reducing progressively the size of the grains. At an intermediate grain size, the pitch of the original sound is still recognizable, although the time structure has already disappeared [m-Bru97, 3'39''-4'12'']. A melody can be played by selecting grains of different pitches and by varying the tempo at which the grains are replayed [m-Bru97, 8'38''-9'10'']. New melodies can even appear out of a two-stage granulation scheme. A first series of grains is defined from the original sample, whereas the second is a granulation of the first one. Because of the stream segregation performed by the hearing system, the rhythmic as well as the harmonic grouping of the grains is constantly evolving [m-Bru97, 9'30''-10'33''].

6.6 Conclusion

The effects described in this chapter are based on the division of the input sound into short segments. These segments are processed by simple methods such as time scaling by resampling, or amplitude multiplication by an envelope. The segment waveform is not changed, thus maintaining the characteristic of the source signal.

Two categories of effects can be obtained, depending on the strategy used to place the segments in time during the synthesis. If the order and organization of extracted segments are carefully maintained, time stretching or pitch shifting can be performed. Basic methods, SOLA and PSOLA, are presented and their characteristics are discussed. These effects aim to produce sounds that are perceived as similar to the original, but are modified in duration or pitch. As often happens with digital audio effects, the artifacts produced by these methods can be used as a method for deformation of the input sound, whilst maintaining its main characteristics. The low computational complexity of time-segment processing allows efficient real-time applications. Nevertheless, these algorithms produce artifacts that limit their scope of application. More advanced methods for time stretching and pitch shifting will be introduced in Chapters 7–11.

The second category changes the organization and the order of the segments to a great extent, and thus leads to time shuffling and granulation. In this case, the input sound can be much less recognizable in the output. The central element becomes the grain with its amplitude envelope and time organization. These techniques can produce results from sparse grains to dense textures, with a very loose relationship with the original sound. It should be noticed that the wide choice of strategies for grain organization implies a sound composition attitude from the user. Thus granulation has became a sort of metaphor for music composition starting from the micro level.

Sound and music

[m-Bru93] L. Brümmer: *The Gates of H*. Computer music. CCRMA 1993. In: CRI, *The Listening Room*, CD edel 0014522TLR, Hamburg, 1995.

[m-Bru97] L. Brümmer: *Le Tombeau de Maurice, for computer-generated tape*. In: *Computer Music* @ CCRMA. CD CCRMAV02, 1997.

[m-Eim62] H. Eimert: *Epitaph für Aikichi Kuboyama*. Electronic music composition, 1962. Studio-Reihe neuer Musik, Wergo, LP WER 60014. Reedition as a CD, Koch/Schwann, 1996.

[m-Fur93] K. Furukawa: *Swim, Swan, composition for clarinet and live-electronics*. ZKM, 1993.

[m-Fur97] K. Furukawa: *Den ungeborenen Göttern*. Multimedia-Opera, ZKM, 1997.

[m-INA3] J.-C. Risset: *Sud, Dialogues, Inharmonique, Mutations*. CD INA C1003.

[m-Par64] B. Parmegiani: *Violostries*, 1964. IDEAMA CD 051 Target Collection, ZKM and CCRMA, 1996.

[m-Par77] B. Parmegiani: *Dedans-Dehors*, 1977. INA-GRM.

[m-Sch98] P. Schaeffer and G. Reibel: *Solfège de L'objet Sonore*. Booklet + 3 CDs. First published 1967. INA-GRM, 1998.

[m-Tru95] B. Truax: Granular time-shifting and transposition composition examples. In *Computer Music Journal*, Volume 19 Compact Disc, Index 6, 1995.

[m-Wis94c] T. Wishart: *Audible Design*, Sound examples. CD. Orpheus the Pantomime, York, 1994.

References

[And95] C. Anderton. *Multieffects for Musicians*. Amsco Publications, 1995.

[BB89] K. Bogdanowicz and R. Blecher. Using multiple processors for real-time audio effects. In *Proc. AES 7th Int. Conf*., pp. 337–342, 1989.

[BJ95] R. Bristow-Johnson. A detailed analysis of a time-domain formant-corrected pitch shifting algorithm. *J. Audio Eng. Soc*., 43(5): 340–352, 1995.

[Car92] T. Cary. *Illustrated Compendium of Musical Technology*. Faber and Faber, 1992.

[Chi82] M. Chion. *La Musique Électroacoustique*. QSJ No 1990, PUF, 1982.

[CR83] R. E. Crochiere and L. R. Rabiner. *Multirate Digital Signal Processing*. Prentice-Hall, 1983.

[Dat87] J. Dattoro. Using digital signal processor chips in a stereo audio time compressor/expander. In *Proc. 83rd AES Convention*, Preprint 2500, 1987.

[DP91] G. De Poli and A. Piccialli. Pitch-synchronous granular synthesis. In G. De Poli, A. Piccialli, and C. Roads (eds), *Representations of Musical Signals*, pp. 187–219. MIT Press, 1991.

[Dut88] P. Dutilleux. Mise en œuvre de transformations sonores sur un système temps-réel. Technical report, Rapport de stage de DEA, CNRS-LMA, June 1988.

[DZ99] S. Disch and U. Zölzer. Modulation and delay line based digital audio effects. In *Proc. DAFX-99 Digital Audio Effects Workshop*, pp. 5–8, 1999.

[End97] B. Enders. *Lexikon Musikelektronik*. Atlantis Schott, 1997.

[Gas87] P. S. Gaskell. A hybrid approach to the variable speed replay of digital audio. *J. Audio Eng. Soc*., 35: 230–238, April 1987.

[Ges98] Y. Geslin. Sound and music transformation environments: A twenty-year experiment at the "Groupe de Recherches Musicales." In *Proc. DAFX-98 Digital Audio Effects Workshop*, pp. 241–248, 1998.

[Ges00] Y. Geslin. About the various types of Phonogènes. GRM, Personal communication, 2000.

[GRH73] T. A. Giordano, H. B. Rothman and H. Hollien. Helium speech unscramblers – a critical review of the state of the art. *IEEE Trans. Audio and Electroacoust*., AU-21(5), 1973.

[Hal95] H. P. Haller. *Das Experimental Studio der Heinrich-Strobel-Stiftung des Südwestfunks Freiburg 1971-1989, Die Erforschung der Elektronischen Klangumformung und ihre Geschichte*. Nomos, 1995.

[HM91] Don Hejna and Bruce R. Musicus. The SOLAFS time-scale modification algorithm. Technical report, BBN, July 1991.

[HMC89] C. Hamon, E. Moulines and F. Charpentier. A diphone synthesis system based on time-domain prosodic modifications of speech. In *Proc. ICASSP*, pp. 238–241, 1989.

[KZZ10] A. von dem Knesebeck, P. Ziraksaz and U. Zölzer. High quality time-domain pitch shifting using PSOLA and transient preservation. In *Proc. of the 129th AES Convention*, 2010.

[Lar98] J. Laroche. Time and pitch scale modifications of audio signals. In M. Kahrs and K.-H. Brandenburg (eds), *Applications of Digital Signal Processing to Audio and Acoustics*, pp. 279–309. Kluwer, 1998.

[Lee72] F. F. Lee. Time compression and expansion of speech by the sampling method. *J. Audio Eng. Soc*., 20(9): 738–742, 1972.

[LJ04] C. Y. Lin and J. S. R. Jang. A two-phase pitch marking method for TD-PSOLA synthesis. In *8th Int. Conf. Spoken Lang. Process*., 2004.

[Mas98] D. C. Massie. Wavetable sampling synthesis. In M. Kahrs and K.-H. Brandenburg (eds), *Applications of Digital Signal Processing to Audio and Acoustics*, pp. 311–341. Kluwer, 1998.

[MC90] E. Moulines and F. Charpentier. Pitch synchronous waveform processing techniques for text-to speech synthesis using diphones. *Speech Commun*., 9(5/6): 453–467, 1990.

[McN84] G. W. McNally. Variable speed replay of digital audio with constant output sampling rate. In *Proc. 76th AES Convention*, Preprint 2137, 1984.

[MEJ86] J. Makhoul and A. El-Jaroudi. Time-scale modification in medium to low rate speech coding. In *Proc. ICASSP*, pp. 1705–1708, 1986.

[ML95] E. Moulines and J. Laroche. Non-parametric technique for pitch-scale and time-scale modification of speech. *Speech Commun*., 16: 175–205, 1995.

[Mol60] A. Moles. *Les musiques Expérimentales*. Trad. D. Charles. Cercle d'Art Contemporain, 1960.

[MVV06] W. Mattheyses, W. Verhelst and P. Verhoeve. Robust pitch marking for prosodic modification of spech using TD-PSOLA. In *Proc. IEEE Benelux/DSP Valley Signal Process. Symp*., 2006.

[Pou54] J. Poullin. L'apport des techniques d'enregistrement dans la fabraction de matières et formes musicales nouvelles. Applications à la musique concrète. *L'Onde Électrique*, 34(324): 282–291, 1954.

[PR99] G. Peeters and X. Rodet. SINOLA: A new analysis/synthesis method using spectrum peak shape distortion, phase and reassigned spectrum. In *Proc. ICMC*, pp. 153–156, 1999.

[PS57] J. Poullin and D. A. Sinclair. *The Application of Recording Techniques to the Production of New Musical Materials and Forms. Application to "Musique Concrète"*. National Research Council of Canada, 1957. Technical Translation TT-646, pp. 1–29.

[Ris98] J.-C. Risset. Example of the musical use of digital audio effects. In *Proc. DAFX-99 Digital Audio Effects Workshop*, pp. 254–259, 1998.

[Roa96] C. Roads. *The Computer Music Tutorial*. MIT Press, 1996.

[Roa98] C. Roads. Micro-sound, history and illusion. In *Proc. DAFX-98 Digital Audio Effects Workshop*, pp. 260–269, 1998.

[RW85] S. Roucos and A. M. Wilgus. High quality time-scale modification for speech. In *Proc. ICASSP*, pp. 493–496, 1985.

[Sch73] P. Schaeffer. *La Musique Concrète*. QSJ No 1287, PUF 1973.

[SPLR02] N. Schnell, G. Peeters, S. Lemouton and X. Rodet. Synthesizing a choir in real-time using pitch synchronous overlap add (PSOLA). In *Proc. 1st IEEE Benelux Workshop Model based Proc. Coding Audio*, 2002.

[Spr55] A. M. Springer. Ein akustischer Zeitregler. *Gravesaner Blätter*, (1): 32–27, July 1955.

[Spr59] J. M. Springer. Akustischer Tempo- und Tonlagenregler. *Gravesaner Blätter*, (13): 80, 1959.

[Tru88] B. Truax. Real-time granular synthesis with a digital signal processor. *Comp. Music J*., 12(2): 14–26, 1988.

[Tru94] B. Truax. Discovering inner complexity: time-shifting and transposition with a real-time granulation technique. *Comp. Music J*., 18(2): 28–38, 1994.

[Tru00a] B. Truax. http://www.sfu.ca/truax/gran.html. Granular synthesis, 2000.

[Tru00b] B. Truax. http://www.sfu.ca/truax/gsample.html. Granulation of sampled sounds, 2000.

[VR93] W. Verhelst and M. Roelands. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *Proc. ICASSP*, pp. 554–557, 1993.

[WG94] M. Warstat and T. Görne. *Studiotechnik – Hintergrund und Praxiswissen*. Elektor-Verlag, 1994.

[Whi99] P. White. *Creative Recording, Effects and Processors*. Sanctuary Publishing, 1999.