DRIVING PITCH-SHIFTING AND TIME-SCALING ALGORITHMS WITH ADAPTIVE AND GESTURAL TECHNIQUES

Arfib D., Verfaille V.

CNRS-LMA
31, chemin Joseph Aiguier
F-13402 Marseille Cedex 20, FRANCE
{arfib, verfaille}@lma.cnrs-mrs.fr

ABSTRACT

This article intends to demonstrate how a specific digital audio effect can benefit from a proper control, be it from sounds and/or from gesture. When this control is from sounds, it can be called "adaptive" or "sound automated". When this control is from gesture, it can be called "gesturally controlled". The audio effects we use for this demonstration are time-scaling and pitch-shifting in the particular contexts of vibrato, prosody change, time unfolding and rythm change.

1. INTRODUCTION

The matter of gestural control of audio systems is a very important subject (see the new COST action named Congas[1]). A link with digital audio effects is easily done when musical instruments are concerned. We here demonstrate how time-scaling and pitch-shifting algorithms may benefit from a gestural control. Moreover the recent research on adaptive effects allows new connections, where parameters extracted from the treated sound can influence either the process or the mapping between gesture and sound. This powerful combination opens new ways in the control of effects.

2. IMPLEMENTATIONS

Time-scaling and pitch-shifting are dual transformations of sound and need specific techniques, either to time-scale a signal with no pitch-shifting and to pitch-shift a signal with no time-scaling. Several digital techniques can be used to time-scale as well as to pitch-shift a sound [1]. Some of them work in the temporal domain (SOLA [2], PSOLA [3]), while others work in the frequential domain (phase vocoder [4, 5], spectral line estimation such as SMS [6], Additive [7], SAS [8]). Some need a pre-analysis, for example to compute the pitch (PSOLA, spectral line estimation) whereas others do not (SOLA, phase vocoder). Several level of analysis can be performed, depending upon the need of a separation between harmonic partials and the noisy component, and the extraction of micro-variations (jitter) as well as macro-variations (vibrato) of frequency.

SOLA uses grains extracted from the original sound: it varies the analysis to synthesis hop size ratio for time-scaling, and resamples each grain for fitch-shifting. PSOLA relies on a synchronization between successive grains which is based on a pitch estimation. Using a phase unfolding, the phase vocoder can also be viewed as a bank filter or a grain processing. The spectral line estimation allows to work on independent components of the sound: the harmonic part and the residual part.

Each of the previous techniques depend upon control values related to time and frequency variations. For time-scaling, these values are dual and called hop size: one for the analysis and one for the synthesis. Usually, one tries to keep the synthesis hop size constant to ensure that the synthesis sound envelope has no amplitude modulation due to the overlap-add of synthesis windows. For pitch-shifting, the main control value is the ratio between the processed frequency and the original frequency. Additional computation provides new control values such as the scaling ratio of the spectral shape (whenever Cepstrum [9] or LPC [10] is used to preserve the formant structure). The possible extraction of jitter, vibrato and noise descriptors gives additional control on these components.

3. THE MATTER OF MUSIC

One has to consider two very different things in the time-frequency domain: the unfolding of a sound with time and the shifting of frequencies.

3.1. Unfolding a sound with time

The temporal evolution of a sound is crucial for the recognition of its properties. The rough reversal or slowing-down of a sound is of course an indication of this, but very subtle variations are also a cause of a big perception change. Vowels and consonants work in two different processes: a consonant isolated from its context is hard to follow or even to hear, and it is also very sensitive to the time evolution: a plosive must be plosive. Rhythm can be lost during a time-scaling process if no attention is taken to preserve recognisable patterns. A vibrato is very sensitive to time-scaling, as only a very short range ($5 - 8Hz$) gives rise to a pleasant sensation. This means that a vibrato should be a variable independent from the time-scaling.

3.2. Shifting the frequencies

Shifting abruptly all the frequencies of a sound often gives rise to a Donald Duck effect. The spectrum evolution has itself two different sides: the harmonic content and the spectral shape. Depending upon the context, one may have to separate the source and filter to independently work on them. The subtle evolution of pitch is

[1] ConGAS: Gesture Controlled Audio Systems, COST Action number 287

linked to expressiveness; fine changes in the micro-modulation of pitch often produce a big change of expressiveness.

4. GESTURAL CONTROL

In order to make a link between a gestural device and a pitch-shifting/time-scaling algorithm, one first has to define the different steps needed to musically control sounds.

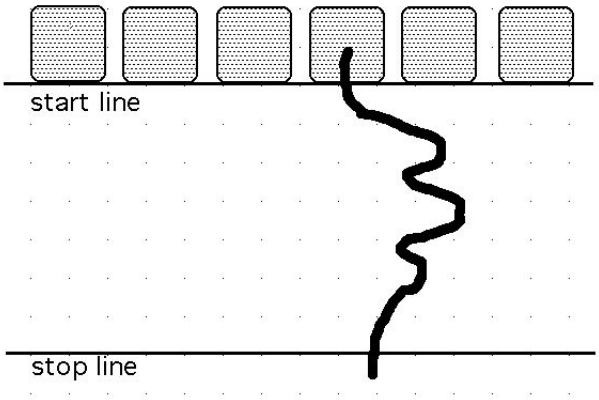

Figure 1: *Example of gesture controlling effects on the Wacom tablet. Time index of the sound is controlled vertically, and pitch-shift horizontally.*

First one has to be able to make the choice of a sound. This can be performed by a selection gesture, and the natural way to do it is to click to initiate the choice of a sound. Then one has to unfold with time this sound in three steps: starting a sound, exploring it and deselecting it. The start can obviously done when leaving the bottom line of a selection square, and deselecting by crossing of a specific stop line.

The system uses a graphic interface (*cf.* fig. 1) designed by J.-M. Couturier. The (x, y) coordinates of the tablet pen are represented onto the screen by a pointer; the user can choose one sound by pressing the pen when the pointer is over one of the six zones at the top of the screen. Then, he/she can play the selected sound by moving between the two horizontal lines: the vertical coordinate controls the time index in the sound, and the horizontal coordinate controls the pitch in order to play vibrato. The sound stops when the "stop line" is crossed.

Due to the nature of the time scaling algorithm, only one coordinate is needed to situate a pointer inside the sound, or its time-frequency representation. These steps have been implemented using *Max/MSP* with a Wacom graphic tablet, and as such this instrument is a very good benchmark test for straight time-scaling algorithms.

The algorithm we have chosen is the SOLA. There are two reasons for this choice: it is very robust, as it does not need any previous pitch extraction or phase unwrapping. As it is a time-domain technique, it only needs a scanning of a table and of the signal. Moreover it can give artefacts which can be used as musical facts which can be in the gesture control of the sounds.

The way we have implemented this algorithm in *Max/MSP* is to generate two series of windowed grains, and to overlap them (*cf.*

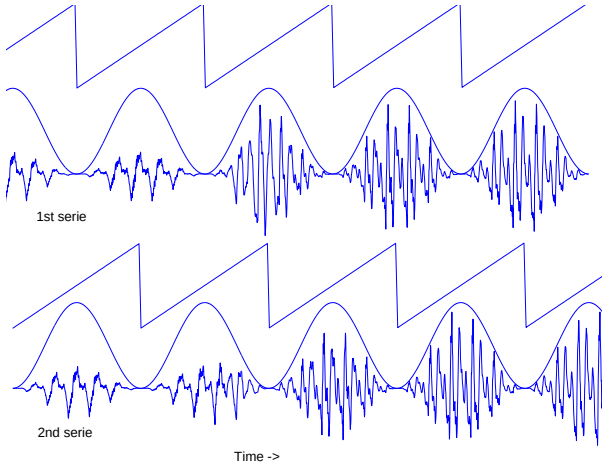

Figure 2: *Sawtooth signal used for controlling the SOLA algorithm. The slope of the sawtooth is proportional to the pitch-shift ratio. The signal out of two overlapping "jog-shuttle" are represented.*

fig. 2). It is also possible to use four series, each one desynchronised by a quarter of the time length of a grain. More specifically for each series, we have used a sawtooth generator, which triggers a pointer to a place in the sound. This pointer is then incremented according to a proportional value of the sawtooth signal. The pitch shifting effect is realised by choosing a different factor for this ratio. This sawtooth signal is also used as an index in a window function to shape the grain.

Figure 3: *Transfer function of the high-pass filter used to remove the dc component of the pitch-shift.*

However, one also wants to make micro-variations such as a vibrato with a pitch-shifting effect. The vibrato is extracted from the gesture by a high-pass filter which removes the dc component of the horizontal coordinate, and the very high frequencies which are not relevant to a vibrato movement. The gesture is sampled at a $1kHz$ rate, and the filter is either a FIR filter or an IIR Chebyshev filter of order 2, similar to what has previously been used to extract the vibrato from a pitch detection of sung voice [11].

A simulation has been first done in *Matlab*, using academic signals. A simplified version of this filter has been finally used in the *Max/MSP* implementation, using the direct placement of two poles and two zeros on the horizontal axis of the x coordonate of the z-plane. The zeros are placed on $(1, 0)$ and the pole on $(r, 0)$ of this plane. This insures that the dc component is removed, and the value of r must be a compromise between a fast answer and

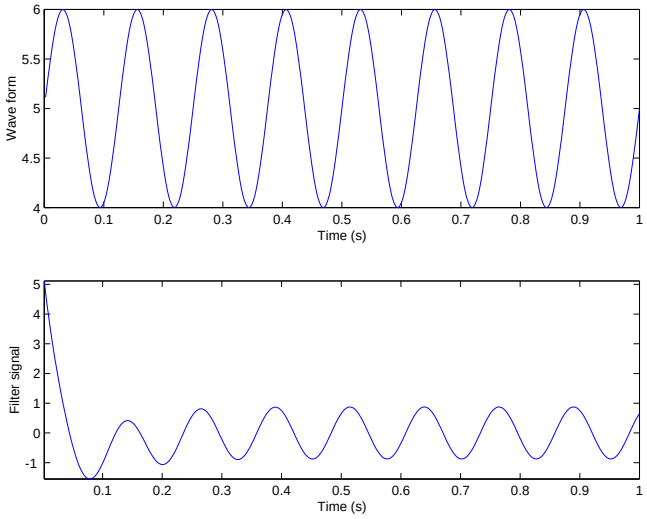

Figure 4: *Academic gesture signal and filtered signal. The dc component of the pitch-shift is removed.*

a reasonable cutoff frequency which does not remove the vibrato frequencies.

As we want to apply pitch variations in a manual way, the sounds that are chosen must be recorded without vibrato. But it is also possible, especially with sung voices to use the natural vibrato superimposed with the manual one.

The musical use of such an instrument may look very simple, but the instrument itself needs some musical practise to be able to look for steady parts of sounds where vibrato can be applied and parts of sound where a specific time stretching is carefully used. Of course many other effects can be applied instead of a vibrato, and with a Wacom tablet, we even have two more gestural values: the pressure that is applied to the stylet and the angle of the stylet.

5. ADAPTIVE EFFECT WITH GESTURAL CONTROL

The principle of adaptive digital audio effects is quite simple: features are extracted from the sound signal $x(n)$, and drive the effect control using specific mapping [12, 13]. This mapping parameters are modified by gesture data $g(n)$ (*cf.* fig. 5).

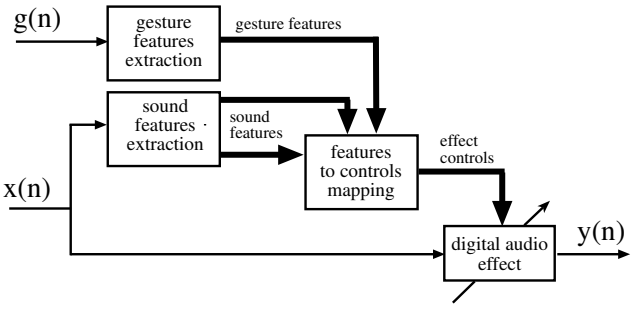

Figure 5: *Diagram of adaptive digital audio effect (A-DAFx) using gesture control to change the mapping between sound features and effect controls.*

This principle of control can be used for any effect. In this paper, we focus on adaptive and gestural control of pitch-shifting and time-scaling, once again using the SOLA technique (however, any "better sounding" technique is also good for this purpose).

5.1. Pitch-shifting and prosody/intonation change

The prosody is defined by linguists as the expressivity of the spoken voice, according to several parameters. Among these parameters, there is the intonation, which is given by the fundamental frequency F_0 (*cf.* fig. 6). The intonation can be described at several levels from the most general pattern to the variations of F_0: the global pattern, the constituent structure, the relief and the micro-prosody. Changing the prosody can be done by changing the intonation, at one of these levels, and sound level as well as time. We now only focus on intonation change.

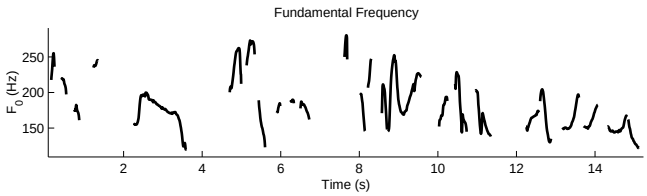

Figure 6: *Fundamental frequency of a spoken voice "lalula".*

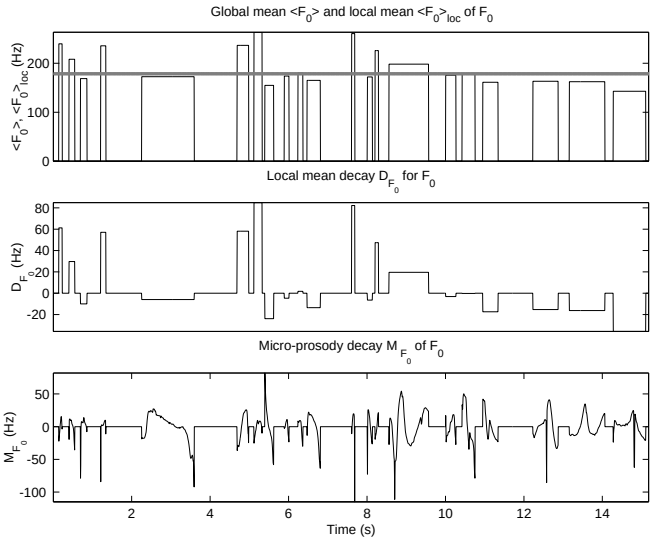

Figure 7: *Fundamental frequency decomposition using a simple voice/unvoiced mask.*

We propose to apply a prosody change using a pitch-shifting which ratio varies according to one of the intonation description level. The fundamental frequency F_0 can be considered as the sum of three components: the mean value $< F_0 >$ of F_0 over the whole sound, the macro-prosody D_{F_0} given by the decay between this global mean and the local mean of F_0 over one phonem, and the micro-prosody M_{F_0} given by the decay between D_{F_0} and F_0 (*cf.* fig. 7). The expression of the pitch depending on time is:

$$F_0(t) =< F_0 > +D_{F_0}(t) + M_{F_0}(t) \quad (1)$$

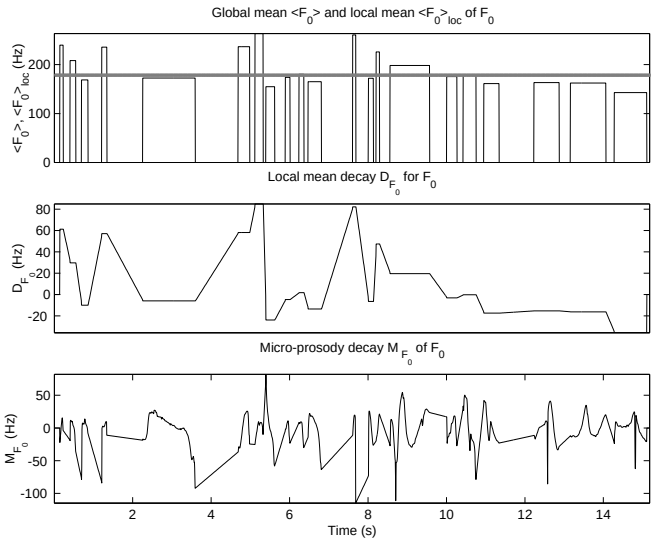

Figure 8: *Fundamental frequency decomposition using an improved voice/unvoiced mask.*

Notice that it can be useful to erase the micro-prosody jumps between voiced and unvoiced portions of the sound, in order to avoid rapid pitch-shifts of the sound. This is achieved by using a modified voiced/unvoiced mask (*cf.* fig. 8), that replaces the jumps between two voiced parts by segment-lines.

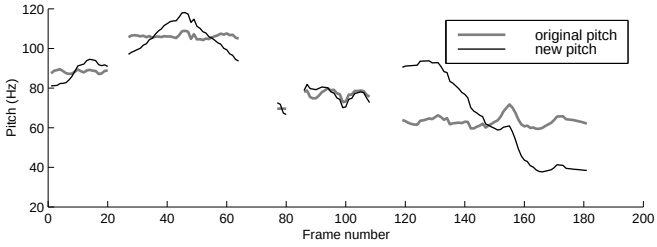

Figure 9: *Pitch-shifting that locally flattens the intonation, using* $\beta = 0$, $\alpha = 1$.

The pitch can be modified by changing the amplitude of any component of its decomposition:

$$\overline{F_0}(t) =< F_0 > +\alpha(t)D_{F_0}(t) + \beta(t)M_{F_0}(t) \qquad (2)$$

with the pitch-shifting ratio $\gamma(t) = \frac{\overline{F_0}(t)}{F_0(t)}$ given by the proportion between the new pitch $\overline{F_0}(t)$ and the original pitch $F_0(t)$. That way, one can erase the pitch variations, and obtain flat intonation on each portion of the segmented sound using $\beta = 0$, $\alpha = 1$ (*cf.* fig. 9), or over the whole sound using $\beta = 0$, $\alpha = 0$ (*cf.* fig. 10). On can also inverse the intonation, locally (*cf.* fig. 11) or globally (*cf.* fig. 12). Using phase vocoder and signal segmentation techniques, we extracted descriptors of intonation for each segment of a non-tempered musical sentence of voice, such as the mean pitch, approximations of the fundamental frequency with polynomials, mean and variance between the fundamental frequency and its approximation. Other features can replace the micro-prosody value

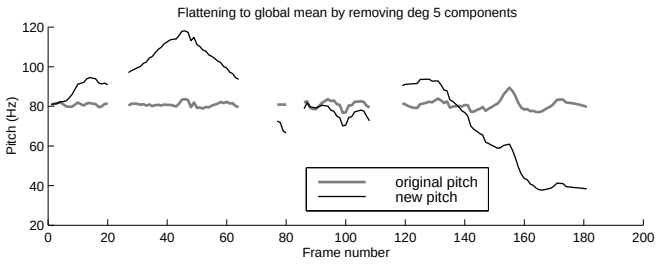

Figure 10: *Pitch-shifting that globally flattens the intonation, using* $\beta = 0$, $\alpha = 0$.

μ_{F_0}, such as the RMS or the centroid, in order to impose another prosody to the sound.

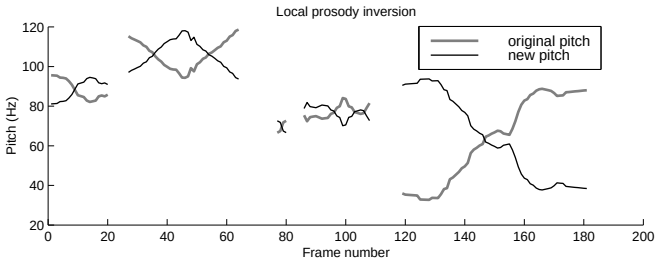

Figure 11: *Pitch-shifting that locally inverse the intonation, using* $\beta = -1$, $\alpha = 1$.

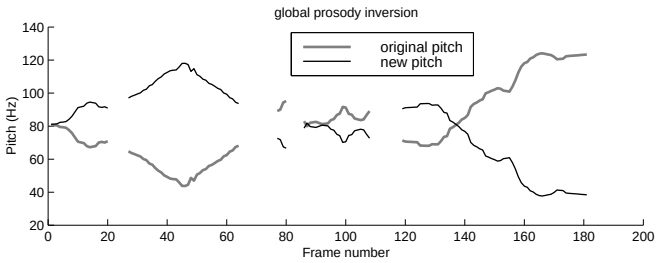

Figure 12: *Pitch-shifting that globally inverse the intonation, using* $\beta = -1$, $\alpha = -1$.

The pitch-shifting ratio curve can be controlled or transformed with time, using a gesture transducer (such as a joystick) to modify the $\alpha(t)$ and $\beta(t)$ values, to replace the micro-prosody curve $M_{F_0}(t)$ by a gesturally controlled combination of other sound features. The prosody is then directly affected by gesture.

Notice that by combining intonation change, amplitude change and non-linear time-scaling, one obtain a prosody change effect.

5.2. Time-scaling and rhythm change

The non-linear time-scaling is obtained while using a time-varying time-scaling ratio $\gamma(t)$; it affects the rythm of the sound by locally changing its speed [12]. A synchronisation constraint can be added in order to preserve the global sound duration. Gesture control is then applied onto the time-scaling control curve $\gamma(t)$. A first way

is to compute the control curve iv by interpolating between the varying ratio $\gamma(t)$ and 1 (corresponding to no time-scaling), thus respecting the constraint of synchronisation. A second way is to modify the curve with gesture data, by addition or multiplication by a value, thus not respecting anymore the synchronisation constraint.

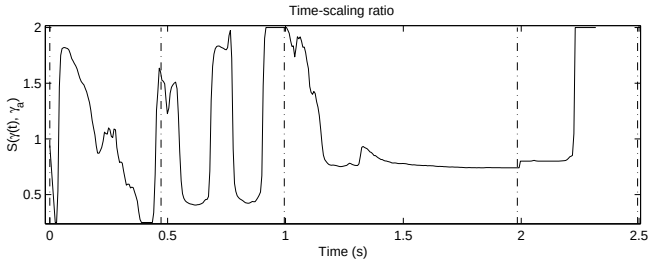

Figure 13: *Varying time-scaling ratio $\gamma(t)$ used as control curve for adaptive time-scaling. Dashed lines correspond to synchronization marks.*

The implementation consists in a time-scaling algorithm with a constant synthesis hop size and a varying analysis hop size. Using the following recursive formulae:

$$\begin{cases} T_A(k) &= t_0 + R_S \sum_{j=1}^{k} \gamma(t_0 + lR_S) \\ T_S(k) &= t_0 + kR_S \end{cases} \quad (3)$$

with $T_A(k)$ the analysis time, $T_S(k)$ the synthesis time, R_S the synthesis hop size and γ the time-scaling ratio, we can compute the transformed sound duration. This means that we can warp the time-scaling ratio curve in order to preserved the sound length (synchronisation constraint):

$$T_S(i) = T_A(i) = t_s \quad (4)$$

Let $\mathcal{S}(\gamma, \gamma_0)$ be the modified value of $\gamma(t)$. The synchronisation constraint can be applied using one of these three methods:
– by addition of a constant value γ_a:

$$\mathcal{S}(\gamma, \gamma_a) = \gamma_a + \gamma$$

– by multiplication by a constant value γ_m:

$$\mathcal{S}(\gamma, \gamma_m) = \gamma_M \times \gamma$$

– by using a power law with a constant value γ_p:

$$\mathcal{S}(\gamma, \gamma_p) = \gamma^{\gamma_p}$$

For each method, we can decide wether to respect of not the initial variation interval I_γ given by the user. The synchronisation by addition without respect of the variation bounds has an analytical solution:

$$\gamma_a = 1 - \frac{\sum_{l=1}^{\mu} \gamma(l)}{\mu} \quad (5)$$

with the notation $\mu = \frac{t_s - t_0}{R_S}$ the number of grains (or iterations) used for time-scaling.

The synchronisation by multiplication witouth respect of the variation bounds has an analytical solution:

$$\gamma_m = \frac{\mu}{\sum_{l=1}^{\mu} \gamma(l)} \quad (6)$$

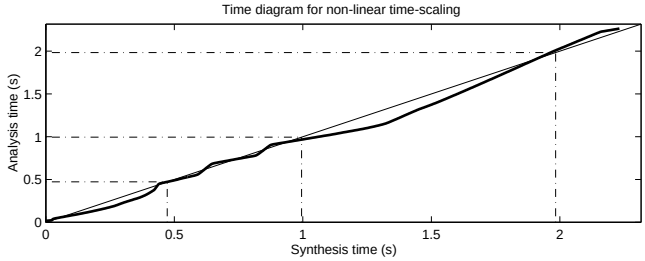

Figure 14: *Analysis/synthesis time diagram for the time-scaling ($y = x$ curve is plotted with a thin line). Dashed lines correspond to synchronization marks.*

Concerning the four other ways to respect the synchronisation constraint, they all require an optimisation scheme, since no analytical solution exist (except the evident solution $\gamma(t) = 1$, where no time-scaling is applied!). The optimisation scheme consists in minimizing the distance between the new sound length $\sum_{l=1}^{\mu} \mathcal{S}(\gamma, \gamma_0)$ and the synchronisation sound length μ. We compute this distance for several values of the synchronisation parameter γ_0 (γ_0 representing γ_a, γ_m or γ_p according to the kind of synchronisation chosen):

$$D = \left| \frac{\sum_{l=1}^{\mu} \mathcal{S}(\gamma, \gamma_0)}{\mu} - 1 \right| \quad (7)$$

The optimum we are looking for is the greatest of all γ_0 values that give a 0 distance, since the synchronisation constraint is respected ($D = 0$) and the transformation is the greatest (the greater γ_0, the greater the distance between the time-scaling ratio and 1,and so the greater the transformation).

Let $\gamma(t)$ vary in I_γ and $\mathcal{H}_I(f)$ be the truncation function of the parameter f, with the truncation interval I. Respecting the variation bounds is done by using $\mathcal{H}_{I_\gamma}(\mathcal{S}(\gamma, \gamma_0))$ instead of $\mathcal{S}(\gamma, \gamma_0)$ for the optimisation.

While applying synchronisation constraint at several places in the sound (given by an onset detector, for example), we can slightly or strongly change the sound rhythm. Using several sound feature, we can compute several time-scaling ratios that vary with time and that provide different rythmic changes.

Considering the fact that we have several time-scaling ratio curves with or without a synchronisation constraint, the gesture control can be applied in two ways: with or without respecting the synchronisation. While interpolating between the "synchronised" time-scaling ratio $\mathcal{S}(t)$ and 1, corresponding to the $y = x$ curve in the analysis/synthesis time diagram (*cf.* fig. 13), we ensure to always respect the constraint of synchronisation. We used a linear interpolation such as the one provided in the GRM tools to interpolate between two presets [14].

A second way to modify the curve with gesture data, is by addition or multiplication of $\mathcal{S}(\gamma, \gamma_0)$ by a value, thus not respecting anymore the synchronisation constraint. The addition or multiplication value is directly given by gesture data, after a fitting mapping. With a multiplicative value, we change the global duration and keep the local non-linearity behaviours. With an additive value, we change the global duration as well as the local non-linearity behaviours.

An improvement consists in the addition of a sine wave curve, which amplitude and frequency are given by gesture. A condition to ensure that the sound is not read backward is to use a low

amplitude sine wave (between 0 and 0.02 is fine is one wants to always read the sound forward, greater values allow nice forward and backward reading of a sound), as well as low frequency (lower than $10Hz$ is good). It is easy to prove that the addition of a sine wave to a time-scale ratio that respect the synchronisation constraint provides a new time-scaling ratio that still respect the synchronisation constraint if there is an integer number of periods between each set of two successive synchronisation points.

We presented many ways to automatically and gesturally modify the local and the global sound unfolding. They provide tools for changing rythm and expressivity changes of a sound (a spoken or sung voice as well as instrumental sounds, such as a jazz improvisation or a given melody). Gestural control can be seen as a second control level, over the adaptive control.

6. CONCLUSION

Special combinations of adaptive effects and gesture control give rise to a very powerful combination of a control directed by the sound itself and by one coming from a gestural device.

The gestural control by itself allows an interpretation of digital effects, and so is a basic strategy for musical applications: it is easier to discover what and where to apply some time-scaling. However one cannot do everything by hand. A new prosody cannot be given only by making a pitch curve on a tablet: it is really better to be guided with real curves, and to interpret them in real time for some of their features. As a matter of conclusion one can say that the alchemy of this combination of gestural and adaptive effects gives more than effects and gesture alone; it allows to structure sound transformations in a way that is reproducible, with or without variations given on the fly. And in fact this is where music comes in.

7. ACKNOWLEDGEMENTS

We greatly acknowledge the financial support of the CNRS (Centre National de la Recherche Scientifique) and the Conseil Régional Provence Alpes Côte d'Azur for the research grant of V. Verfaille; as well as the support of the Conseil Général des Bouches du Rhône concerning the research project "Le Geste Créatif en Informatique Musicale" (Creative Gesture in Computer Music). Many thanks to J.-M. Couturier for developing the interface used in sec. 4.

8. REFERENCES

[1] U. Zoelzer, Ed., *DAFX - Digital Audio Effects*, John Wiley & Sons, 2002.

[2] S. Roucos and A. Wilgus, "High quality time-scale modification for speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1986, pp. 493–6.

[3] E. Moulines and F. Charpentier, "Pitch synchronous waveform processing tehcniques fot text-to-speech synthesis using diphones," *Speech Communication*, vol. 9, no. 5/6, pp. 453–67, 1990.

[4] M. R. Portnoff, "Implementation of the digital phase vocoder using the fast fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 24(3), pp. 243–8, 1976.

[5] M. Dolson, "The phase vocoder: a tutorial," *Computer Music Journal*, 1986.

[6] X. Serra and J. O. Smith, "A sound decomposition system based on a deterministic plus residual model," *Journal of the Acoustic Society of America, Supp. 1*, vol. 89(1), pp. 425–434, 1990.

[7] A. Freed, X. Rodet, and Ph. Depalle, "Synthesis and control of hundreds of sinusoidal partials on a desktop computer without custom hardware," in *Proceedings of the International Conference on Signal Processing Applications & Technology (ICSPAT'92), San Jos*, 1992.

[8] M. Desainte-Catherine and S. Marchand, "Structured additive synthesis: Towards a model of sound timbre and electroacoustic music forms," *Proceedings of the International Computer Music Conference (ICMC'99)*, pp. 260–3, 1999.

[9] A. M. Noll, "Short-time spectrum and "cepstrum" techniques for vocal pitch detection," *J. Acoust. Soc. Am.*, vol. 36(2), pp. 296–302, 1964.

[10] J. A. Moorer, "The use of linear prediction of speech in computer music applications," *J. Audio Eng. Soc.*, vol. 27, no. 3, pp. 134–40, 1979.

[11] D. Arfib and N. Delprat, "Selective transformations of sound using time-frequency representations: An application to the vibrato modification.," in *104th Convention of the Audio Engineering Society, Amsterdam*, 1998.

[12] V. Verfaille and D. Arfib, "Adafx: Adaptive digital audio effects," in *Proceedings of the COST-G6 Workshop on Digital Audio Effects (DAFx-01), Limerick, Ireland*, December 2001.

[13] V. Verfaille and D. Arfib, "Implementation strategies for adaptive digital audio effects," in *Proceedings of the COST-G6 Workshop on Digital Audio Effects (DAFx-02), Hamburg, Germany*, September 2002.

[14] E. Favreau, "Phase vocoder applications in grm tools environment," in *Proceedings of the COST-G6 Workshop on Digital Audio Effects (DAFx-01), Limerick, Ireland*, 2001.