New phase-vocoder techniques for real-time pitch-shifting, chorusing, harmonizing and other exotic audio modifications

Jean Laroche and Mark Dolson
Joint E-mu/Creative Technology Center
1600 Green Hills Road
P.O.Box 660015
Scotts Valley, CA95067
Email: jeanl@emu.com markd@emu.com

March 3, 2000

Abstract

The phase-vocoder is a well-established tool for time-scaling and pitch shifting speech and audio signals. Its theory is now well understood and improvements have been proposed to reduce artifacts commonly encountered when time-expanding signals by large factors [1, 2, 3, 4]. In the literature, the phase-vocoder has been described primarily as a tool for time-scaling rather than pitch shifting, the latter usually being achieved by a combination of time-scaling and sampling rate conversion [5, 6]. This article focuses mainly on pitch-scale modification of speech and audio signals, and discusses the drawbacks of the standard time-scale/resampling technique. Two alternative techniques are presented that significantly reduce the complexity and computational cost, while offering dramatically extended capabilities. In particular, the new techniques, which operate solely in the frequency domain, enable chorusing, harmonizing and non-standard frequency modifications such as partial-stretching (non-linear frequency scaling), frequency inversions and so on.

0 Introduction

time-scale and pitch-scale modifications of speech and audio signals have now become a standard feature in many products ranging from sound-design software to telephone answering systems, musical effect processors, professional CD players, hard-disk recorders and so on.

As the computation power of DSP and general purpose processors keeps on increasing, more elaborate techniques such as the phase-vocoder, once confined to academic research or dedicated hardware, progressively make their way into inexpensive consumer products or PC-based software. Simultaneously, more research has been devoted to improving their performance and reducing their cost, see for example [1, 7, 3, 4] for improved phase-vocoder time-scaling techniques. The phase-vocoder has always been a highly praised technique for time-scale modifications of speech and audio signals, being free of the artifacts usually encountered in time-domain techniques [6, 5], such as warbling and tempo-modulation. The most standard way to carry out pitch-scale modifications in the phase-vocoder framework is to first perform time-scale modification, then perform time-domain sampling rate conversion on the resulting signal. For example, in order to raise the pitch of a signal by 2 while keeping its duration unchanged, one would use the phase-vocoder to time-expand the signal by a factor 2, leaving the pitch unchanged, and then upsample the resulting signal by a factor 2, thereby restoring the original duration while raising the pitch by a factor 2. In this paper, the standard technique is shown to have several drawbacks: it only enables linear pitch-scaling modifications (whereby all frequencies are multiplied by the same ratio) and the computational cost per output sample is a factor of the pitch-scaling modification ratio, potentially becoming increasingly larger for extreme modification factors. Two alternative techniques are presented that eliminate these drawbacks. In these techniques, the pitch-scale modification is carried out exclusively in the frequency domain, which enables multiple, non-linear frequency modifications (frequency-dependent modification ratios). The computational cost is shown to be independent of the type and amount of modification and is significantly less than in the standard technique, the arc tangent and phase-unwrapping stages having been eliminated. Furthermore, the simplest technique can operate at a low overlap ratio (50%), which further lowers its costs by a factor 2, compared to standard techniques which usually operate at a 75% overlap.

This paper is divided into two sections. The first section is devoted to the analysis of the standard phase-vocoder based pitch-scale modification algorithm and of its shortcomings. The second section presents two alternative techniques based on peak picking and local frequency shifting, and discusses their respective advantages.

1 Standard phase-vocoder pitch-scaling algorithm

1.1 phase-vocoder based time-scale modification

Because the most standard phase-vocoder based pitch-modification techniques involve a time-scaling stage, we devote this small part to the discussion of the most important aspects of phase-vocoder based time-scaling. However, since these techniques are well understood, we will not give a detailed description, but simply introduce our notations and emphasize a few important points. The reader is referred to [5] or [6] for descriptions of the most standard techniques, and [1, 2, 7] for improved techniques that reduce the artifacts in the modified signal.
In the following, we will denote $h(n)$ the analysis window, $w(n)$ the synthesis window, R_a the input hop size in samples (the number of samples separating two analysis windows) and R_s the output hop size in samples. The analysis short-term Fourier transform at frame u and frequency Ω_k is $X(t_a^u, \Omega_k)$ where t_a^u is the analysis time at frame u and $\Omega_k = \frac{2\pi k}{N}$ is the center frequency of the k-th vocoder "channel". N is the size of the discrete Fourier transform. The synthesis Fourier transform, after modification, is $Y(t_s^u, \Omega_k)$ where t_s^u denotes the synthesis time at frame u.
In the rest of this paper, the analysis window is supposed to be symmetrical around $n = 0$. In addition, the Fourier transform is supposed to be "centered at $n = 0$",

$$X(t_a^u, \Omega_k) = \sum_{n=-N/2}^{N/2} x(n + t_a^u) h(n) e^{-j\Omega_k n}$$

In practice though, Fast Fourier Transform algorithms introduce an implicit delay equal to half the size of the transform, in which case some of the results below become invalid because the delay introduces a linear phase term in the transform. An easy way to counter that is to circularly advance the signal by $N/2$ samples before taking the Fourier transform.

1.2 Using time-scale modification and resampling.

A pitch-scale modification by a factor β consists of multiplying the frequencies of all the components in the signal by β while keeping the signal's time-evolution unchanged. Thus, a modification by a factor $\beta = 2$ raises the pitch of a harmonic signal by an octave and a modification by a factor $\beta = .5$ lowers the pitch by an octave.
The standard technique for performing a factor β pitch-scale modification using the phase-vocoder relies on a two-stage process:

1. In the first stage, the phase-vocoder is used to perform a time-scale modification of the signal, i.e., to modify its duration by a factor β while keeping its frequency content unchanged. Here, $\beta > 1$ means the duration of the signal is increased.

2. In the second stage, the time-scaled signal is resampled at a new sampling rate equal to the original sampling rate divided by β. At the end of this stage, the duration of the resulting signal is equal to that of the time-scaled signal divided by β which is equal to the duration of the original signal. The resampling stage has the effect of modifying the frequency content of the signal, which is the desired effect.

Note that the order in which these two stages are carried out can be reversed, but the cost of the resulting algorithm will not be the same: For upward pitch-shifts ($\beta > 1$), it is less costly to first perform the sampling rate conversion stage, and then apply the time-scaling stage on the result. For downward pitch-shifts ($\beta < 1$), time-scaling should be performed first, followed by the resampling stage. The reason is as follows: The cost of the resampling stage is proportional to the number of samples at the output of this stage. The cost of the time-scaling stage is also proportional to the number of output samples. To generate L samples of pitch-shifted samples, performing the sampling rate conversion first requires calculating L/β intermediate samples, then L time-scaled samples. The overall cost *per output pitch-modified sample* can be approximately expressed as

$$C_1 = \frac{1}{\beta} C_r + C_{ts} \quad \text{(resampling first)} \tag{1}$$

where C_{ts} is the cost per output sample of the time-scaling stage, and C_r that of the resampling stage. This expression clearly overlooks the algorithm overhead which strongly depend on the specific implementation, but is still useful for comparisons.
Carrying out the time-scaling stage first requires calculating $L\beta$ time-scaled samples, then L resampled sam-

ples, yielding an overall cost per output sample of

$$C_2 = \beta C_{ts} + C_r \quad \text{(time-scaling first)} \tag{2}$$

It is clear from inspecting Eqs. (1) and (2) that if $\beta > 1$ then the first option is less costly, while the converse is true if $\beta < 1$. In situations where the pitch-scale modification factor β is constrained to be larger or smaller than 1 it is practical to choose the most favorable option. However, if β can vary above or below 1 during processing, the on-the-fly reversal of the order in which the resampling and time-scaling stages are carried out would undoubtedly generate significant overhead which could offset the cost-saving. If the stages are carried out in a fixed order, regardless of the modification factor, then one of Eqs. (1) and (2) applies to all values of β, which means that the algorithm becomes more and more costly as β grows larger and larger, or smaller and smaller, depending on the order chosen.
Note that if resampling is performed first, the size of the analysis window and the hop size in the time-scaling stage should be modified according to the modification factor β, to reflect a constant duration in seconds. Since the size of the FFT can also be scaled accordingly, and because the cost of an overlap/add process is roughly independent from the FFT size for a constant overlap in percent, this rescaling does not necessarily increase the cost of the algorithm. In practice though, the FFT size which must be a power of two cannot be rescaled arbitrarily, and the cost of the algorithm becomes slightly larger for large β.

The resampling stage (or interpolating stage), can be performed in the time-domain by use of any of the many resampling schemes. An inexpensive, low-quality technique consists of using linear interpolation. Higher quality can be obtained by use of higher order Lagrange interpolation, or by the Smith-Gosset algorithm [8], with a significant increase of computation cost.
Alternately, the resampling stage can be performed in the frequency domain, and can be combined with the time-scaling stage, as described in [5].

1.3 Drawbacks of the standard technique

The standard pitch-scale modification technique, which combines time-scale modification and resampling, has several drawbacks. The main drawback is that the cost per output sample is a function of the modification factor, as shown by Eqs. (1) and (2). While this can be an attractive feature, for example if the most favorable processing order can be selected as a function of the modification factor, it can also be a nuisance if the processing order is fixed, since in that case, the algorithm can become more and more costly depending on the value of β. An algorithm with a cost independent of the modification factor is generally preferable.
Another drawbacks of the standard technique is that only *one* "linear" pitch-scale modification is allowed, i.e., the frequencies of all the components are multiplied by the same factor β. In particular, signal "harmonizing" (multiple, superimposed pitch-scale modifications with different factors) cannot be implemented in one pass, and requires repeated processing. An algorithm that would be more flexible in terms of how frequencies are altered could provide a much wider range of creative modifications to the user. The techniques described below allow such flexible modifications.

2 Peak-based pitch effects in the phase-vocoder

The underlying idea behind the algorithms described in this section is the following: The short-term Fourier transform of a single input complex exponential with a constant frequency is a peak located around the exponential's frequency. The spectral shape of the peak is the signature of the temporal shape of the analysis window. If instead of rescaling the frequency axis (which changes the peak frequency but also alters the spectral shape of the peak) as described in [5], the frequency bins around the peak are *shifted* to a new frequency, then this new spectral shape corresponds to the original analysis window modulated at the new frequency. More specifically, if the original signal is

$$x(n) = e^{j(\omega n + \phi)}$$

the short-term signal at time t_a^u is

$$x^u(n) = e^{j(\omega(n+t_a^u)+\phi)} h(n)$$

and its Fourier transform is,

$$X(t_a^u, \Omega_k) = e^{j(\phi + \omega t_a^u)} H(\Omega_k - \omega)$$

where $H(\Omega)$ is the Fourier transform of the analysis window $h(n)$. If we now shift this Fourier transform by a frequency $\Delta\omega$, i.e.,

$$Y(t_a^u, \Omega_k) = X(t_a^u, \Omega_k - \Delta\omega)$$

then the corresponding short-term signal is simply

$$y^u(n) = x^u(n) e^{j\Delta\omega n} = e^{j(\phi+\omega t_a^u)} e^{j(\omega+\Delta\omega)n} h(n)$$

which shows that $y^u(n)$ corresponds to the same analysis window modulated at a different frequency $(\omega + \Delta\omega)$ with an additional phase term.
The important point here, is that because the spectrum has not been rescaled, but merely shifted in frequency, the short-term signal is not shrunk or expanded in

time, and its duration is still that of the analysis window $h(n)$, which makes it possible to use the same input and output hop size. It is shown in Appendix A that if the input signal is a complex exponential,

$$x(n) = e^{j(\omega n+\phi)}$$

if in the phase-vocoder we set $t_a^u = t_s^u = uR_0$, and

$$Y(t_s^u, \Omega_k) = X(t_a^u, \Omega_k - \Delta\omega)e^{j\Delta\omega uR_0} \quad (3)$$

and if the standard phase-vocoder condition for perfect-reconstruction is met:

$$\sum_{i=-\infty}^{\infty} g(n+iR_0)h(n+iR_0) = 1 \quad \forall n \quad (4)$$

then the output signal is a perfectly frequency-shifted complex exponential:

$$y(n) = e^{j[(\omega+\Delta\omega)n+\phi]}$$

In order to accommodate time-varying frequencies ω^u and shifts $\Delta\omega^u$, Eq. (3) can be replaced by

$$\begin{aligned} &Y(t_s^u, \Omega_k) = X(t_a^u, \Omega_k - \Delta\omega^u)e^{j\theta^u} \text{ with} \\ &\theta^u = \theta^{u-1} + \Delta\omega^u R_0 \end{aligned} \quad (5)$$

which is equivalent to Eq. (3) for a constant frequency shift $\Delta\omega^u = \Delta\omega$.

This suggests a very simple algorithm for pitch-scale modification: In each phase-vocoder frame, 1) find the peaks in the spectrum (which are assumed to indicate underlying sinusoids), and 2) for each peak, shift the bins around it to a new frequency, taking into account the phase correction term in Eq. (5). Notice that $\Delta\omega$ does not necessarily correspond to an integer number of frequency bins, so Eq. (5) might require interpolation, since $X(t_a^u, \Omega_k)$ is only known at discrete frequencies Ω_k.

The successive stages of this algorithm are described in more detail in the following sections, but the decisive advantages of this technique over the standard one are already clear: 1) There is a great flexibility as to how each sinusoidal component will be altered, since $\Delta\omega$ can be a function of the channel index k, 2) the input and output hop size, and the FFT size do not need to be a function of the pitch-scale amount, 3) the phase-correction term in Eq. (5) does not require the knowledge of the instantaneous frequency ω, and therefore, no arctangent or phase-unwrapping is needed. In addition, we will see that in some cases, an overlap as low as 50% can be used, with no loss of quality.

2.1 Peak-detection

The peak detection can be fairly coarse, since all we need to do is detect the main sinusoids in the signal. In the simplest implementation, a channel whose amplitude is larger than its four nearest neighbors is said to be a peak; this criterion is both simple and cost-effective, but might fail to distinguish local maxima due to the presence of a sinusoid from local maxima corresponding to the lobes of the Fourier transform of analysis window. Any more refined scheme can be used instead, although it was found that in practice, this very basic technique yields very good results.

Following this stage, the frequency axis can be split into "regions of influence" located around each peak. Again, a very simple scheme is to cut the frequency axis half-way in between two consecutive peaks, assigning each half to the closest peak. Another reasonable scheme is to look for the channel with the lowest amplitude between two consecutive peaks, and make this channel the limit between the two regions of influence.

2.2 Calculating the amount of frequency shift

A very nice feature of this algorithm is that every peak can be shifted to an arbitrary frequency. If the desired effect is a standard pitch-scaling with a ratio β, then the peak should be shifted by $\Delta\omega = \beta\omega - \omega$ where ω is the frequency of the sinusoid responsible for the peak. However, only an approximate value of ω is known, namely Ω_{k_0}, where k_0 is the peak channel, and therefore $\Delta\omega$ is only known approximately, which can be a problem. In practice, if the FFT size is large enough, then Ω_{k_0} might end up being a good enough estimate of ω. If this is not the case, for example if a very precise amount of pitch shifting is desirable, then the estimate of ω can be refined by use of a quadratic interpolation, whereby a parabola is fitted to the peak channel and its two neighbors and the maximum of the parabola is taken to indicate the true sinusoidal frequency. This is known to yield the exact frequency for a pure sinusoid and a Gaussian analysis window if the transform is expressed in dB.

Linear frequency-scaling is not the only possibility at this point. One-step "Harmonizing" becomes possible, because one can shift a given peak to different locations, as determined by the harmonizing ratios: For example, to harmonize a melody to a fourth and a seventh, each peak could be shifted to two frequencies, one corresponding to the ratio $2^{5/12}$, the other to the ratio $2^{10/12}$. Chorusing can also be obtained by harmonizing with ratios close to 1. With the standard technique, these effects would require multiple passes, making them prohibitively expensive. Other interesting effects can be obtained by using a ratio β itself a factor of frequency. For example, setting $\beta(\omega) = \beta_0 + \gamma\omega$ turns a harmonic signal (one where the frequencies of all the partials are integer multiples of a fundamental frequency) into an inharmonic signal, or an inharmonic

signal into a harmonic signal. Another possibility consists of shuffling the frequencies around, completely altering the spectral content of the signal. Interestingly, our technique makes it possible to apply the same frequency manipulations allowed by sinusoidal representations [9, 10, 11, 12, 13], *in real-time*, and without the hassle of the preliminary analysis stage.

2.3 Shifting the peaks

Once the amount of frequency shift $\Delta\omega$ is known, two separate cases arise depending on whether $\Delta\omega$ does or does not correspond to an integer number of frequency channels.

Integer shifts When $\Delta\omega$ corresponds to an integer number of channels, the shift does not require any interpolation, and is just a matter of copying the values of the Fourier transform from one set of channels to another. This is by far the simplest case. Notice however, that two consecutive regions of influence, after being shifted, can overlap (in which case they can simply be added) or become disjoint (in which case null spectral values can be inserted) as shown in Fig. 1.

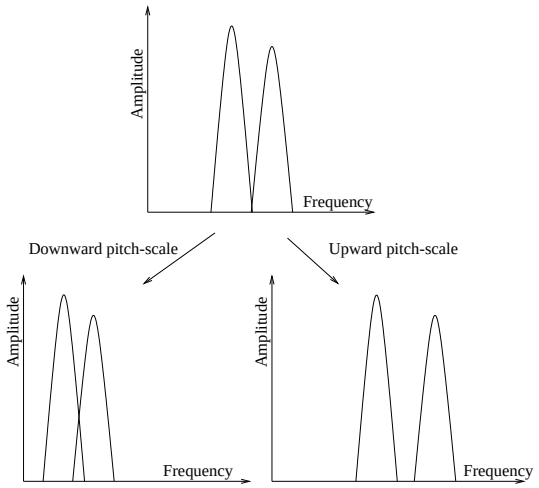

Figure 1: Shifting peaks in frequency: downward pitch-scaling can cause the areas around peaks to overlap (left) and upward pitch-scaling can cause them to become disjoint (right)

Non-integer shifts When $\Delta\omega$ corresponds to a non-integer number of channels, Eq. (5) requires interpolating the spectrum between discrete frequency values. The operation is essentially a frequency-domain fractional delay. A lot has been written about fractional time-delays [14, 15, 16, 17, 18], and the same techniques can be applied for fractional frequency-delays. The simplest technique consists of using linear interpolation (both the real and the imaginary part are linearly interpolated), which in the dual domain (here, in the time-domain), can introduce an undesirable modulation. In the worst case of a .5 frequency bin shift, the linear interpolation introduces an attenuation at the beginning and end of the short-term signal. Specifically, the half-channel shifted version of $X(t_a^u, \Omega_k)$ is given by $Y(t_a^u, \Omega_k) = 0.5(X(t_a^u, \Omega_k) + X(t_a^u, \Omega_{k+1}))$, which yields

$$y^u(n) = x^u(n) \cos \pi \frac{n}{N} \quad -\frac{N}{2} \leq n < \frac{N}{2}$$

where N denotes the size of the FFT. The short-term signal is *amplitude-modulated* by a cosine function. Assuming that the analysis and synthesis windows were designed for perfect reconstruction (i.e., satisfied Eq. (4)), then the output signal $y(n)$ will also exhibit amplitude modulation. This is illustrated

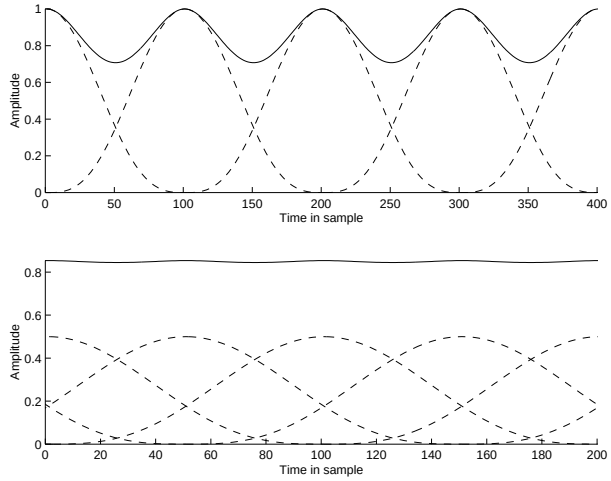

Figure 2: Amplitude modulation caused by the frequency domain linear interpolation for a half-bin shift. Top corresponds to a 50% overlap, bottom to a 75% overlap, for a Hanning input window and a rectangular synthesis window. The dashed lines are the individual cosine modulated output windows $h(n)g(n)$ and the solid line is the resulting overlap-add modulation.

in Fig. 2 for a 50% and a 75% overlap add, a Hanning analysis window, a rectangular synthesis window and a FFT size equal to the size of the windows. It is shown in appendix B that the modulation introduces sidebands whose levels are a function of the window type and of the overlap. For an input sinusoid, at a 50% overlap, the sidebands are about 21dB down from the sinusoid's amplitude, which is quite audible. As a result, a 50% overlap *cannot be used* if linear interpolation is to be used.

At a 75% overlap, however, the amount of amplitude modulation drops significantly: The sidebands are about 51dB down from the sinusoid's amplitude. This level of modulation is much less conspicuous, if audible at all. Fig. 3 shows the modulation in the frequency domain for a sinusoid with a normalized frequency equal to .04, at 50% and 75% overlaps.

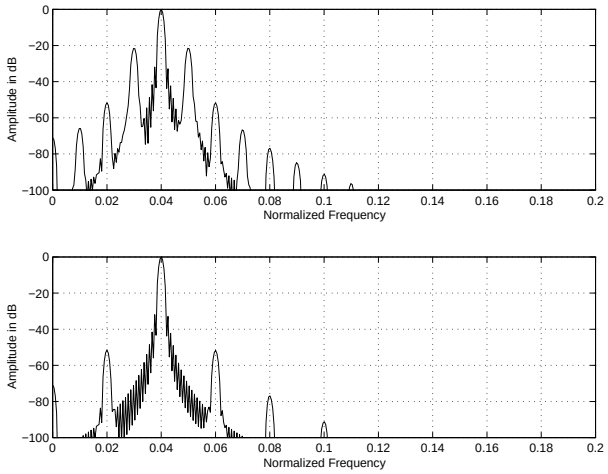

Figure 3: Amplitude modulation caused by the frequency domain linear interpolation for a half-bin shift, seen in the frequency domain for an input sinusoid of normalized frequency 0.04. Top corresponds to a 50% overlap, bottom to a 75% overlap, for a Hanning input window and a rectangular synthesis window.

In order to still be able to use a 50% overlap, one could use a FFT size larger than the analysis window length, or a higher-quality interpolation scheme, such as allpass or high-order Lagrange interpolation. Note, however, that the additional cost of these techniques might very well offset the savings brought by the use of a 50% overlap instead of a 75%, depending on the number of peaks in the spectrum.

In conclusion, if only integer frequency shifts are allowed, then no interpolation is needed, and a 50% overlap can be used. If non-integer frequency shifts are a requirement, then linear interpolation with a 75% overlap yields good results.

2.4 Adjusting the phases

As shown in Eq. (5) the phases of the shifted bins need to be adjusted for the output short-term signals to overlap coherently. Eq. (5) only requires the calculation of a pair of cosine and sine per peak, and one complex multiplication per vocoder channel around the peak. This is significantly simpler than the standard technique, which requires one arc tangent and one phase-unwrapping per vocoder channel, in addition to the cosine, sine and complex multiplication.

When only frequency shifts corresponding to integer numbers of channels are allowed, and when the hop size is a submultiple of the FFT size, Eq. (5) becomes even simpler: In that case, $\Delta\omega^u = 2\pi n/N$ where N is the FFT size, and n is an integer, and $R_0 = N/m$ with m integer. As a result, $\Delta\omega^u R_0 = n2\pi/m$, i.e., is always a multiple of $2\pi/m$. For example, if the overlap is 50%, then $m = 2$ and $\Delta\omega^u R_0$ is always a multiple of π, and so is θ^u (provided $\theta^0 = 0$). As a result, no cosine or sine calculation is necessary and the rotation is a simple change of sign! This remark makes the algorithm based on integer-channel shifts even more attractive.

In both cases, it is useful to note that because the channels around a given peak are rotated by the same angle θ^u, the differences between the phases of the channels around a peak in the input short-term Fourier transform are preserved in the output short-term Fourier transform. This is similar to the phase-locking scheme referred to as "Identity Phase-Locking" in reference [2] which was shown to dramatically minimize the "phasiness" artifact often encountered in phase-vocoder time or pitch-scale modifications.

3 Discussion and conclusion

The two techniques presented above exhibit several desirable properties, when compared to the standard time-scale/resampling scheme for pitch-scale modifications of audio signals. Their cost is independent of the modification factor, they allow for a much wider range of modifications and they are significantly simpler and more cost-effective. When large FFT sizes are used (50 to 60 ms), for example for input signals that contain low-pitch signals with closely spaced harmonics, the simplest pitch-shifting scheme can be used. This scheme, which restricts the frequency shifts to be integer numbers of channels, only requires a 50% overlap, does not call for spectral interpolation or angle calculation, and uses a trivial phase-adjusting equation. As a result, its cost is roughly that of the unavoidable Fourier transform and the overlap add stages. Its drawback is that the pitch-scale factor can only be approximate, with the approximation getting better as the FFT size increases, and that the frequency ratios between the partials in the signal are only approximately preserved.

Large FFT sizes are undesirable however, as they tend to worsen the "phasing" artifacts often noticeable in modified signals [2, 1]. When using small FFT sizes (10 to 20ms), the simplest scheme no longer yields good results, and spectral interpolation has to be used. This in turns requires a 75% overlap to minimize amplitude modulation problems, and a slightly more complex phase-adjustment stage. The overall algorithm remains far less complex than the standard phase-vocoder technique. It can be easily implemented on a PC, performing real-time pitch-scale modification of a mono 48kHz file, utilizing less than 40% of the cpu power of a pentium-pro running at 200MHz (the simplest algorithm would utilize less than 20%).

The new algorithms also allow for much more flexible frequency manipulations: In particular, harmonizing and chorusing can be implemented in one pass at a

marginal additional cost, whereas such effects would require multiple passes with the standard algorithm. More exotic effects, akin to those allowed by sinusoidal models, are also very simple to implement.
Finally, both algorithms implicitly implement the "Identity Phase-Locking" technique described in [2], and therefore produce much higher-quality modifications than standard, non phase-locked algorithms.

Arguably, the two techniques presented above bear some similarities with techniques use in sinusoidal modeling, [19], coding/modification [20] or synthesis [21], as they manipulate spectral peaks caused by the presence of "sinusoid-like" signals. One significant difference is that the actual shape of the spectral peak (in both amplitude and phase) is not synthesized in our algorithms, but merely modified. By contrast, in sinusoidal modeling, coding or synthesis, one would *replace* the spectral peaks by those corresponding to the underlying sinusoids, based on their estimated amplitudes phases and frequencies. An important consequence of this remark is that in the absence of modification (no frequency-shifting) our algorithms produce a signal strictly identical to the original one, whereas sinusoidal modeling does not.

A Perfect frequency shift of a complex exponential

In this section, we show that for a complex input exponential

$$x(n) = e^{j(\omega n+\phi)}$$

using $R_a = R_s = R_0$ and setting

$$Y(t_s^u, \Omega) = X(t_a^u, \Omega - \Delta\omega)e^{j\Delta\omega u R_0} \tag{6}$$

yields a perfect frequency shifted complex exponential, provided the analysis and synthesis windows satisfy a condition for perfect reconstruction.
At time $t_a^u = uR_0$ the short-term input signal is

$$x_a^u(n) = x(n + uR_0)h(n)$$

where $h(n)$ is the analysis window. This yields:

$$X(t_a^u, \Omega) = e^{j(\omega u R_0+\phi)}H(\Omega - \omega)$$

in which $H(\Omega)$ is the Fourier transform of the analysis window. As a result,

$$\begin{aligned} Y(t_s^u, \Omega) &= X(t_a^u, \Omega - \Delta\omega)e^{j\Delta\omega u R_0} \\ &= e^{j(\omega u R_0+\phi+\Delta\omega u R_0)}H(\Omega - \omega - \Delta\omega) \\ &= e^{j((\omega+\Delta\omega) u R_0+\phi)}H(\Omega - \omega - \Delta\omega) \end{aligned}$$

and the corresponding short-term output signal is

$$y^u(n) = e^{j((\omega+\Delta\omega) u R_0+\phi)}e^{j(\omega+\Delta\omega)n}h(n)$$

Now the output signal is obtained by overlap-adding the short-term signals in the following way:

$$y(n) = \sum_{u=-\infty}^{\infty} y^u(n - uR_0)g(n - uR_0)$$

and this yields

$$\begin{aligned} y(n) = \sum_{u=-\infty}^{\infty} e^{j((\omega+\Delta\omega) u R_0+\phi)}e^{j(\omega+\Delta\omega)(n-uR_0)} \\ h(n - uR_0)g(n - uR_0) \end{aligned}$$

which simplifies as

$$y(n) = e^{j((\omega+\Delta\omega)n+\phi)} \sum_{u=-\infty}^{\infty} h(n - uR_0)g(n - uR_0)$$

If the standard phase-vocoder condition for perfect-reconstruction is met:

$$\sum_{i=-\infty}^{\infty} g(n + iR_0)h(n + iR_0) = 1 \quad \forall n$$

then the output signal is a perfectly frequency-shifted complex exponential:

$$y(n) = e^{j(\omega+\Delta\omega)n+\phi}$$

which completes the proof.

B Overlap-add modulation

In this section, we investigate the time-domain amplitude modulation caused by overlap-adding cosine-modulated windows. Given a window $w(n)$ and an overlap hop R_0, the signal resulting from overlap-adding $w(n)$ every R_0 samples can be written as:

$$\begin{aligned} w_o(n) &= \sum_{i=-\infty}^{\infty} w(n - iR_0) \\ &= w(n) \star \sum_{i=-\infty}^{\infty} \delta(n - iR_0) \end{aligned} \tag{7}$$

where $\delta(n)$ is the dirac function (zero except for $n = 0$ where it is 1) and $\star$ denotes time-domain convolution. Using a standard result on the Fourier transform of a series of dirac functions, we can express the Fourier transform of $w_o(n)$ as:

$$W_o(\Omega) = W(\Omega)\frac{1}{R_0} \sum_{i=-\infty}^{\infty} \delta(\Omega - i\frac{2\pi}{R_0})$$

which shows that $w(n)$ has a line spectrum and the amplitude of each line is obtained by *sampling* $W(\Omega)$ at frequencies $\frac{2i\pi}{R_0}$. If a sinusoid of frequency ω_0 is amplitude-modulated by $w(n)$, then the spectrum of the resulting signal will be $W(\Omega - \omega_0)$, i.e., $W(\Omega)$ shifted to the frequency of the sinusoid, as shown in Fig. 3

For the overlap-add process to introduce no amplitude modulation, $W_o(\Omega)$ must be equal to $\delta(\Omega)$ ($w_o(n) = 1$) and equivalently, $W(i\frac{2\pi}{R_0})$ must vanish for any i integer different than 0.

Let us first examine the case where no cosine-modulation is introduced during the peak-shifting process. For a Hanning analysis window of length N, and a rectangular synthesis window, $w(n)$ is a length-N Hanning window and it is well known that the Fourier transform of such a window is zero for $\Omega = 2k\pi/N$ with $k \geq 2$ integer [22]. Consequently, any hop size $R_0 = N/k$ will yield a constant overlap-add signal, because $W(i\frac{2\pi}{R_0}) = W(i\frac{2k\pi}{N}) = 0$ except for $i = 0$. We just restated the well-known fact that Hanning windows overlap-add to unity, provided the hop size is equal to the window length divided by any integer larger than 1.

When cosine modulation is introduced, the Fourier transform of $w(n)$ is no longer the same. In the same case as above, but with a cosine modulation of the window, the Fourier transform of $w(n)$ no longer vanishes at $\Omega = 2k\pi/N$, as shown in figure 4. In the 50% overlap case, the first modulation component, at frequency $4\pi/N$, has a null amplitude for the Hanning window (perfect overlap) but a -21dB amplitude for the cosine-modulated Hanning window. For a 66% overlap, the first modulation component, at frequency $6\pi/N$, has a -41dB level, and for a 75% overlap, the first modulation component, at frequency $8\pi/N$, has a -51dB level.

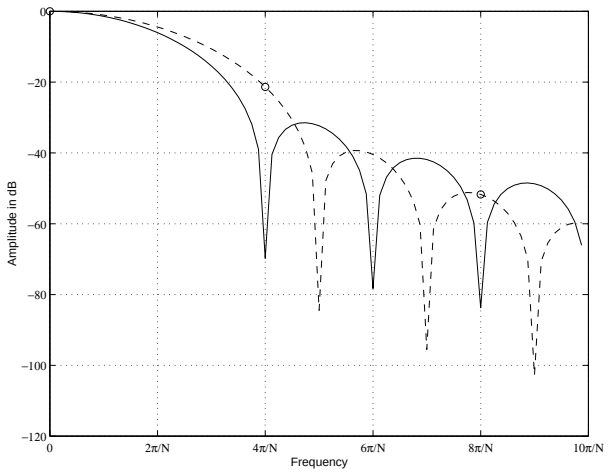

Figure 4: Fourier transform of the Hanning window (solid line) and of the cosine modulated Hanning window (dashed line). For a 50% overlap, the modulation has sinusoidal components at multiples of $4\pi/N$ with a null level for the Hanning window, and levels decreasing from -21dB for the modulated Hanning. At 75% overlap, for the modulated Hanning window, the largest component other than DC appears at $8\pi/N$ at a -51dB level.

References

[1] M.S. Puckette. Phase-locked vocoder. In *Proc. IEEE ASSP Workshop on app. of sig. proc. to audio and acous.*, New Paltz, NY, 1995.

[2] J. Laroche and M. Dolson. Improved phase vocoder time-scale modification of audio. *IEEE Trans. Speech and Audio Processing*, 7(3):323–332, May 1999.

[3] J. Laroche and M. Dolson. Phase-vocoder: About this phasiness business. In *Proc. IEEE ASSP Workshop on app. of sig. proc. to audio and acous.*, New Paltz, NY, 1997.

[4] J. Laroche and M. Dolson. About this phasiness business. In *Proc. Intern. computer music conf.*, Thessaloniki, 1997.

[5] E. Moulines and J. Laroche. Non parametric techniques for pitch-scale and time-scale modification of speech. *Speech Communication*, 16:175–205, Feb 1995.

[6] J. Laroche. Time and pitch scale modification of audio signals. In M. Kahrs and K. Brandenburg, editors, *Applications of Digital Signal Processing to Audio and Acoustics*. Kluwer, Norwell, MA, 1998.

[7] H.J.S. Ferreira. An odd-DFT based approach to time-scale expansion of audio signals. *IEEE Trans. Speech and Audio Processing*, 7(4):441–453, Jul 1999.

[8] J. Smith and P. Gossett. A flexible sampling-rate conversion method. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Diego, CA, Mar 1984.

[9] J.S. Marques, L.B. Almeida, and J.M. Tribolet. Harmonic coding at 4.8 kb/s. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, volume 1, pages 17–20, 1990.

[10] R. J. McAulay and T. F. Quatieri. Speech analysis/synthesis based on a sinusoidal representation. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-34(4):744–754, Aug 1986.

[11] X. Serra and J. Smith. Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music J.*, 14(4):12–24, Winter 1990.

[12] E. B. George and M. J. T. Smith. Analysis-by-synthesis/Overlap-add sinusoidal modeling applied to the analysis and synthesis of musical tones. *J. Audio Eng. Soc.*, 40(6):497–516, 1992.

[13] M. Kahrs and K. Brandenburg. *Applications of Digital Signal Processing to Audio and Acoustics.* Kluwer, Norwell, MA, 1998.

[14] V. Valimaki and T.I. Laakso. Fractional delay digital filters. In *Proc. IEEE Int. Symposium on Circuits and Systems*, Chicago, IL, 1993.

[15] T.I. Laakso, V. Valimaki, M. Karjalainen, and U. Klaine. Splitting the unit delay [fir/all pass filters design]. *IEEE Signal Processing mag.*, 13(1):30–60, Jan 1996.

[16] S. Tassart and P. Depalle. Analytical approximations of fractional delays: Lagrange interpolators and allpass filters. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Munich, Germany, 1997.

[17] W. Putnam and J. Smith. Design of fractional delay filters using convex optimization. In *Proc. IEEE ASSP Workshop on app. of sig. proc. to audio and acous.*, New Paltz, NY, 1997.

[18] R.C. Williamson P.J. Kootsookos. Fir approximation of fractional sample delay systems. *IEEE Trans. Circuit and Syst. II*, 43(3):269–271, Mar 1996.

[19] T. F. Quatieri and R. J. McAulay. Speech transformations based on a sinusoidal representation. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, volume 2, pages 489–492, 1985.

[20] S.N. Levine, T.S. Verma, and J.O. Smith. Multiresolution sinusoidal modeling for wideband audio with modifications. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, 1988.

[21] M. Goodwin and X. Rodet. Efficient Fourier synthesis of nonstationary sinusoids. In *Proc. Intern. computer music conf.*, San Francisco, CA, 1994.

[22] A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing.* Prentice Hall, Englewood Cliffs, New Jersey, 1975.