LOW LATENCY AUDIO PITCH SHIFTING IN THE FREQUENCY DOMAIN

Nicolas Juillerat, Béat Hirsbrunner

University of Fribourg, Switzerland

ABSTRACT

This paper presents a low latency pitch shifting algorithm based on the Short-Time Fourier Transform (STFT). Unlike existing STFT-based implementations of pitch shifting, the presented algorithm is more robust to reductions of the Fourier transform size. As a result, it achieves latencies as low as 12ms and still produces good quality, whereas other algorithms are performing much worse with similar low latency constraints. The presented algorithm also provides an alternate way of mitigating the well-known phasiness problem of the phase vocoder.

***Index Terms*—** Audio, Pitch shifting, Low latency, Fourier transform

1. INTRODUCTION

Pitch shifting is a digital audio effect whose purpose is to change the pitch of a sound or music without altering its speed. This is performed by scaling all the frequencies by a constant factor. Various implementations have been proposed, mainly based on the phase vocoder [27] and on synchronous overlap-add (SOLA) [9, 24]. The former are working in the frequency domain, using the short time Fourier transform (STFT), while the later are working in the time domain.

The primary aim of this paper is to investigate the problem of latency. In a real-time system, the latency of the phase vocoder is determined by the size of the Fourier transform. Typical values are between 2048 and 8192 at 44.1kHz, corresponding to about 46 to 186ms. The latency of SOLA techniques depends on the sizes of the considered blocks, and they typically have similar latencies [28]. While further reducing the Fourier transform or block size reduces the latency, it also reduces the quality of the result down to unacceptable levels.

In this paper, an algorithm working in the frequency domain (using the STFT) is presented that shows to be highly resistant to reductions of the Fourier transform size. Hence it can achieve latencies of less than 12ms while providing a quality significantly higher than other approaches. This makes it a good candidate for live situations in which low latency constraints are imposed [10].

Section 2 gives an overview of related work. Section 3 presents the new algorithm step by step. It is compared to other approaches in Section 4 by means of experimental observations of spectral properties and artifacts. Section 5 proposes a possible explanation of the experimental results. Section 6 discusses future works and then Section 7 gives conclusions.

2. RELATED WORK

Various authors have investigated latency issues regarding digital audio effects in general [3, 5, 15]. An attempt to provide a low latency phase vocoder has resulted in a promising solution [4]. Yet it requires parallel hardware, and only divides the latency of existing phase vocoder implementations by 3, possibly getting down to 16ms. The presented solution achieves even lower latencies without requiring special hardware.

Another proposal working in the time domain has recently been proposed by the authors [1] and closely compares with the present work. The present algorithm differs in that it works in the frequency domain rather than in the time domain, and hence has a completely different implementation. It is also more efficient in term of required computing power and even runs on modest computers. On the other hand, the previous algorithm [1] can achieve slightly lower latencies on average, with the drawback of having a frequency dependent latency (low frequencies are delayed compared to high frequencies, turning impulses into chirps).

Pitch shifting is known to be a difficult problem, and most algorithms suffer from various artifacts. Time domain approaches typically have transient duplication, warbling or tempo modulation artifacts [9, 24] whereas frequency domain approaches have phasiness and transient smearing problems [16, 17, 19]. These problems are reduced by the use of more advanced and high quality approaches. Yet, these techniques are resorting either to hybrid algorithms involving transient detection or to multi-resolution analyses [7, 8, 14, 18], and are thus much more complex than the algorithm presented in this paper. They are also introducing an unacceptable amount of latency and are hence not suitable for live performances.

3. OVERVIEW OF THE ALGORITHM

This section gives a brief overview of the proposed new algorithm. It then discusses its expected behavior regarding phasiness and detuning. Finally, it discusses how the algorithm handles specific modulation artifacts.

3.1. Existing Approaches

The traditional phase vocoder works in two steps. In a first step, frequency components of the sound are estimated as accurately as possible (*analysis*), using either interpolation between the bins of the STFT frames, or using the phase derivatives from one STFT frame to another. In a second step, the detected frequencies are scaled and *synthesized* back. Because the frequencies are scaled, their phases are no longer coherent from one STFT frame to the next. Hence a specific phase propagation process is required to ensure smooth transitions between the frames [19, 27].

This phase propagation process is the main cause of phasiness. Previous improvements have attempted to reduce it by keeping the phase relationships between frequencies related to the same sound. This notion of "sound" is subjective and different implementations have been proposed depending on its interpretation: *Loose* phase locking only considers a few adjacent frequency bins, as they are expected to belong to the same original frequency as a result of the windowing process [21]. *Rigid* phase locking consider the entire "region of influence" of individual peaks of the spectrum [16]. Various other variations have been proposed. A paper has formalized these schemes as attempts to preserve the *vertical phase coherence* among frequency bins of the same frame [17].

3.2. Basic Idea of the Algorithm

The proposed algorithm also preserves vertical phase coherence. However, unlike the phase vocoder approaches, it does not rely on an accurate signal analysis. Instead, the analysis-synthesis process of the STFT is reduced to a simple scaling.

Let denote by Ω_x the complex value (named phasor in the reminder of this paper) of the DFT bin number x of a given STFT frame. In a first step, every phasor Ω_a is simply copied to phasor Ω_b. The bin number b of the scaled phasor is computed using *only* the bin number a of the original phasor, as follows:

$$b := \lfloor ka + 0.5 \rfloor \tag{1}$$

Here k is the desired scale factor. In short, everything is just scaled and rounded to the nearest DFT bin number. In a second step, the phase of each resulting phasor Ω_b is modified by multiplying it by a root of unity as specified by equation (2), where p is the frame number of the STFT, N is the DFT size, and O is the number of overlapped frames in the STFT[1].

$$\Omega_b := \Omega_b e^{-i \frac{(b-a)p}{O} \frac{2\pi}{N}} \tag{2}$$

If ϕ is the phase of a pure and steady tone centered in bin a on the first frame ($p = 0$), its phase on the frame p is given by $\phi + 2\pi \frac{ap}{O}$. Hence, by additionally taking the move from bin a to bin b into account, equation (2) is a correct phase propagation process. Its simplicity compared to previous approaches [16, 19] is due to the simple, "bin-centered" frequency approximations given by equation (1).

If $O = 1$, meaning that the frames are not overlapping, equation (2) is an identity because a, b and p are all integer values. For $O > 1$, it equals identity whenever p is a multiple of O. In other words, perfect vertical phase coherence among frequencies is preserved on every O frame of the STFT. For intermediate frames, only $O - 1$ different phase changes are applied, and each different change is applied to a group of frequencies. Frequencies of a given group are thus still phase-coherent with each other.

Before the last step of the algorithm is presented in Section 3.5, the artifacts that are expected from the steps depicted above are discussed.

3.3. Effect on Phasiness

The presented algorithm only affects the phases of frequencies in a very limited way. Vertical phase coherence is perfectly preserved on every O frame, and preserved among large groups of frequencies on other frames.

The phase changes applied to frequency groups are hence expected to no longer introduce smearing, and to have a different repercussion on transients. For instance, in the presence of an impulse-like transient (containing energy in all, or almost all frequencies), each frequency group captures part of the transient, and the different phase modifications slightly skew these parts in time. As the number of groups corresponds to the overlapping factor, it is quite small (typically between 2 and 8). Hence transient *duplications* are expected in place of transients *smearing* in the result. This was indeed confirmed by listening tests. Regarding steady sounds, phasiness or chorus effects are no longer heard.

Figure 1 illustrates a transient processed by three algorithms: the standard phase vocoder [27], the improved vocoder proposed by Laroche and Dolson [16] (with phase locking) and the presented algorithm. The improved phase vocoder exhibits less transient smearing than the standard phase vocoder, but still more than the proposed algorithm. The proposed algorithm on the other hand duplicates the transient, as predicted.

3.4. Detuning

The scaling process depicted by equation (1) is a crude approximation of the scaled frequency: the initial frequency f_a of the phasor Ω_a is assumed to be centered in a DFT bin (yielding at most an error of half a bin), and the resulting frequency f_b of the phasor Ω_b is rounded to a DFT bin (yielding another error of half a bin). If N is the DFT size, consider the maximum error E as the ratio between the worst synthesized

[1] An overlap factor $O = 4$ with this definition corresponds to a 75% overlapping. In the literature, it is sometimes defined differently.

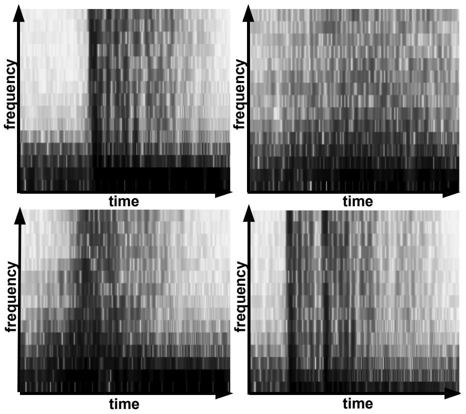

Fig. 1. Spectrogram of a transient (top left), pitch shifted with the standard phase vocoder (top right), the improved phase vocoder (bottom left) and the presented algorithm (bottom right).

frequency and the corresponding perfectly scaled frequency. For a sampling rate f_s, it is bounded by:

$$E \leq \frac{(f_a + \frac{f_s}{2N})k + \frac{f_s}{2N}}{kf_a} \tag{3}$$

Using typical values such as $f_s = 44100$, $f_a = 440$, $k = 1.2$ and $N = 2048$, equation (3) already gives a maximum error of about 1.045 which is almost a half-tone. At first glance, this looks like a highly audible error. However listening tests revealed that the error was only slightly audible for tones with very few harmonics (such as pipes), and almost imperceptible for most other instruments. To understand and model this surprising fact, it is necessary to refer to previous results on psychoacoustics and pitch perception. Namely, previous studies have shown that the perceived pitch of a sound depends on *all* its harmonics. Hence, the errors (3) for individual harmonics of a given sound will take different unrelated values and will hence tend to *cancel each other* [1, 20]. The more harmonics, the more cancellations (due to the law of large numbers), and the best quality in term of frequency tuning.

There is still a limit on the amount of errors that is acceptable, and experimental tests showed that it was not possible to use a DFT size of less than 1024 samples without introducing major detuning artifacts.

However, a simple trick allows a drastic reduction of the detuning effect and hence permits the use of smaller DFT sizes. The idea is to keep the same DFT size N for the analysis, and to use a larger DFT size for the synthesis. For a simple implementation, the synthesis DFT size can be chosen as an integer multiple m of the analysis DFT size. Equation (1) then becomes

$$b := \lfloor mka + 0.5 \rfloor$$

And equation (2) can be replaced by

$$\Omega_b := \Omega_b e^{-i\frac{(b-ma)p}{mO}\frac{2\pi}{N}}$$

As a result, the detuning error given by equation (3) becomes:

$$E \leq \frac{(f_a + \frac{f_s}{2mN})k + \frac{f_s}{2mN}}{kf_a}$$

In other words detuning errors are basically divided by m. In practice, it was found that $m = 1$ (same DFT size for analysis and synthesis) is sufficient for analysis DFT sizes of 4096 and higher (at 44.1kHz) and yields quality comparable to other algorithms. This observation is consistent with previous research involving scaled frequencies that are rounded in a similar way [16]. For an analysis DFT size N smaller than 4096, setting m somewhere between 4 and $\frac{4096}{N}$ empirically showed to give a good trade-off.

Note that using m larger than 1 proportionally increases the number of transient duplications in the result. In fact, vertical phase coherence is then only preserved every mO frames of the STFT. As the synthesis DFT is larger, transient duplications are also more spread in time. In practice, cropping the result of the inverse synthesis DFT by the initial frame size solves the problem. Cropping also ensures that no additional latency is introduced by the use of a larger synthesis window. However, this comes at the price of new artifacts which are difficult to describe and analyze, but for small values of m (typically 2 or 4), they remain minor compared to transient duplications.

While the use of a larger synthesis DFT size could also be applied to other existing algorithms such as the improved phase vocoder, this only hardly improves their quality. The reason is that most existing techniques are already using some kind of interpolation during the synthesis, which has a similar effect to increasing the synthesis DFT size [16].

3.5. Handling the Modulation Effect

The presented algorithm has to deal with another and new problem. The scaling approximation in equation (1) introduces a severe amplitude modulation in the result, which is due to the windowing process of the STFT. This modulation is illustrated in Figure 2 for a von Hann analysis window, and scaling ratios of 1.5 and 2. Indeed, a frequency f_a centered on a DFT bin a actually affects the three bins ($a-1$, a, $a+1$) during the analysis because of the windowing process[2]. After a scaling ratio of 2, this affects the three bins ($b-2$, b, $b+2$) in the result. The consequence is to transform the initial modulation (Figure 2a) of the STFT frame into a “doubled” version, as shown in Figure 2b. With a scaling ratio of 1.5, the same frequency either affects the three bins ($b-2$, b, $b+1$) or ($b-1$, b, $b+2$) after the scaling. In both cases, the

[2] This corresponds to the main lobe of the analysis window. The side lobes have negligible amplitudes and are hence ignored.

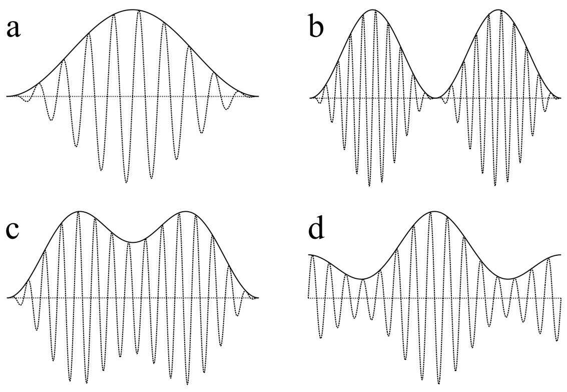

Fig. 2. Transformations of the Hann analysis window. (a) Initial window, (b) after a scaling by 2, (c) after a scaling by 1.5 with 50% overlap (even frames), (d) after a scaling by 1.5 with 50% overlap (odd frames).

resulting window is the same and is shown in Figure 2c. Finally, the phase rotations applied in equation (2) further alter the resulting window. Figure 2d shows the resulting window for odd-numbered frames with an overlapping factor of 2 and a scaling ratio of 1.5. This window is no longer zero at the frame's boundaries, which can introduce additional artifacts. Note that with an overlapping factor of O, a *cycle* of O different altered windows is produced.

Amplitude modulations occur because the resulting altered windows do not sum up to a flat curve. Increasing the overlapping factor does *not* solve the problem, as it also increases the number of different phase rotations, and hence the length of the cycle of different altered windows.

A solution is first proposed for the special cases of frequencies centered on a DFT bin, and for scaling ratios of 1.5 and 2. Then the next section shows that the proposed solution, while inexact in the general case, still offers a significant and valuable improvement.

The problem is solved in two steps: First a *synthesis* window is applied after the scaling process to ensure zero values at the frame's boundaries. Second, the cycle of altered analysis windows is computed for the current scaling ratio, overlap factor, analysis window and synthesis window; and the resulting amplitude modulation curve is calculated. After the inverse DFT and overlap-add process, the resulting time-domain samples are *divided* by the computed amplitude modulation curve, in order to "demodulate" the result.

While most previous research usually suggest the use of windows that sum up to a flat curve, the use of "demodulation" has already been investigated successfully before, as a way of producing high quality filters with windows that do not add up to a constant, such as the Kaiser or Chebyshev ones [25]. This has not yet been applied to pitch shifting though.

3.6. Generalisation

The previous section illustrated the altered analysis window for frequencies centered on a DFT bin, and scaling ratios of 1.5 and 2. With these assumptions, the resulting amplitude modulation curve can be computed *exactly* and hence be used for a perfect demodulation of the transformed signal.

On the other hand, perfect demodulation is *not* possible for arbitrary scaling ratios. With a scaling ratio of 1.75 for instance, a frequency f_a centered on bin a (and hence affecting the three bins $(a-1,\ a,\ a+1)$) can result in three different configurations after the scaling process (depending on the actual bin number a). The possible configurations are $(b-2,\ b,\ b+2)$, $(b-2,\ b,\ b+1)$ and $(b-1,\ b,\ b+2)$. The analysis window can thus be altered in two different ways corresponding to either a ratio of 1.5 or a ratio of 2.

In order to cope with arbitrary ratio, the presented algorithm solves this problem by using an *averaged* and weighted modulation curve between the two curves corresponding to the nearest scaling ratios that can be predicted exactly. In other words, the rounding errors are considered as a random process instead of being correlated to the signal. With a ratio of $\frac{15}{8}$ for example, an averaged curve between the two modulation curves corresponding to the ratios 1.5 ($\frac{12}{8}$) and 2 ($\frac{16}{8}$) is used. In this case, the curves are weighted by 25% and 75% respectively. The weights are proportional to the distances of the ratio $\frac{15}{8}$ to the ratios 1.5 and 2.

The demodulation process is only approximate as well for frequencies that are not centered on a DFT bin, although empirical tests have shown the errors to be relatively small in that situation.

The used approximations significantly reduce the modulation compared to a version that do not treat it at all. However it is clear that some artifacts remain: the resulting modulation is only approximately demodulated and hence a residual modulation can be perceived. Like with the detuning artifact discussed in Section 3.4, listening tests again revealed that this was mostly pronounced on tones with few harmonics. The explanation is similar to that of Section 3.4: for sounds with many harmonics, the resulting residual amplitude modulations of individual harmonics will tend to cancel each other. Yet more investigations in psychoacoustics should be done to accurately model and quantify this fact.

An empirical test was conducted in order to verify the relevance of the approximation in the particular case of sounds with a rich spectrum. The test was conducted by feeding the algorithm with short fragments (64 STFT frames with an overlapping factor of 8) of white noise of constant amplitude with ten different scaling ratios. Then the amplitude curve of the result (after the proposed demodulation scheme) was approximated for various scaling ratios by averaging the absolute sample values of the resulting signals from 1000 fragments. The results of this test showed that the resulting amplitude curve was mostly flat, with amplitude variations of less than 5% within the approximated amplitude curve.

3.7. Stereo Handling

As a last note, it is known that processing the left and right channels of a stereo signal independently usually results in a loss of the stereo field. This is true for most existing phase vocoders and for most time domain approaches. The reason is that the slightest difference in an analyzed parameter between the two channels may generate phase differences between the channels, that may eventually last for ever. As a result, a centered sound might get out of phase and seem to play in both channels independently after the transformation.

Solutions have been proposed to properly handle the stereo field with the phase vocoder and time domain approaches [12]. However, the presented algorithm has nice a particularity: it already handles the stereo field perfectly and does not need any such special treatment! The reason is that the algorithm is entirely independent from the signal characteristics. In other words, it does *not* perform any analysis of the signal, but transform it "blindly". As a consequence, all channels are transformed in the same way, regardless of their actual content. This property is shared by the previous algorithm proposed by the authors [1] and is also explained by the absence of an analysis of the input signal.

4. EXPERIMENTAL RESULTS

4.1. Quality and Latency

The pitch shifting algorithm presented in this paper has been implemented as a part of the existing PitchTech framework [6]. The presented algorithm, and the *improved phase vocoder* proposed by Laroche and Dolson [16] have both been tested on a wide range of audio signals, in two different configurations.

In the first configuration, the two algorithms are tuned to get high quality results, within reasonable speed. The improved phase vocoder is set to use 4096-point DFTs (at 44.1kHz). The presented algorithm is set to use 2048-point analysis DFTs and 4096-point synthesis DFTs. Both algorithms are set to use an overlapping factor of 8.

In the second configuration, the two algorithms are constrained to achieve a latency of less than 12ms. This actually means that a 512-point DFT has to be used for the improved phase vocoder. The presented algorithm uses a 512-point analysis DFT but a 2048-point synthesis DFT, as the size of the synthesis DFT does not affect the latency.

Music excerpts processed with the two algorithms in both configurations are available on-line [29]!![3] The first configuration does not reveal significant differences in term of quality between the two algorithms. As suggested in Section 3.3, transients are affected in a different way, making a formal comparison difficult.

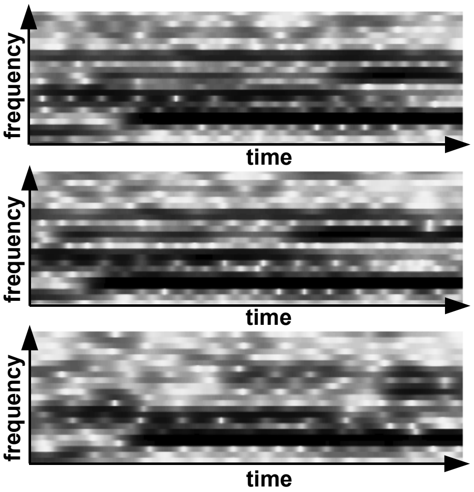

Fig. 3. Spectrogram of a polyphonic music excerpt (top), pitch shifted with the presented algorithm (middle) and the improved phase vocoder (bottom).

The second configuration on the other hand reveals notable differences. The improved phase vocoder exhibits various random attenuations and frequency modulations of the resulting signal, audible as a "roughness" effect.

Figures 3 and 4 illustrate the problem on real musical signals. The spectrograms illustrate parts of a music excerpt that has been processed by the presented algorithm and the improved phase vocoder, with the constraint of producing a latency below 12ms. In Figure 3, a long standing frequency is significantly attenuated in the case of the improved phase vocoder. In Figure 4, various closely-spaced constant frequencies have been transformed into varying frequencies (highlighted by ellipses) by the improved phase vocoder. This artifact shows up at many other places in the transformed music, and will be explained in Section 5.

Figure 3 and 4 show that the two artifacts of the improved phase vocoder are *not* present with the presented algorithm (although other minor artifacts are visible). This suggests that the presented algorithm is more resistant to reduction of the DFT size, and hence provides a better alternative in term of quality when low latency constraints are enforced, such as in live performances.

Preliminary results with the *standard* phase-vocoder [27] showed that is does not cope well with low latency constraints either, and hence is not a viable alternative. Note that the standard phase-vocoder is usually considered of lower quality than the improved phase vocoder anyway [13].

[3] The algorithm presented in this paper is referred to as the "Ocean" algorithm in the on-line pages. Some other pitch shifting algorithms have been included for comparison but are not addressed in this paper.

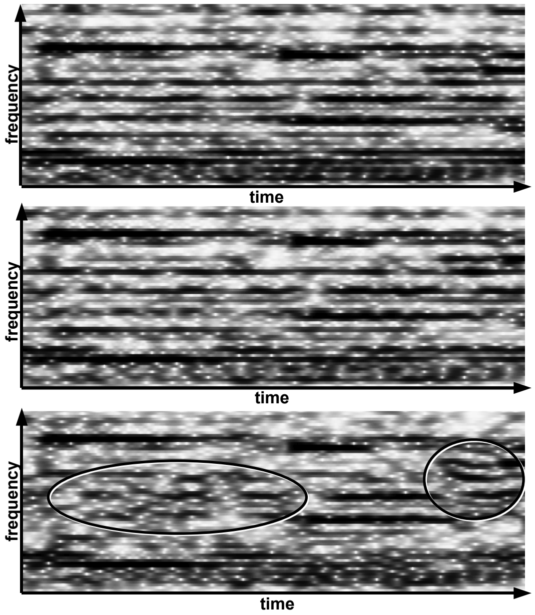

Fig. 4. Spectrogram of a polyphonic music excerpt (top), pitch shifted with the presented algorithm (middle) and the improved phase vocoder (bottom). The ellipses highlight regions where (mostly) constant frequencies were analyzed by the improved phase vocoder as varying frequencies and show up as such in the result.

4.2. Speed

It is worth noting that the presented algorithm does not require *any* transcendental function such as sines and cosines, and can be implemented entirely using the (real, imaginary) representation of complex numbers. The roots of the unity used for the phase modifications depicted in Section 3.2 can be precomputed, as well as those used for the fast Fourier transform. Furthermore, the overlapping factor has only a weak effect on the quality, and reasonable results are already produced with an overlapping factor of 2 for moderate scaling ratios.

Comparison with an implementation of the *fast* version of the improved phase vocoder [16] (using the same framework and the same DFT sizes to minimize differences due to different architectures) showed the presented algorithm to run about 10% faster on the same computer (2.4 GHz Pentium CPU running Windows) when an overlapping factor of 2 is chosen. For high quality results though, an overlapping factor of 8 is necessary as well as a synthesis DFT size twice as large as the analysis DFT size. With these settings the presented algorithm showed to run about 30% slower than the *high quality* version of the improved phase vocoder. Various fine-tuned optimizations in either implementations could potentially affect these results, and it is expected that the results might significantly vary if the algorithms were implemented in another programming language or architecture anyway. Nevertheless it reveals that the two algorithms are expected to be within the same order of magnitude in term of speed. Note that the improved phase vocoder (and hence the presented algorithm) is several times faster than the standard phase vocoder [16].

5. LOW LATENCY ANALYSIS

Section 3 described a new pitch shifting algorithm and predicted some of its properties. Namely the fact that it removes phasiness, and that it introduces transient duplications and amplitude modulations. But nothing directly explains why this algorithm is (according to the observations made in Section 4.1) more resistant under low latency constraints.

This section attempts to give an explanation by showing why the other tested algorithm, the *improved phase vocoder*, is *less* resistant under low latency constraints.

5.1. Background

The argumentation partially relies on previous results stating that the improved phase vocoder introduces an artifact when it has to synthesize a sinusoid with a frequency that varies over time (such as a vibrato). The problem lies in the fact that the STFT performs a cross-fading between synthesized frames, where an interpolation of the synthesized parameters would be more accurate [2, 22, 23]. It is shown here that with a small DFT size, the same artifact occurs even in the presence of constant-frequency sinusoids only, because a varying frequency is erroneously *analyzed*. Note that the sole detection of a varying frequency in the presence of constant frequencies is expected to introduce artifacts as well, as illustrated by Figure 4.

As a reminder, the improved phase vocoder relies on accurate estimations of individual dominant frequencies of the sound, as in sinusoidal modeling [26]. This is done by first locating the peak magnitudes in the spectrum. In a second step, interpolation is used to accurately estimate the frequency of a peak. Refer to [16] for a detailed description of the algorithm.

5.2. The Prototype Signal

Assuming a DFT size of N and a sample rate f_s, the "size" B of a DFT bin in Hz is given by $B = \frac{f_s}{N}$. Now assume a signal that is the sum of three sinusoids s_1, s_2 and s_3 of frequencies $(k-1)B$, $(k+\epsilon)B$ and $(k-\epsilon)B$, respectively, and with a zero phase. Assume that ϵ is non-zero, but substantially smaller than B. This means that s_1 falls into the DFT bin $k-1$ while s_2 and s_3 both fall into the same DFT bin k. For convenience, it is also assumed that the amplitude of s_1 is 1 and the amplitudes of both s_2 and s_3 are 0.5. Finally, it is assumed that a von Hann analysis window is used in the STFT.

The aim of the next section is to show that in presence of this prototype signal made of constant frequency sinusoids, the improved phase vocoder analyzes a single *varying* frequency.

5.3. Analysis

The two sinusoids s_2 and s_3 can be viewed as a single sinusoid of frequency kB subject to an amplitude modulation by using the following trigonometric relation:

$$\frac{1}{2}sin((k+\epsilon)B)+\frac{1}{2}sin((k-\epsilon)B) = sin(kB)cos(\epsilon B) \quad (4)$$

Therefore, the prototype signal can be equally viewed as the sum of two constant frequency sinusoids, one of which is subject to a slow amplitude modulation. These are the sinusoid s_1 and the signal (4), respectively.

Note that both the sinusoid s_1 and the signal (4) have frequencies that are multiple of the DFT bin size B. Hence if their phases are the same in the first frame, they are also the same on every O frame (O is the overlapping factor used in the STFT).

Regarding signal (4), as ϵ is smaller than B, the STFT *will* actually analyze a single frequency centered on the DFT bin k, whose amplitude varies between -1 and 1 over time. In some STFT frames, the analyzed amplitude will be close to zero and in other frames it will be close to one. As ϵ is unrelated to B, these situations also occur on frames where the signal (4) has the same phase as s_1. For the sake of simplicity, it can be assumed that the analyzed amplitude of the signal (4) exactly equals one in some frames and exactly equals zero in other frames. This assumption only introduces errors that are negligible in the following analysis.

On STFT frames where the amplitude of the signal (4) is zero, only the sinusoid s_1 shows up in the STFT, and a single frequency of $(k-1)B$ is analyzed by the improved phase vocoder, as expected [16].

On frames where the amplitude of the signal (4) is one, both signals affect the DFT bins. More precisely the sinusoid s_1 affects the bins $k-2$, $k-1$ and k with magnitudes of 0.25, 0.5 and 0.25[4], and the signal (4) affects with the same magnitudes the three bins $k-1$, k and $k+1$. As a result, the DFT bins from $k-2$ to $k+1$ take the summed magnitudes of 0.25, 0.75, 0.75 and 0.25. It is easy to figure out that the peak-based frequency estimation used by the improved phase vocoder will again analyze a *single frequency*, but of a *different* value than in the previous case. With parabolic interpolation for instance, it analyzes a frequency of $(k-\frac{1}{2})B$ instead of $(k-1)B$.

It is known that time-varying frequencies yield to artifacts in the synthesis with the improved phase vocoder [22, 23]. To conclude the argument, note that the assumption that ϵ is smaller than the DFT size B is more likely to occur in real-world signals precisely when B is large, that is, when the DFT size is small.

In the case of the new algorithm presented in this paper, there is no explicit frequency estimation and this problem does not show up. Hence reducing the latency by reducing the DFT size increases an artifact of the improved phase vocoder that is not present in the presented algorithm.

6. FUTURE WORKS

The presented algorithm works and results in fair audio quality, but it uses various approximations. While pointers have been given why these approximations are acceptable using previous research, some issues are still under investigation. For instance, it might be relevant to investigate the effect of the demodulation used in the time domain on the resulting pitch of individual frequencies, if any.

A future working direction would be to investigate if further improvements are possible by combining the proposed algorithm with hybrid or multi-resolution techniques. The presented algorithm for instance gives the worst results with pure sinusoidal signals, because the detuning and modulation effects described in Sections 3.4 and 3.5 are only reduced when many frequencies are present in the signal. Techniques based on spectral models [18, 26] suggest to handle dominant frequencies separately, a scheme that could potentially improve the presented algorithm. The main question to answer with regards to the initial goal is whether this can be done without introducing too much additional latency.

7. CONCLUSION

This paper presented a novel frequency domain audio pitch shifting algorithm that gives a quality superior to other known approaches when low latency constraints are imposed. It was validated on various audio signals and a possible explanation of its resistance to low latency constraints was proposed by studying a prototype signal.

From the theoretical point of view, an alternate way of reducing phasiness was presented, with the help of approximations and a demodulation process. It was also shown that various artifacts that seem unacceptable at a first glance are usually not significant in practice, because most sounds are built of many harmonics.

From the practical point of view, a new pitch shifting algorithm was proposed, that satisfies the low latency constraints imposed by “live” processing situations, in which a audio signal has to be captured, processed and played back in real-time within less than 20ms. The proposed algorithm has been shown to properly cope with latencies that are sufficiently small to be tolerated by a human performer that hears both the original and the transformed audio signals.

[4] These values correspond to the magnitude Fourier transform of the von Hann analysis window [11].

8. REFERENCES

[1] N. Juillerat, S. Shubiger-Banz, S. Müller Arisona, *Low Latency Audio Pitch Shifting in the Time Domain*, IEEE International Conference on Audio, Language and Image Processing, pp. 29 – 35, 2008.

[2] N. Juillerat, S. Shubiger-Banz, S. Müller Arisona, *An Hybrid Time and Frequency Domain Audio Pitch Shifting Algorithm*, Proc. of the 125th Audio Engineering Society Convention, paper number 7637, 2008.

[3] E. Lee, T. Karrer, J. Borchers, *An Analysis of Startup and Dynamic Latency in Phase Vocoder-Based Time-Stretching Algorithms*, International Computer Music Conference, 2007.

[4] R. Bradford, R. Dobson, J. Ffitch, *The Sliding Phase Vocoder*, International Computer Music Conference, Copenhagen, Denmark, vol. II, pp. 449 – 452, 2007.

[5] Jens Gulden, *JJack - Using the JACK Audio Connection Kit with Java*, Linux Audio Conference, Berlin, Germany, 2007.

[6] N. Juillerat, S. Müller Arisona, S. Schubiger-Banz, *Real-time, Low Latency Audio Processing in Java*, International Computer Music Conference, Copenhagen, Denmark, vol. II, pp. 99 – 102, 2007.

[7] T. Karrer, E. Lee, J. Borchers, *PhaVoRIT: A Phase Vocoder for Real-Time Interactive Time-Stretching*, International Computer Music Conference, 2006.

[8] E. Ravelli, M. Sandler and J. P. Bello, *Fast Implementation for Non-Linear Time-Scaling of Stereo Signals*, Proc. of the 8th Int. Conference on Digital Audio Effects, 2005.

[9] D. Dorran, *Audio Time-Scale Modification*, PhD Thesis at the Dublin Institute of Technology, 2005.

[10] N. Lago, *The Quest for Low Latency*, International Computer Music Conference, 2004.

[11] R. G. Lyons, *Understanding Digital Signal Processing*, Prentice Hall PTR, April 2004.

[12] U. Zölzer, *DAFX - Digital Audio Effects*, John Wiley & Sons, 2002.

[13] J. B. Sanjaume, *Audio Time-Scale Modification in the Context of Professional Post-production*, PhD Thesis at the university Pompeu Fabra, Barcelona, 2002.

[14] S. M. J. Hoek, *Method and Apparatus for Signal Processing for Time-Scale and/or Pitch Modification of Audio Signals*, Sigma Audio Research Limited, US Patent 6266003, 2001.

[15] Kees van den Doel, Dinesh K. Pai, *JASS: A Java Audio Synthesis System For Programmers*, Proc. of the International Conference on Auditory Display, Espoo, Finland, 2001.

[16] J. Laroche, M. Dolson, *New Phase-Vocoder Techniques for Real-Time Pitch-Shifting, Chorusing, Harmonizing and Other Exotic Audio Modifications*, Journal of the Audio Engineering Society, Vol. 47, No. 11, 1999.

[17] J. Laroche, M. Dolson, *Improved Phase Vocoder Time-Scale Modification of Audio*, IEEE Transactions on Speech and Audio Processing, vol. 7, no. 3, pp. 323 – 332, May 1999.

[18] S. Levine, J. O. Smith, *A Sines+Transients+Noise Audio Representation for Data Compression and Time/Pitch Scale Modifications*, 105th Audio Engineering Society Convention, San Francisco, 1998.

[19] J. Laroche, M. Dolson, *Phase-Vocoder: About This Phasiness Business*, Applications of Signal Processing to Audio and Acoustics, 1997.

[20] R. Meddis, L. O'Mard, *A Unitary Model of Pitch Perception*, The Journal of the Acoustical Society of America, Volume 102, Issue 3, pp. 1811-1820, 1997.

[21] M. Puckette, *Phase-locked Vocoder*, IEEE Conference on Applications of Signal Processing to Audio and Acoustics, 1995.

[22] M. Goodwin, A. Kogon, *Overlap-Add Synthesis of Nonstationary Sinusoids*, International Computer Music Conference, 1995.

[23] M. Goodwin, X. Rodet, *Efficient Fourier synthesis of nonstationary sinusoids*, International Computer Music Conference, 1994.

[24] J. Laroche, *Autocorrelation Method For High-Quality Time/Pitch-Scaling*, IEEE Proc. Workshop Appl. of Signal Processing to Audio and Acoustics, 1993.

[25] G. Smart and A. B. Bradley, *Filter Bank Design Based on Time Domain Aliasing Cancellation with Non-Identical Windows*, IEEE Trans. on Acoustics, Speech and Signal Processing, vol. ASSP-34, No. 5, pp. 1153-1161, Oct. 1986.

[26] T. Quatieri, R. McAulay, *Speech Transformations Based on a Sinusoidal Representation*, IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 489 – 492, 1985.

[27] J. L. Flanagan, R. M. Golden, *Phase Vocoder*, Bell Syst. Tech. J., vol. 45, pp. 1493 – 1509, Nov. 1966.

[28] O. Parviainen, *SoundTouch Audio Processing Library*, http://www.surina.net/soundtouch/

[29] N. Juillerat, B. Hirsbrunner, *Companion page*, http://www.pitchtech.ch/Confs/ICALIP2010/index.html