Design of a pitch quantization and pitch correction system for real-time music effects signal processing

Corey Cheng[*†]
[*]Massachusetts Institute of Technology, 617-253-2268, coreyc@mit.edu
[†]EconoSonoMetrics, LLC, coreyicheng@gmail.com

Abstract— **This paper describes the design of a practical, real-time pitch quantization system intended for digital musical effects signal processing. Like most modern pitch quantizers, this system can be used to pitch correct and even reharmonize out-of-tune singing to alternative musical scales simultaneously (e.g. major, minor, diminished, etc.) Pitch Quantization can also be intentionally exaggerated to produce distinctive effects processing which results in an emotionally inflected and/or "robotic" sound. This system uses intentionally simple signal processing algorithms which make real-time processing possible on constrained devices. In particular, we employ tools such as an octave resolver and range limiter, grain boundary expansion and contraction, and transient detection to enhance the performance of our system.**

I. INTRODUCTION

Pitch quantization is an audio resynthesis technique which alters a harmonic signal's fundamental frequency f_0 so that its resynthesized f_0 is chosen from a finite set of frequencies. In a typical musical application, pitch quantizers operate on vocal signals in which singing is slightly out-of-tune. By first estimating f_0 and then resynthesizing the singing so that the new f_0 lies in, say, a known diatonic major or minor scale, an audio engineer can correct a singer's pitch.

In addition to this very practical use whose aim is to produce a certain transparency in audio, creative applications of pitch quantizers can create special musical effects or serve different musical functions, such as pitch shifting / transposing vocals to higher or lower registers; reharmonizing vocals according to a new musical key from major to minor or diminished scales, etc. In addition, recent popular musics exaggerate the use of pitch quantizers to produce a very fashionable "robotic" vocal effect, popularized by artists such as Cher and T-Pain. Some current pitch quantization programs are the Antares "Autotune," Celemony "Melodyne," and Smule "I Am T-Pain" products[1][2][3].

In general, pitch quantizers work by first estimating the fundamental frequency f_0 in a small segment of audio and then resynthesizing that segment of audio according to a new f_0. In this sense, pitch quantizers rely heavily on frequency estimation and pitch period estimation, and these processes comprise the most important part of a pitch quantizer. Because pitch quantizers make use of this estimation, they are also closely related to time compression and expansion systems, which usually exploit these estimates to resynthesize the original segments with different lengths [4][5][8][9]. Frequency and pitch period estimation are well-developed and can be done with a myriad of different time- and frequency-domain methods, some of which have become very intricate. These estimators are at the heart of many other sophisticated musical signal processing techniques such as music transcription, multiple fundamental tracking, and source separation [6][8].

However, the purpose of this paper is to show how some very simple, low-complexity methods can be used to create a practical pitch quantization system which can produce pitch corrected, pitch shifted, and/or reharmonized audio with quality approaching currently available commercial methods. With some careful tuning "tricks" we show that more complex methods are unnecessary, and that these simpler methods can produce a system suitable for real-time implementation on resource constrained devices.

II. SYSTEM DESIGN

A. Requirements

We designed and implemented a pitch quantization system which accomplishes pitch correction, pitch shifting, and pitch reharmonization according to an arbitrary musical scale, such as diatonic major or minor scales, microtonal scales, just intoned / Pythagorean scales, etc. The system produces the exaggerated "robotic" voice effect described above. It operates in real time for 44.1kHz and 8.0 kHz 16-bit mono audio on a mobile device (iPhone 3GS), and has a reasonably low latency of about 20-40ms in order to produce real-time pitch correction without unduly disturbing a performing vocalist. Because the system will primarily operate on singing speech, it preserves the sound quality of transients in certain speech sounds like fricatives and plosives. It is robust enough to handle inexperienced users' inadvertent speech input, such as half-voiced speech, laughing, and low-signal level input. In terms of code complexity, both MATLAB and C/C++ prototypes were implemented in two months by a single programmer, and produced audio quality comparable to commercially available software.

B. Design

Based on these constraints and some key observations, we designed and implemented the pitch quantization system in Figure 1. The system divides input audio into three non-overlapping blocks of L=1024(256) samples at 44.1 kHz (8kHz), producing an ideal latency of one frame, or 23.2ms (32ms). Fast autocorrelation methods, similar to those used in [11], determine the dominant pitch period pp, in samples, in a

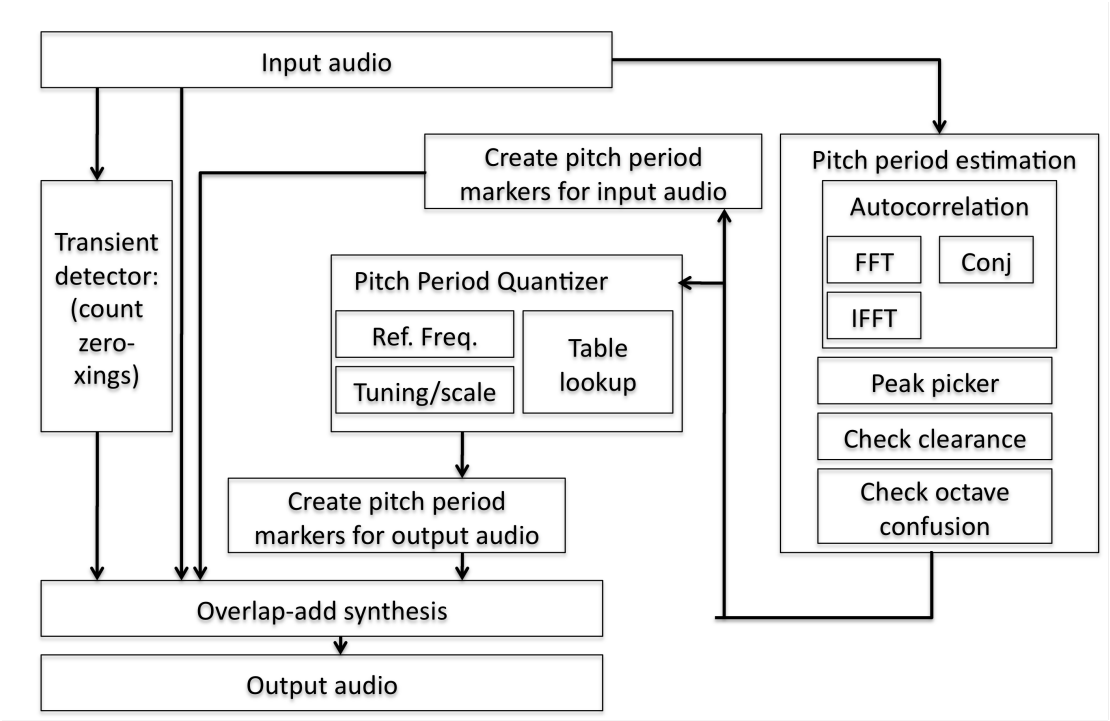


Figure 1. Block diagram of proposed pitch quantizer.

limited subset of these three windows' samples. The quantization block chooses a new pitch period pp' from a lookup table comprised of pitch periods corresponding to the notes of a specific musical scale. Next, the system employs a popular time-domain speech resynthesis technique, pitch-synchronous overlap- add (PSOLA), shown in in Figure 2. Using this method, individual grains of sound having length approximately pp' (samples) are extracted from the middle input window at regular intervals of pp, and are replaced in the output frame buffer at regular intervals corresponding to pp'. The middle frame is then sent to the output, resulting in the single frame latency, and is saved for the next frame's processing.

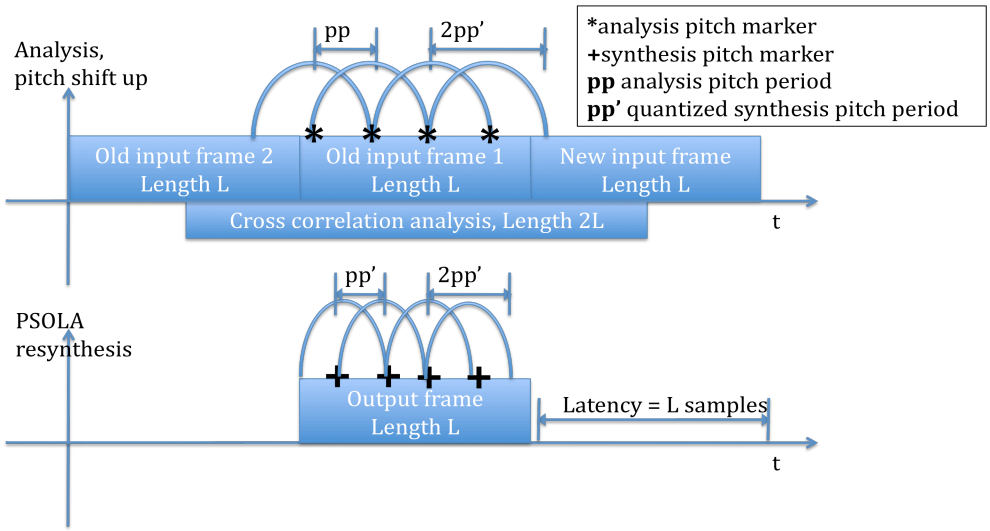
This system is similar to one of the pitch correction methods in [9], but employs some additional enhancements to increase its performance, which we discuss next.

C. Enhancements to pitch period estimation

One key observation in designing this system is to realize that potentially the most complicated part of the quantizer, the pitch period estimator, does not have to be extremely accurate, and thus can be greatly simplified. One reason is that the quantizer will only need to choose a pitch period from those corresponding to a musical scale having relatively few pitches spaced far apart in frequency. Another reason is that quantized pitch periods do not have to be exactly time synchronized with true pitch periods. Restricting the quantized pitch period to change only under certain circumstances, in a slightly delayed manner, can stabilize the output and also produce good artistic results. In fact, this is exactly how a pitch quantizer produces the currently fashionable "robotic" voice effect. To be clear, building hysteresis into the change from one frame's pp' to another does not increase the system latency at all; however, delaying the change in pp' for not only one but perhaps several frames can be used to good practical and artistic effect. For these reasons, we choose the simple autocorrelation method of pitch period estimation, especially since pitch periods can be read directly off of autocorrelation records. Thus, even though true pitch f_0 can change significantly in a single frame, the system only has to produce pitch period estimates accurate to 50-100 cents in pitch for diatonic scales, and synchronized to within 40-60ms of true pitch.

Nonetheless, pitch period estimates need to be accurate enough to resolve higher frequencies, where pitch periods which differ by only a few samples can differ significantly in corresponding frequency and pitch class. These estimators also need to be able to detect frequencies in the low male range. When dealing with low pitches, the larger the autocorrelation window size, the more computational power is required, but the lower the pitch can be detected. The naïve solution is to increase the length L of all of the analysis windows and to use the longest possible autocorrelation analysis window of 3L. However, this significantly increases the system latency and computational burden. We employ a simple solution which decouples the input audio frames from the autocorrelation window. We use a variable autocorrelation window size, nominally 2L samples, centered around the second input audio frame. In this manner, we sacrifice a slightly larger computational burden for some increase in low frequency performance, while keeping the system latency the same, at L samples. This scheme is shown in Figure 2.

a)



b)

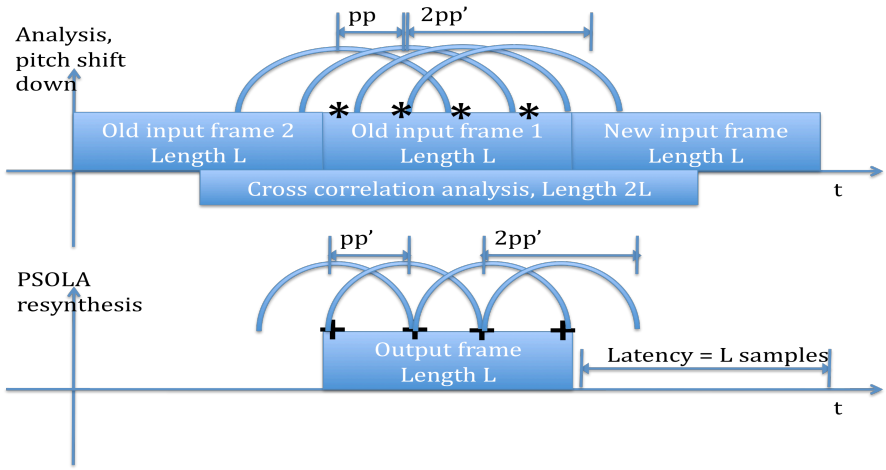


Figure 2. PSOLA-based based pitch quantization. Figure 2a (Figure 2b) shows pitch shifting upwards (downwards) to a target pitch period pp' using an initial pitch period estimate of pp. Grains of audio extracted from the input frames are truncated (enlongated) on either side before insertion into the output frame if more than three grains overlap at any given point (if silence would result in using non-enlogated grains).

Another enhancement we introduce to improve sound quality is an octave resolver and an intervallic range limiter. First, we limit the allowed change of quantized pitch period estimates to a major tenth upward and downward from one frame to the next, since it is highly unusual for most singers to make this change within 32ms, the maximum frame size of the current system.

Next, we use an octave resolver to remove octave ambiguities in some frames. It is well known that for certain signals, peaks in the autocorrelation function with similar strengths can occur at lags corresponding to octaves above or below the true fundamental. For example, the transition from glottal to harmonic sounds in the same frame of audio can produce these types of octave ambiguities in the autocorrelation function.

We use the following simple scheme to determine whether an octave error has occurred. While the two components of this algorithm are both imperfect and quite simple when used alone, their joint use produces surprisingly good results:

1. Limit the range of intervallic change from pp in the current frame to pp' in the previous frame. If pp' differs from pp by a musical interval larger than a major 10th, ignore the current frame's pp and use pp' from the previous frame instead. Goto step 5.
2. Retain the ordinates of the strongest three peaks from a simple peak picker operating on the autocorrelation function of the current frame.
3. If:
   a. the strongest peak in a frame is not higher than the next highest peak by a certain percentage, or clearance; and if
   b. all three peaks in the current frame are in octave relationships to within a certain hysteresis; and

c. if the autocorrelation function from the previous frame produced a quantized pitch period estimate pp' in the same pitch class but in a different octave than the strongest peak in step a; then
d. an octave error has occurred.

4. If an octave error has occurred in step 3, then ignore the pitch period estimate pp of the current frame and use the same quantized pitch period estimate pp' of the previous frame in the PSOLA resynthesis of the current frame.
5. Complete the rest of the PSOLA resynthesis using the chosen pp'.

D. Enhancements to PSOLA-based resynthesis

Another key observation is that time-domain speech resynthesis methods such as PSOLA preserve the fine time-domain structure of transient signals very well. We choose not to use popular frequency domain techniques like the phase vocoder for this reason, since more complicated tools are needed to handle transients and to avoid some "phasy" resynthesis artifacts [7].

As shown in Figure 2, the basic PSOLA algorithm reorganizes windowed grains of input sound originally regularly spaced at intervals of pp into the output frame at regularly spaced intervals of pp'. The input grains have length 2pp', and the difference between pp and pp' determines the amount of overlap. If pp<pp', the resulting audio is shifted downwards to the next quantized pitch level, and vice-versa. If pp=pp', there is a 50% overlap in the grains in the output frame, the resynthesis essentially produces no pitch change, and the output and input audio sound largely the same even though the waveforms are slightly different.

While time domain processing is fast and simple, there are some important details to implement in order to produce good sounding audio. For example, when pp>>pp', pitch is shifted upwards by a large amount. In this case, several grains can overlap at any one instant in time, producing an unwanted comb filtering, flangy effect. To avoid this, The grains of audio in Figure 2a are reduced in size from 2pp', one sample at a time from the left and right sides, until only a maximum of three grains overlap. Similarly, when pp<<pp', pitch is shifted downwards by a large amount, and the grains of sound can be placed so far apart in the output that there is silence between them. This can cause an unnatural, pulsed glottal effect – a usually undesirable effect for transparent audio but which we note can be an interesting artificial effect in some cases. To avoid this in the current implementation, the grains of audio in Figure 2b are augmented one sample at a time on either end past a total length of 2pp' so as to avoid silence between grains. An important wrinkle in this scheme is that there need to be enough samples in the three available input windows to allow for grain expansion at low frequencies, and this requirement can further restrict pp to have a lower than originally designed upper bound.

We also investigated a number of other PSOLA parameters, but in the interest of both simplicity and sound quality, we left these parameters out. For example, there is some debate as to whether the grains should be taken (replaced) at regular or irregular intervals in the input (output) frames. Some PSOLA systems are careful to extract grains from the input so as to align as best as possible from grain to grain certain time-domain features, such as waveform peaks or zero-crossing locations. To this extent, we tried using an intelligent grain selection system to observe these parameters. However, we found that irregularly taken (replaced) and/or feature-aligned grains produced an unstable, inconsistent wavering output at the cost of a large mount of computing power.

E. Enhancements to transient performance

Another key observation is that the time-domain nature of PSOLA resynthesis preserves the sound quality of transient speech sounds such as fricatives and plosives better than other methods. While this is another reason we choose to use PSOLA resynthesis, these sounds need to be handled correctly in a pitch shifting application. Changing the quantized pitch period pp' abruptly in frames near transients can produce undesirable artifacts such as scratches and clicks due to a changing grain overlap percentage in these frames. Therefore, what is needed is a way to essentially turn off the entire pitch quantization system in these frames, so that these transients are simply passed through unaltered to the output.

We use a simple transient detector constructed from a zero-crossing detector to determine whether or not a transient is present in the current frame. If the number of zero crossings is high, we declare a transient is in the current frame, and artificially set the estimated pitch period pp for the current frame to the quantized pitch period pp' in the previous frame, ignoring the current frame's actual estimate of pp. By forcing pp=pp', the input audio is effectively passed to the output without pitch correction, as explained in the previous section. In this manner, the transient is left intact and unprocessed. This tactic works because this reconstruction property does not rely on the exact value of pp'.

F. Pitch period quantizer

The actual pitch quantizer block in Figure 2 is very basic, and consists only of a comparator and a lookup table which holds the pre-computed pitch periods of a known and desired scale. The lookup table simply contains pitch periods corresponding to notes from the scale's predetermined reference pitch such as A4=440 Hz. Nonetheless, this block produces many different musical effects by changing how the comparator works and maps pp to the pp' values in the lookup table. Inclusion of hysteresis to the comparator adds robustness and stabilizes the audio output, while using always-round-up or round-down strategies can produce an intermittent, inflected diatonic sharpening or flattening effect while still keeping the singer in tune with a particular scale. The quantization block also implements transposition effects by mapping pp' to another table entry a fixed number of scale degrees higher or lower. Finally, the quantization block can also reharmonize vocals on the fly by using a lookup table with pitch periods derived from the desired scale. For example, using a major scale lookup table with vocals sung in the parallel minor key will force the resynthesized output to be in the major mode.

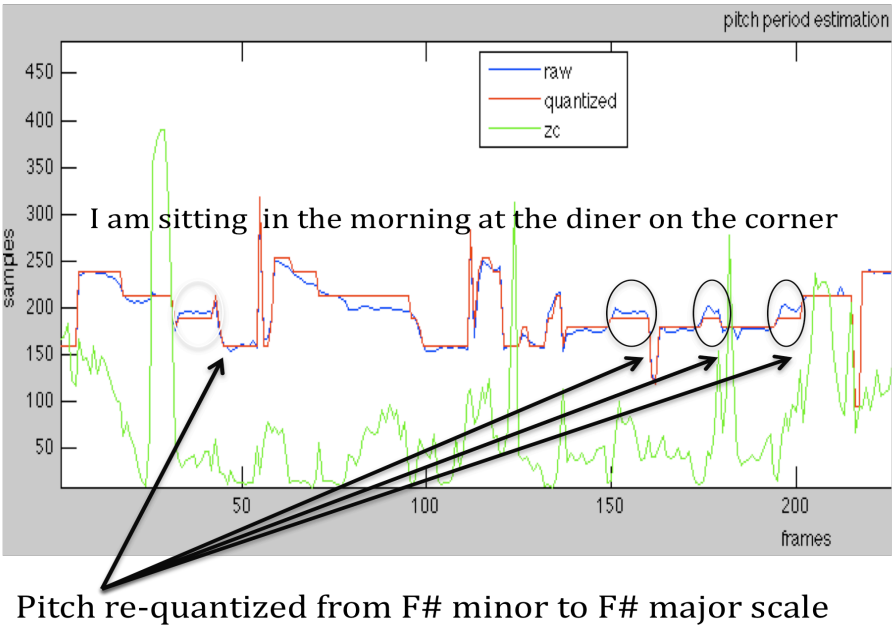


Figure 3. Pitch quantizer used for reharmonization of Suzanne Vega's "Tom's Diner" from minor to major mode.

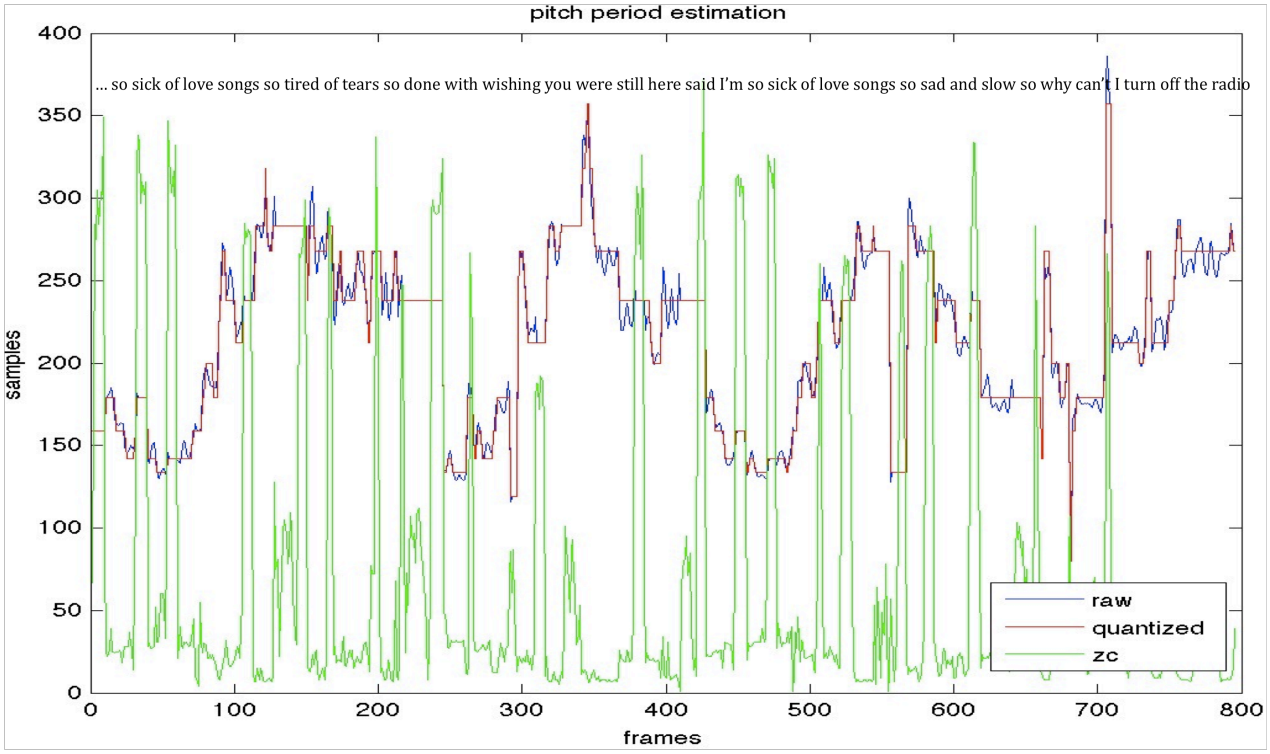


Figure 4. Pitch quantizer used for pitch correction of Ne-Yo's "So Sick."

III. EXAMPLES

Figure 3 gives an example of how the pitch quantizer can be used to reharmonize vocals from a minor to a major mode. The source audio is the beginning to a well-known pop song by Suzanne Vega called "Tom's Diner." In this example, the syllables "sit"; "morn"; "dine"; and "corn" are all sung on the pitch "A", the third degree of the F# natural minor scale, a minor third above F#. The pitch quantizer can be used to convert the syllables above to sound at A#, one semitone higher, so that the segment sounds as if it is in the key of F# major. Figure 3 shows how the raw pitch period estimates are mapped to quantized pitch periods which lie in the F# major scale. This figure also shows how zero crossings increase near some fricatives, such as the beginning of the syllable "sit," and force the pitch quantizer to turn off near those areas.

Figure 4 gives another example of the pitch quantizer. This excerpt is from another pop song by Ne-Yo called "So Sick." This longer excerpt shows how out of tune vocals can be fixed, and gives more extensive examples where the zero crossing transient detector forces the pitch quantizer off.

While we did not conduct formal listening tests, we informally compared our system to the output from the Antares Autotune product line, and the found that the two systems produced comparable results. Interested readers are welcome to contact the author for sound examples.

IV. Shortcomings and Room for Improvement

Our system can be improved in a number of ways. The pitch period detector can be fooled by gutteral sounds like /g/, and the octave resolver is not perfect in all cases. The limited size of the autocorrelation analysis window in Figure 2 places a lower bound on detected frequencies which is not often low enough for deep male voices. Also, pitch periods in our system are only allowed to be whole number of samples. This causes some problems at high frequencies, where very short pitch periods cannot exactly match the resolution required by scale notes in the upper registers. The problem is most pronounced for high female singers at the lower operating sampling rate of 8kHz.

Although the ideal latency is one frame in the current implementation, we note that overall latency of a production system is highly dependent on the underlying hardware and driver set of a given platform. So, while our system has an idea latency of one frame, we have noted a higher actual latency on different platforms. This prototype has been implemented for iOS running on the iPhone 3GS, for Apple OSX using the Core Audio framework as an Audio Unit, for Apple OSX using the PortAudio API, and for a Ubuntu 12.04 (embedded linux) platform on the Beagleboard XM using the ALSA driver set and JACK audio framework. Each of these systems handles hardware latency in different ways, and each adds a minimum of one to three more frames of latency to the overall signal path.

Notwithstanding extra platform-dependent latency, we note that the ideal latency of our system can be made considerably lower than one frame by using a time-domain pitch period detector which operates directly on the time domain waveform. These methods require only a few pitch periods for their computations, as opposed to requiring autocorrelation functions computed from a full frame of samples. In fact, we first experimented with the methods given in [10] and [12] and found them to be extremely fast and efficient. However, these methods require careful tuning and can be hard to control, so we opted for the currently implemented autocorrelation technique.

The logic relating different components of the system can be difficult and complex. For example, different combinations of exceptions occurring at the same time can be difficult to handle. Octave confusion, low signal power, and the occurrence autocorrelation functions without strong peaks can happen in combination, and care must be taken to handle these cases in a reasonable manner.

The quantization block in Figure 2 is not fully automatic. This is because its block's lookup table must be constructed from a predetermined reference pitch, such as defining A4=440 Hz. A fully automatic system would determine the reference pitch automatically.

V. Conclusion

This paper introduced a practical pitch quantization system using simple signal processing algorithms. The system exploits the time-domain nature of the PSOLA algorithm to preserve transient quality, and uses other tools like an octave resolver and grain boundary expansion and contraction to improve sound quality. We have implemented this system to produce a pitch correction system comparable to other commercially available products, and we look forward to continuing to port our algorithm to other platforms, such as imbedded and newer mobile devices.

References

[1] "Autotune" application from Antares. Website: http://www.antarestech.com/products/auto-tune-7.shtml

[2] "Melodyne" application from Celemony. Website: http://www.celemony.com/cms/index.php?id=products_editor

[3] "I Am T-Pain" iPhone application from Smule Corp. Website: http://iamtpain.smule.com/

[4] Crockett, Brett G. "High quality multi-channel time-scaling and pitch-shifting using auditory scene analysis." New York: *115th Audio Engineering Society Convention*, Paper 5948, 2003.

[5] James, Nichols. "An Interactive Pitch Defect Correction System For Archival Audio." Budapest, Hungary: *20th International Audio Engineering Society Conference: Archiving, Restoration, and New Methods of Recording*, 2001.

[6] Klapuri, Anssi P. "Multiple Fundamental Frequency Estimation Based on Harmonicity and Spectral Smoothness." *IEEE Transactions on Speech and Audio Processing*, vol. 11 no. 6, November 2003.

[7] Laroche, J. "Phase vocoder: about this phasiness business." New Paltz, NY: *1997 ASSP Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*.

[8] Laroche, Jean and Dolson, Mark. "New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects." New Paltz, New York: *1999 Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*.

[9] Lech, M. and Kostek, B. "A system for automatic detection and correction of detuned singing." Paris: *Acoustics '08*, European Acoustics Association.

[10] Rabiner, Lawrence and Shafer, Ronald. *Digital Processing of Speech Signals*. New York: Prentice Hall, 1978.

[11] Verhelst, W. and Roelands, Marc. "An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech." Minneapolis: *1993 International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

[12] Zölzer, Udo (ed.). *DAFX – Digital Audio Effects*. West Sussex, England: John Wiley & Sons, 2002.