ACOUSTIC ECHO CANCELLATION WITH ARBITRARY PLAYBACK SAMPLING RATE

Jack W. Stokes and Henrique S. Malvar

Microsoft Research
One Microsoft Way, Redmond, WA 98052, USA

ABSTRACT

This paper introduces a new architecture for implementing subband acoustic echo cancellation (AEC) with arbitrary playback sampling rate. Typically, in AEC algorithms for audio or video conferencing, the sampling rates for the signals played through the speakers and captured from the microphones are identical. For speech recognition while playing CD-quality music and Internet gaming with voice chat, the playback sampling rate is usually higher than the capture rate. A direct solution is to apply a sampling rate converter to the playback signal before feeding it to the AEC, but that is complicated if many sampling frequencies must be supported. We propose a more efficient solution for subband AEC: we perform the sampling rate conversion as a frequency-domain interpolation that matches the transform lengths of the playback and capture signals. Results show that the new AEC architecture has a small computational cost and only a minimal reduction in echo attenuation.

1. INTRODUCTION

Acoustic echo cancellation (AEC) removes the echo captured by a microphone when a sound is simultaneously played through speakers located near the microphone. In the past, many AEC algorithms have been proposed for telecommunication scenarios such as videoconferencing and speakerphones [1][2]. Typically, the sampling rates for the signal captured from the microphone and the signal played through the speakers are identical and are dictated by the voice codecs used in the application. For example, high-end videoconferencing systems use wideband or super-wideband codecs, with sampling at 16 kHz or 32 kHz, respectively, while low end, plain old telephone system (POTS) speakerphones use 8 kHz sampling. However, new scenarios often require that the playback sampling rate be different (and usually higher) than the capture sampling rate.

For example, a speech recognition system may capture the microphone signal at 16 kHz or 22.05 kHz and needs to cancel the echo from any source played by the computer such as CD quality music at 44.1 kHz or a DVD audio stream sampled at 48 kHz. A PC-based videoconferencing system needs to cancel the 44.1 kHz system sounds generated by the computer.

Enabling these new scenarios requires some form of sampling rate conversion (SRC) to match the playback and capture sampling rates. A traditional sampling rate converter based on a polyphase filter structure [3] could be used, for example, but a high-quality result would require long filters that would increase the number of computations per sample, and it would also require additional data buffering structures. Furthermore, each combination of playback and capture sampling rates would require a different filter coefficient table. In this paper we present a more efficient solution in terms of code and memory size, which is applicable if the AEC uses a subband structure: after computing the transforms for the capture and playback signals, the transform for the playback signal is interpolated in the frequency domain to match the transform size of the capture signal for the appropriate number of frequency bins. We show that this approach leads to good results in practice, especially when the subband decomposition uses the modulated complex lapped transform (MCLT) [2].

This paper is organized as follows. In Section 2, we describe the system architecture for the AEC algorithm with arbitrary playback sampling rate. Performance results are discussed in Section 3, and conclusions are provided in Section 4.

2. AEC ARCHITECTURE DESCRIPTION

An adaptive subband based AEC system is shown in Fig. 1. The audio signal to be played out of the speaker, x, with sampling rate F_x is sent to the digital-to-analog converter (D/A). The resulting analog signal is then played out through the speakers and produces an echo at the microphone. In addition to the echo from the speakers, the audio signal captured by the microphone is also composed of the desired speech and background noise.

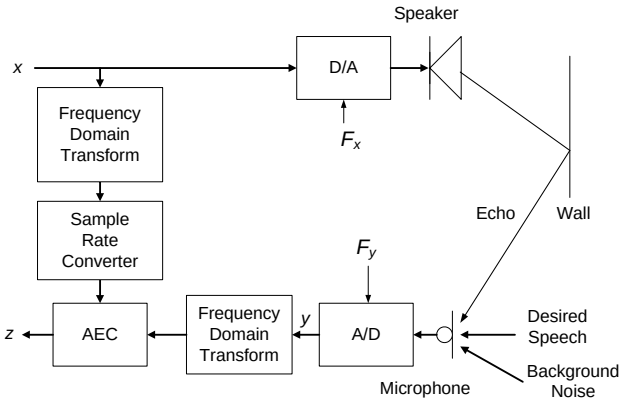

Figure 1: Acoustic echo cancellation system.

The analog audio signal from the microphone is then converted to the digital capture signal *y* by the analog-to-digital converter (A/D), which operates at a sampling rate F_y. The processed microphone signal *z* has the echo removed by the AEC module.

AEC is often performed using adaptive subband filtering using transforms such as the fast Fourier transform (FFT) or the modulated complex lapped transform (MCLT), as shown in Fig. 1 [2], in which *z* is a transform-domain output. If the playback sampling rate F_x matches the capture sampling rate F_y, then the two frequency-domain transforms can have the same length, as in [2]. For the general case where $F_x \neq F_y$ we need to apply an SRC to the playback signal. There are a number of ways to accomplish that task, including time domain SRC, exact frequency-domain SRC and interpolated frequency-domain SRC, which we review next.

2.1 Time-Domain SRC

SRC can be achieved using time-domain techniques based on multirate filtering [3]. Linear interpolation is the simplest approach, but it leads to aliasing levels that produce audible distortion and significantly compromise the AEC performance. With multirate filtering, every combination of different capture and playback sampling rates supported by the system must be handled by either separate polyphase finite impulse response (FIR) filters, or by a very long polyphase FIR filter that can be stepped at different increments for the various sampling rates.

2.2 Exact Frequency-Domain SRC

Another approach is to compute the exact frequency domain transform for each of the playback sampling rates supported by the system. In a standard frequency domain-sampling approach [4], sampling rate conversion occurs after the frequency-domain transform for the signal *x* in Fig. 1. For example, with a CD-quality playback signal at sampling rate at 44.1 kHz processed with 20 millisecond frames, we need an 882-point MCLT, which can be implemented [5] by a 1764-point FFT. The FFT length is factorable as 1764 = 2*2*3*3*7*7. Therefore, the 882-point MCLT could be implemented using the generalized Cooley-Tukey FFT [6]. To perform SRC in this case where the transform length exactly matches the number of points in a frame, we simply discard the frequency domain coefficients for the bands above the capture sampling rate, when the playback sampling rate is higher than the capture sampling rate. Likewise, when the playback sampling rate is lower than the capture sampling rate, SRC simply includes zero padding the frequency domain bands of the transformed playback signal up to the length of the transformed capture signal.

2.3 Interpolated Frequency-Domain SRC

The main disadvantage of the approach above is that it requires FFTs whose lengths are not easily factorable, leading to more complex and significantly less efficient implementations. We now present a new architecture that combines a frequency domain transform whose length is a power of 2, and a sampling rate converter using a simple frequency-domain interpolation. This approach slightly degrades the quality of the AEC algorithm as compared to the exact frequency domain approach, but is more efficient for real-time implementation. The new transform size $\hat{N}$ is given by

$$\hat{N} = 2 \wedge \left(\lceil \log 2(N) \rceil \right) \quad (1)$$

where N is the exact transform size and $\lceil n \rceil$ is the ceiling function of n. Again, we consider the case of CD-quality music playing at a sampling rate of 44.1 kHz with 20 ms buffers, for which the size of the exact MCLT transform is 882 and the size of the new MCLT is 1024. Unlike the 882-point MCLT required for the exact frequency domain transform, the 1024-point MCLT can be implemented using 2048-point FFT [5], which will be much faster than the 1776-point FFT required for the 882-point MCLT fir the exact frequency domain transform. One issue with this approach is that the size of the new MCLT window is larger than the number of samples required for the exact frequency-domain transform. There are several ways of handling this mismatch, as listed in Table 1. For completeness we have included option number 5, even though it performs worse than the other widowing methods.

Method Number	Windowing Method Description
1	Exact transform with 1764-point window and 1764 data samples.
2	Interpolate with 2048-point window and 2048 data samples overlapped back in time
3	Interpolate with 2048-point window and 1764 data samples zero padded at the beginning
4	Interpolate with 2048-point window and 1764 data samples zero padded equally at both ends
5	Interpolate with 2048-point window and 1764 data samples zero padded at the end

Table 1: Different methods of interpolating playback subbands.

After running the longer MCLT on the speaker data *x*, we now need to convert the frequency domain subbands to match the appropriate frequency bin locations of the capture data. We achieve that via linear interpolation, in the form

$$X'(m) = \left[(n+1) - m\frac{\Delta X'}{\Delta X} \right] X(n) + \left[m\frac{\Delta X'}{\Delta X} - n \right] X(n+1) \quad (2)$$

where $X(n)$ is the nth frequency bin of the transformed speaker signal *x*, $X'(m)$ is the mth frequency bin of the linear interpolated transform, and ΔX and $\Delta X'$ are the widths of the frequency bins for the transformed speaker signal and desired speaker signal (i.e. equivalent to the bin width of the capture signal), respectively. Higher order interpolation could also be used, but in the next Section we show that even linear interpolation as in (2) provides very good results.

3. PERFORMANCE EVALUATION

We now evaluate the performance of the new AEC architecture using frequency-domain interpolation, in terms of numerical accuracy and CPU consumption. For these results, the capture sampling rate is 16 kHz, typical of wideband conferencing,

while the playback sampling rate is 44.1 kHz, typical of system audio or CD-quality music playback. We consider only mono playback in this paper, since stereo playback leads to additional issues, which we will address in a future paper. Likewise, the capture signal is also mono.

The captured echo signal is simulated using a transfer function measured in standard corporate office with approximate dimensions 10'×10'×8'. The office's transfer function is first estimated with the playback and capture sampling rates set at 44.1 kHz. Convolving the music signal with the office's transfer function simulates an echo at 44.1 kHz. Next the simulated echo signal is downsampled via a high-quality polyphase filter to the desired capture sampling rate of 16 kHz.

In this paper we perform the subband AEC using adaptive subband filtering and an MCLT-based subband decomposition, as described in [2] (other subband transforms could be used, such as those based on oversampled FFT filter banks [3]). In the following results, the capture signal is processed using a 320-point MCLT while the playback signal is processed using a 1024-point MCLT for methods 2–4 in Table 1, and a 882-point MCLT for the exact transform method (method 1 in Table 1). The complex adaptive filters in each subband are processed using normalized least mean square (NLMS), as in [2]. We compare the results based on the echo return loss enhancement (ERLE), in dB, defined as:

$$ERLE(n) = 10\log_{10}\left(\frac{\mathrm{E}\{y^2(n)\}}{\mathrm{E}\{z^2(n)\}}\right) \tag{3}$$

where E{} is the expected value at time sample *n*. In this paper, we compute the expected value for non-overlapping length-*N* blocks as

$$ERLE(k) = 10\log_{10}\left(\frac{\mathrm{var}\{y(n:n+N-1)\}}{\mathrm{var}\{z(n:n+N-d-1)\}}\right) \tag{4}$$

for the *k*th block of data where *n* is the time index at the beginning of the data block, var{} is the variance of the block of data, and *d* represents the processing delay due to the AEC processing. For the AEC processing with the MCLT, *d* is equal to two frames of data (e.g. 640 samples at 16 kHz).

The numerical results provided in Fig. 2 compare the first four methods given in Table 1. For the results in Figs. 2 and 3, we set *N* equal to 5 times the frame size which is 100 milliseconds for the 20 millisecond frames used in this experiment. We see that the results from all methods are comparable, so to improve the readability, we next show in Fig. 3 the difference between ERLE for windowing methods 2–4 and the ERLE for the exact frequency-domain transform.

Since the results in Figs. 2 and 3 show the performance of the various methods on a short-term time scale, we next seek to compare the ERLE results based on the statistics of a longer time scale. Therefore we set *N* equal to 100 times the frame size (2-second blocks) in Figs. 4 and 5. Again, we compare the ERLE for the first four methods in Table 1 in Fig. 5, and the ERLE differences for windowing methods 2-4 and method 1 in Fig. 6.

As we can see from Figs. 2–5, the exact frequency domain transform usually provides the highest ERLE, as expected, although other methods can produce better short-term ERLE results. Comparing the ERLE results for the various interpolated window methods, overlapping back in time (method 2) works

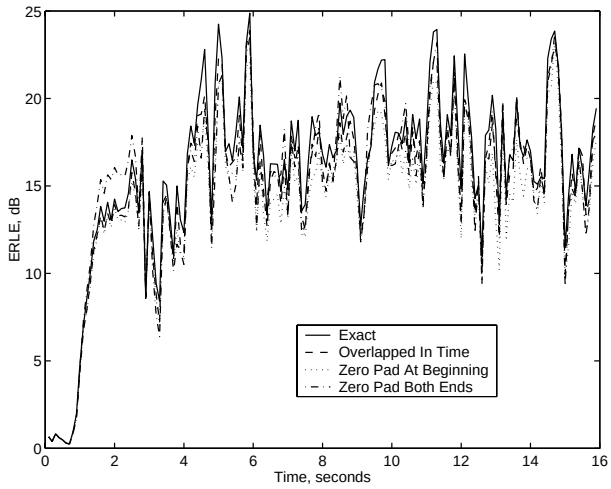

Figure 2: ERLE for methods 1–4 in Table 1, averaged over 100 ms blocks.

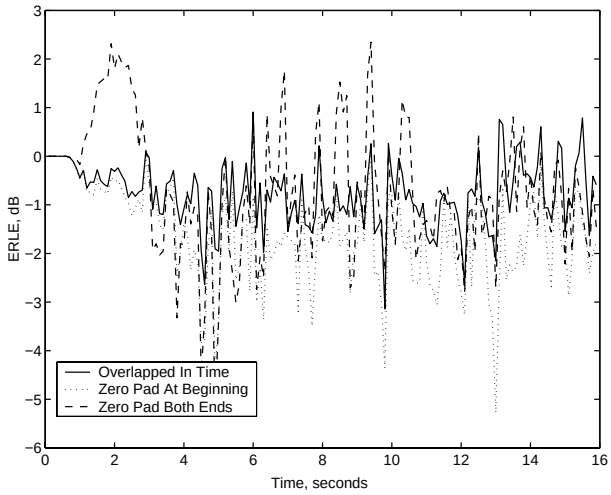

Figure 3: ERLE differences between methods 2–4 and the exact transform used in method 1, averaged over 100 ms blocks.

best and reduces the ERLE by less than 1.0 dB when compared to the exact frequency-domain transform (method 1, with an 882-point MCLT). Figures 4 and 5 show that, on average, zero padding at both ends is worse than using an overlapping window in time, but performs better than zero padding at the beginning of the window. This result is due to the fact that more energy in the signal is preserved by zero padding the tails of the window than zero padding a longer portion of the signal in the beginning of the window.

We should emphasize that our interpolated frequency-domain SRC method does not use an overlap-add or overlap-save structure, which would be equivalent to a linear time-invariant filter [3]. We just interpolate the frequency-domain coefficients within a frame, and thus our interpolation procedure is a periodically time-varying operator [3], as are the AEC

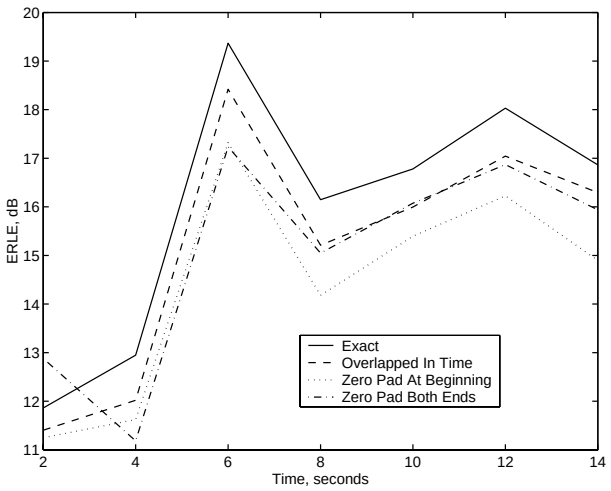

Figure 4: ERLE for methods 1–4 in Table 1 averaged over blocks of 2 s.

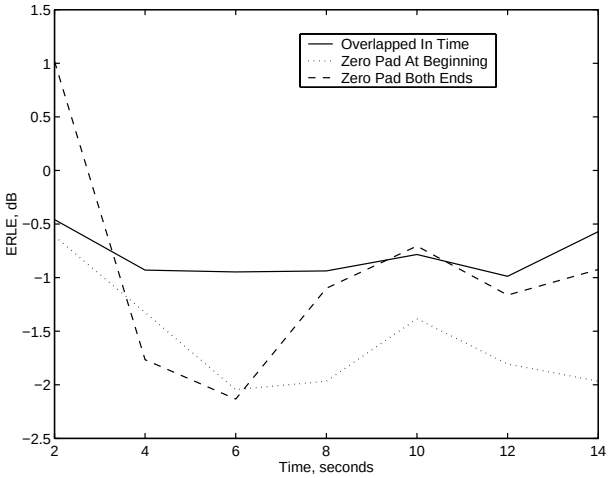

Figure 5: ERLE differences between methods 2–4 and the exact transform used in method 1, averaged over blocks of 2 s.

adaptive filters in Fig. 1 [2]. Thus, the small ERLE reduction is caused by the additional aliasing of our SRC. In practice, audible quality is also very important, and listening test have shown that our interpolated frequency-domain SRC method does not introduce any audible artifacts, especially under the presence of the mild artifacts caused by the nonlinear spectral attenuation that usually follows the adaptive AEC filters [2].

Our main motivation for proposing the interpolated frequency-domain SRC method was to reduce the code complexity and memory requirements to perform SRC for a variety of supported sampling frequencies. However, computational complexity increases with frequency-domain SRC. Thus, in Table 2 we compare CPU consumption for the various windowing methods. The results are all measured from C-language implementations running on an 800 MHz Intel® Pentium III. For reference, we included the CPU usage of an AEC system that does not need SRC, by running it with a 16 kHz playback signal; that requires 3.9% of the CPU. For a 44.1 kHz playback signal, we see that running the exact transform consumes 135% of the CPU, while our interpolated frequency-domain SRC method requires 5.9% of the CPU. So, our method leads to only a modest increase in CPU load compared to the bandlimited playback, while the exact transform is prohibitively expensive.

Method	CPU Utilization
Exact transform-domain interpolation	135%
Playback signal at 16 kHz, no interpolation	3.9%
Playback signal at 44.1 kHz, MCLT-domain interpolated SRC	5.9%

Table 2: Computational complexity of the proposed MCLT-domain SRC vs. the exact transform method.

4. CONCLUSION

In this paper, we have developed a new frequency domain interpolation architecture which allows AEC to be performed while a full bandwidth signal is played through the speakers. The new architecture allows AEC to be run with many new scenarios including speech recognition, internet gaming, and CD quality music playback. The results indicate that although none of the interpolated windows result in a perfect reconstruction filter bank, overlapping the data backwards in time is preferable to any type of zero padding. The new algorithm is extremely fast and can be implemented in real time. Furthermore, the new algorithm only introduces a small degradation in the ERLE compared to using the exact transform.

REFERENCES

[1] C. Breining et. al., "Acoustic echo control. An application of very-high-order filters", *IEEE Signal Processing Magazine*, vol. 16, pp. 42–69, July 1999.

[2] H. S. Malvar, "A Modulated Complex Lapped Transform and Its Applications to Audio Processing", Proc. *ICASSP*, pp. 1421–1424, March 1999.

[3] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[4] P. J. S. G. Ferreira, "Interpolation in the time and frequency domains," *IEEE Signal Processing Letters*, vol. 3, pp. 176–178, June 1996.

[5] H. S. Malvar, "Fast algorithm for the modulated complex lapped transform", *IEEE Signal Processing Letters*, vol. 10, pp. 8–10, Jan. 2003.

[6] R. E. Blahut, *Fast Algorithms for Digital Signal Processing.* Reading, MA: Addison-Wesley, 1987.

[7] M. M. Sondhi, D. R. Morgan, and J. L. Hall, "Stereophonic acoustic echo cancellation; an overview of the fundamental problem," *IEEE Signal Processing Letters*, vol. 2, pp. 148–151, Aug. 1995.