Committee T1 Performance
Standards Contribution

Document Number:	T1A1.5/96-111
TIBBS File:	6a151110.doc
DATE:	June 3, 1996
STANDARDS PROJECT:	Analog Interface Performance Specifications for Digital Video Teleconferencing/Video Telephony Service (T1Q1-12)
SUBJECT:	Estimation of audio delays
SOURCE:	NTIA/ITS
CONTACT:	Stephen Voran Voice: (303) 497-3839 Fax: (303) 497-5323 e-mail: sv@bldrdoc.gov
KEY WORDS:	Audio Delay, Speech Delay, Delay Estimation
DISTRIBUTION:	Working Group T1A1.5 (announced via t1a15@t1.org)
ABSTRACT:	This contribution presents an algorithm for estimating the delay of telephony band speech. The algorithm features a coarse stage that uses speech envelopes and a fine stage that uses speech power spectral densities. Observations on the performance of the algorithm and its potential for extension beyond telephone band speech are offered.

An Algorithm for Estimating the Delay of Telephony Speech

1. Introduction

This informational contribution contains a description of an algorithm for estimating the delay of telephony band speech. The description is provided to T1A1.5 with the hope that it may be of use in the development of ANSI T1 Standard on Visual Channel Delay and Frame Rate Measurement (T1A1.5/96-101.) Section 2 provides a description of the algorithm, which features a coarse stage that uses speech envelopes, and a fine stage that uses speech power spectral densities. Section 3 offers observations on its performance, and its potential for extension beyond telephone band speech.

2. Description of the Algorithm

2.1 Application and Overview

The algorithm was developed at the Institute for Telecommunication Sciences, (ITS) as a component of the ITS research on perception-based objective audio quality assessment tools. One desired output from this research is a perception-based algorithm that compares the contents of two digital audio files in a way that is consistent with human perception and judgment. Figure 1 shows how these two files are created and where the delay estimation algorithm is used. O of the files contains digital samples of a reference audio signal that went into the audio system under test. The other file contains digital samples of the audio signal as it came out of the audio system under test. These two files are referred to as the reference file and the test file, respectively. Because the audio system under test is a physical device, audio events can emerge from its output only after they have been presented to its input. The time required for the audio output to react to an input audio event is the delay of that audio system. If like segments of the reference and test files are to be compared by an objective audio quality assessment tool, the delay of the audio system under test must be removed before the comparison is made. Before the delay can be removed, it must be estimated. The algorithm described in Section 2 estimates the delay of the contents of the test file relative to the contents of the reference file. It was designed for use on 4 kHz bandwidth speech, sampled at a rate of 8000 samples/second. Its potential for other uses is discussed in Section 3.

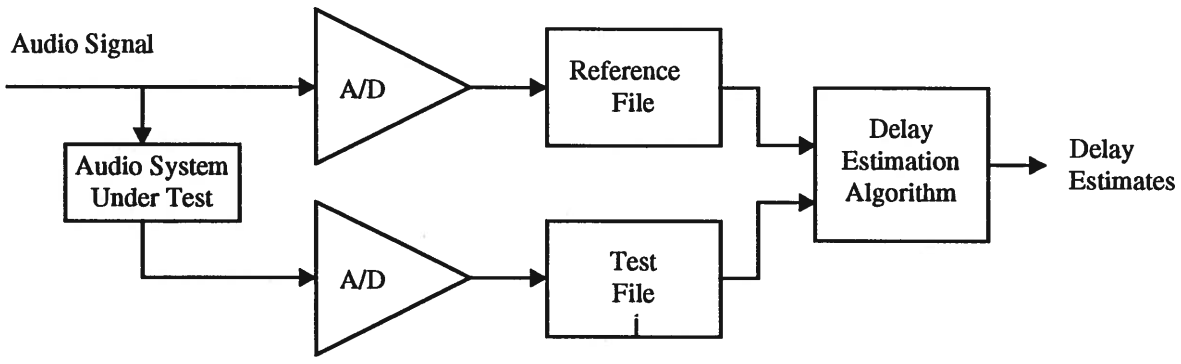

Figure 1. Example application of delay estimation algorithm

The algorithm features a coarse stage that uses speech envelopes, and a fine stage that uses speech power spectral densities. The coarse stage estimates delay to the nearest 4 ms. Whenever possible, the fine stage then refines this estimate to the nearest sample (125 μs). Each stage involves a search over a range of possible delay values. The two stage process is efficient because the coarse stage can search a wide range of delay values, but at low resolution. If that same range were searched at high resolution, many more computations would be required. Once the coarse stage has finished its work, its low-resolution estimate can often be refined to a high resolution estimate by the fine stage that follows. The fine stage needs to search only a narrow range of delay values, consistent with the uncertainty of the coarse estimate.

Many digital speech coders do not preserve the speech waveform. By preserving a higher level statistical description of the speech waveform, but not the speech waveform itself, these devices achieve substantial compression ratios, and still generate output speech that sounds very similar to the input speech. When speech waveforms are not preserved, waveform cross-correlation and other waveform matching techniques will give ambiguous or erroneous delay estimates. For this reason, the coarse stage of the ITS algorithm uses speech envelopes, which are preserved by digital speech coders. Likewise, the fine stage uses speech power spectral densities (psd's) which are usually approximately preserved. For some speech coders and other speech devices, psd's are not adequately preserved and fine estimates are not possible. In other cases, multiple fine estimates will give inconsistent results. This indicates that, in the high resolution view at least, the delay is not constant. In these situations the coarse delay estimate, along with its inherent uncertainty, becomes the total delay estimate.

Finally, note that the timing relationship between the two A/D converters in Figure 1 must be known if an absolute delay value is required. While only positive delays are physically possible, the samples in the test file may lead or lag the samples in the reference file, depending on when the two A/D converters were started. If their starts were synchronized, then the samples in the test file must lag the samples in the reference file.

2.2 Coarse stage

In our present application, delay values ranging from S_n=-.8 to S_p=.8 seconds are possible. When S_n and S_p are referred to in the general sense in the description that follows, we assume that $S_n \leq 0$ and $S_p \geq 0$. The variables S_n , S_p ,and others are collected for convenience in Table 1. We extract L=6 seconds worth of samples (48,000 samples) from identical locations in the reference file and the test file. These are placed in arrays that are named *ref* and *test*. The mean of each array is removed, and one array is scaled so that the two arrays have the same standard deviation. Next, 6400 samples (equivalent to .8 seconds) are removed from the start and end of *test*. In general, one would remove $|S_n| \cdot f_s$ samples from the start of test, and $S_p \cdot f_s$ samples from the end of test, where f_s is the sample rate of the audio signal.

Variable	Description	Value
L	Length of speech segment used	6 seconds
S_n	Range of negative delay values to search	.8 seconds
S_p	Range of positive delay values to search	.8 seconds
D	Down-sampling factor	32
m	Frame size used in fine stage	64 samples
n_1	Number of fine estimates done	6
n_2	First required number of fine estimates	4
n_3	Second required number of fine estimates	3
t_1	Correlation threshold for fine estimates	.707
t_2	Consistency threshold for fine estimates	10 samples

Table 1. Variables and their values for the current ITS application: 4 kHz bandwidth speech sampled at 8000 samples/second

The coarse stage works by cross-correlating speech envelopes. These envelopes are calculated as follows. The signals in *ref* and *test* are rectified by taking the absolute value of each sample. Each array is then low-pass filtered to create envelopes with a bandwidth of $f_s/(2 \cdot D)$. We used D=32, and a seventh order IIR Butterworth filter with a -3 dB point at 125 Hz. The resulting envelopes still have f_s=8000 samples per second but bandwidth only to $f_s/(2 \cdot D)$=125 Hz. By Nyquist's theorem, we can reduce the sample rate by a factor of D without loosing any envelope information. We down-sample by a factor of D by retaining only every D^{th} sample in each envelope. These envelopes are stored in the arrays called *ref_env* and *test_env*. It is this low-pass filtering and subsequent down-sampling that gives the coarse stage its reduced resolution and reduced computational load.

Once the two speech envelopes have been extracted, the mean value is removed from each and a cross-correlation value is calculated for each possible shift of *test_env* relative to *ref_env*:

$$cc_s = \frac{\left(\sum_{i=1}^{M} ref\_env_{i+s} \cdot test\_env_i \right)}{\left(\sum_{i=1}^{M} ref\_env_{i+s}^2 \right)^{\frac{1}{2}} \left(\sum_{i=1}^{M} test\_env_i^2 \right)^{\frac{1}{2}}}$$

$$\text{for } \frac{s_n \cdot f_s}{D} \le s \le \frac{s_p \cdot f_s}{D}, \quad \text{where } M = \frac{\left(L - |s_n| - s_p\right) \cdot f_s}{D}.$$

Equation 1

The array *cc* holds the values of the cross-correlations between the speech envelopes in *ref_env* and *test_env* at every possible shift. These results are then smoothed with a low-pass FIR filter. This symmetric, second-order filter uses the coefficients {.25, .50, .25}. After this smoothing, the largest value in *cc* is taken as an indication of the coarse delay:

$$coarse\_delay = s' \cdot D \text{ samples}$$
$$= \frac{s' \cdot D}{f_s} \text{ seconds,} \qquad \text{where } s' \text{ maximizes } cc_s.$$

Equation 2

We consider the uncertainty in the coarse delay estimate to be $\pm D$. One might argue that the uncertainty is $\pm D/2$, but $\pm D$ is more conservative. Note that the second term in the denominator of Equation 1 is a normalizing constant that is required to get a true cross correlation value between -1 and 1, but does not have any impact on the smoothing and peak-finding that follow Equation 1.

2.3 Fine Stage

In many cases, the uncertainty inherent in the coarse estimate can be reduced substantially by the fine stage of the delay estimation algorithm. In our application, the fine stage is performed at n_1 =6 locations in the original L=6 seconds of the test and reference files. Locations are selected in the reference file, and corresponding locations, offset by the coarse delay estimate calculated in Equation 2, are selected from the test file. At each location $8 \cdot D$ samples are taken from the reference file and stored in an array called *ref* and $2 \cdot D$ samples are taken from the test file and stored in an array called *test*. In both cases, the samples should be nearly symmetric about the selected location. If that location is x for example, samples used in *ref* extend from x-$4 \cdot D$ to x+$4 \cdot D$-1 inclusive, while the samples used in *test* extend from (*x*+*coarse_delay*)-*D* to (*x*+*coarse_delay*)+*D*-1 inclusive.

If the contents of the files are known a priori, the 6 locations may be picked to insure that none of them fall into regions of silence. The fine delay estimation algorithm will not work in silent regions. If the contents of the files are unknown, then random locations can be selected. For each location, *ref* and *test* must be tested to see that neither has a level that is more than 30 dB below the average level of the corresponding file. When a location fails this test, a new location is selected and tested. This process is repeated until 6 locations that pass this level test are found.

The fine stage works by cross-correlating power spectral densities (psd's). The psd's are calculated as follows. The $8 \cdot D$ samples in *ref* are broken into groups of m=64 samples, and these groups are called frames. There are $8 \cdot D - m + 1 = 193$ frames and they are numbered 1 through 193. The first sample of the i^{th} frame is sample number i, and the last sample of the i^{th} frame is sample number $m+i-1$. The $2 \cdot D$ samples in *test* comprise a single, length m=64 frame. Each frame is multiplied by a Hamming window, and then transformed to the frequency domain using a length m=64 FFT. In the frequency domain, only the first $(m/2)+1=33$ complex samples of each frame are unique, so only those samples are saved. The magnitude of each retained sample is taken, resulting in the square root of the power spectral density of each frame. (We ignore the square root and refer to these results as psd's for simplicity.)

The mean value of each psd is then removed and a cross-correlation value is calculated between the test psd and each of the 193 reference psd's:

$$cc_s = \frac{\left(\sum_{i=1}^{\frac{m}{2}+1} ref\_psd[s]_i \cdot test\_psd_i \right)}{\left(\sum_{i=1}^{\frac{m}{2}+1} ref\_psd[s]_i^2 \right)^{\frac{1}{2}} \left(\sum_{i=1}^{\frac{m}{2}+1} test\_psd_i^2 \right)^{\frac{1}{2}}}$$

where $ref\_psd[s]_i$ is the i^{th} sample of the s^{th} reference psd, $1 \le s \le 8 \cdot D - m + 1$,

and $test\_psd_i$ is the i^{th} sample of the test psd.

Equation 3

The array *cc* now holds the values of the cross-correlations between the reference and test psd's at each time-domain shift. The largest value in *cc* is taken as an indication of the fine delay:

$$fine\_delay = s' - (3 \cdot D + 1) \text{ samples},$$
$$= \frac{s' - (3 \cdot D + 1)}{f_s} \text{ seconds}, \quad \text{where } s' \text{ maximizes } cc_{s'}.$$

Equation 4

This procedure is repeated for each of the 6 locations. The result is 6 estimates of the fine delay, and 6 corresponding cross-correlation values. Note that each of the fine delay estimates must fall between $-3 \cdot D = -96$ and $5 \cdot D - m = 96$, inclusive. Also note that the second term in the denominator of Equation 3 is a normalizing constant that is required to get a true cross-correlation value between -1 and 1, but does not have any impact on the peak-finding in Equation 4.

Once the 6 fine delay estimates and corresponding cross-correlation values have been calculated, they are filtered by a simple algorithm. First we determine which of the 6 results are informative. Each of the 6 cross-correlation values is tested and only those greater that $t_1 = .707$ are retained. By this process, only cross-correlations that explain at least half the variance of the psd in *test_psd* are retained.

Next we determine which of the informative fine delay estimates are also consistent with the coarse delay estimate. Since the uncertainty in the coarse delay estimate is $\pm D$ samples, and the coarse delay has been removed, only fine delay estimates between $-D$ and D samples are retained. If there are fewer than $n_2 = 4$ fine delay estimates remaining after these two tests, the fine stage terminates and final delay estimate is simply the coarse delay estimate, with its uncertainty of $\pm D$ samples.

In the event that $n_2 = 4$ or more fine delay estimates remain, we next look for consistency among those fine delay estimates. This is done searching for the largest sub-set of fine delay estimates $\{fine\_delay_i\}$ that are no more than $t_2 = 10$ samples from each other:

$$\max_i\{fine\_delay_i\} - \min_i\{fine\_delay_i\} \le t_2.$$

Equation 5

If the largest such set of fine delay estimates contains fewer than $n_3 = 3$ fine delay estimates the fine stage terminates and final delay estimate is simply the coarse delay estimate, with its uncertainty of $\pm D$ samples. If 3 or more fine delay estimates are in the set defined by Equation 5, then the mean of those fine delay estimates is added to the coarse delay estimate to create the final delay estimate. The standard deviation of the surviving fine delay estimates

might be used to derive an uncertainty value for the final delay estimate. This would only be meaningful if n_3 were significantly larger than the current value of 3.

3. Observations on the Algorithm

Our experience with this algorithm is limited to 4 kHz bandwidth speech with a sample rate of f_s=8000 samples/second, and all variables set as indicated in Table 1. A simple experiment shows that, under laboratory conditions, the delay estimation algorithm performs as we would expect. This experiment uses 30 speech files, each containing a distinct English language sentence pair. Three female and three male talkers were used, and each spoke five sentence pairs.

The 30 files were passed through the Modulated Noise Reference Unit (MNRU) at 6 different settings. The MNRU is a reference condition that adds amplitude correlated noise to a speech signal[1]. This allows the MNRU to simulate the quantization noise that is produced by many waveform coders. The MNRU has a single control variable called Q, which essentially specifies the SNR of the output speech in dB. Our experiment uses Q = 0, 10, 20, 30, 40, and 50. Table 2 shows the mean and standard deviation (calculated across the 30 files) of the total estimated delay through the MNRU as a function of Q. Table 1 also indicates the percentage of the 30 files for which the fine stage was able to produce a useable delay estimate.

MNRU Setting	Mean of Estimated Delays	Standard Deviation of Estimated Delays	Percentage of Files with Useable Fine Estimate
Q = 0 dB	0.2 Samples	0.9 Samples	3%
10	1.1	1.6	97
20	1.5	0.7	100
30	1.6	0.6	100
40	1.7	0.7	100
50	1.7	0.6	100

Table 2. Delay Estimation Results for MNRU

Table 2 shows that when the speech waveform is severely distorted (Q=0 dB) the fine stage rarely produces a useable estimate. When the speech waveform is less distorted ($Q \geq 20$ dB) the fine stage always produces a useable estimate, and the standard deviations of the delay estimates are smaller than when the waveform is more distorted. Also, as the speech waveform becomes less distorted, the mean value of the estimated delay converges to 1.7 samples. This delay is caused by a filter inside the MNRU.

The 30 files were also passed through the T-Reference Condition (T-Ref) at 6 different settings. The T-Ref is a reference condition that removes and interpolates samples of digital speech signals to create a form of frequency modulation plus aliasing that can sound similar to some lower bit-rate coders[2]. The T-Ref has a single control variable called T, and smaller values of T correspond to larger amounts of distortion in the output speech signal. Our experiment uses $T = 2, 4, 8, 16, 32$, and 64. The theoretical, time-averaged delay of our software T-Ref implementation is $-\frac{2}{3}\cdot\frac{256}{T}$. Table 3 includes this theoretical delay value, as well as the mean and standard deviation (calculated across the 30 files) of the total estimated delay, and the percentage of the 30 files for which the fine stage was able to produce a useable delay estimate.

T-Ref Setting	Theoretical Average Delay	Mean of Estimated Delays	Standard Deviation of Estimated Delays	Percentage of Files with Useable Fine Estimate
T = 2	-85.3 Samples	-87.5 Samples	18.7 Samples	0%
4	-42.7	-37.3	12.1	7
8	-21.3	-30.9	5.9	33
16	-10.7	-8.2	7.9	53
32	-5.3	-5.8	2.6	87
64	-2.7	-3.0	0.9	97

Table 3 Delay Estimation Results for T-Reference

Table 3 reveals the following. The mean of the delay estimates track the theoretical delay values. The ability of the fine stage to produce a useable result increases as the waveforms become less distorted. The standard deviation of the delay estimates decreases as the waveforms become less distorted. Since the uncertainty of the coarse stage is ± 32 samples, the difference between the mean of the delay estimates and the theoretical delay value is larger when the coarse stage operates alone, and is smaller when when the fine stage provides a refinement.

The algorithm also performs as we would expect under non-laboratory conditions. We have used the algorithm to estimate the delays of over 10,000 test speech files. These files include sentences and sentence pairs, and cover over 100 distinct combinations of speech coders and network conditions or reference conditions, 36 talkers and 3 languages. The fine stage of the algorithm produces a useful estimate all of the time when Adaptive Differential Pulse Code Modulation (ADPCM) coders are tested, some of the time when Vector Sum Excited Linear Predictive (VSELP) coders are tested, and never when Sinusoidal Transform Coders (STC's) are tested. The coarse stage always produces a useful estimate. Using either a priori information on coder delays, or a pair-wise listening technique, we have verified that the total

delay estimates produced by the algorithm are correct, within the uncertainties inherent in the test files.

Preliminary time-trials for a typical telephony band application were also conducted. When f_s =8000 samples/second, and L≈6 seconds, we used the algorithm to search for positive delays up to 500 ms. (S_n=0 and S_p=.5 seconds). Using non-optimized, interpreted code on a 133 MHz P5, we found that each delay estimate could be computed in 6 seconds. In other words, this version of the algorithm runs in real-time. We have describe an algorithm that generates a single delay estimate for a pair of files that were completely digitized before the algorithm started. It is clear that this algorithm could also be implemented in an "on-line" or "live-data" fashion. Such an implementation would continuously read two streams of audio samples and generate a continuing stream of delay estimates.

We would expect this algorithm to perform similarly on wideband telephony speech (7 kHz bandwidth, f_s=16,000 samples/second.) We would expect performance to be much worse on more general audio signals. In particular, music can exhibit periodicities at many different time scales, ranging from milliseconds to 10's of seconds. These intrinsic periodicities can make delay estimation an extremely challenging task. A thorough study of a wide range of general audio signals might yield a set of values for the variables in Table 1 that would give this algorithm maximal robustness against this intrinsic challenge.

4. Summary

We have presented a description of an algorithm for estimating the delay of telephony band speech. The algorithm features a coarse stage that uses speech envelopes, and a fine stage that uses speech power spectral densities. This approach is motivated by the need for delay estimates for audio and speech devices that do not preserve waveforms. The algorithm behaves as desired for speech signals that have been distorted by the MNRU, the T-Ref, and combinations of real speech coders and networks. The coarse stage of the algorithm always produces a useful estimate with an uncertainty of ± 32 samples (4 ms). The fine stage of the algorithm produces a refinement to that estimate whenever sufficient spectral detail is preserved to make such a refinement meaningful. We hope that the information provided here will be helpful to T1A1.5 in the development of ANSI T1 Standard on Visual Channel Delay and Frame Rate Measurement (T1A1.5/96-101.)

References

[1] ITU-T Recommendation P.810, "Modulated Noise Reference Unit," Geneva 1995.

[2] B. Cotton, "New Reference Condition for Very Low Bit Rate Voice Coder Evaluation" Proc. IEEE Global Telecommunications Conference, Orlando, 1992.