SKEW DETECTION AND COMPENSATION FOR INTERNET AUDIO APPLICATIONS

Orion Hodson, Colin Perkins, and Vicky Hardman

Department of Computer Science
University College London
Gower Street, London, WC1E 6BT, UK.

ABSTRACT

Long lived audio streams, such as music broadcasts, and small differences in clock rates lead to buffer underflow or overflow events in receiving applications that manifest themselves as audible interruptions. We present a low complexity algorithm for detecting clock skew in network audio applications that function with local clocks and in the absence of a synchronization mechanism. A companion algorithm to perform skew compensation is also presented. The compensation algorithm utilises the temporal redundancy inherent in audio streams to make inaudible playout adjustments. Both algorithms have been implemented in a simulator and in a network audio application. They perform effectively over the range of observed clock rate differences and beyond.

1. INTRODUCTION

When sampling an audio signal for digital transmission, the sample clock will differ from its nominal rate due to variations in the quartz crystal oscillators and the components that regulate their frequency. During the implementation of a network audio system [3] for workstations and personal computers we observed $\pm 0.5\%$ variation between nominally similar clocks. This has undesirable effects, since when the sender's clock is faster than the the receiver's, samples will accumulate; consuming memory and increasing the delay. As the memory available for the audio buffer is exhausted interruptions occur as frames are dropped from stream. Conversely, when the sender's clock is slower than that of the receiver, audio played out at the receiver becomes interrupted as the playout buffer runs dry.

We present two algorithms: one for detecting clock skew, and one for compensating for its effects. Due to the presence of loss and jitter in the packet arrival process, it is not sufficient to merely observe playout buffer occupancy to detect clock skew. Instead, obervation of the packet arrival process is necessary, making the detection of clock skew a non-trivial problem. Our compensation algorithm uses a novel low complexity pattern matching algorithm, to identify periods where adjustments may be performed.

Existing research on Internet audio applications has focused on issues arising from the best effort nature of network, such as loss concealment and forward error correction [9], and the calculation of suitable playout points in the absence of synchronization mechanisms [7, 10]. Work has also been undertaken on the detection of clock skew between hosts [6, 8], though this is the first work we are aware of that addresses the issues of transparent skew adjustment for network audio applications.

This paper is structured as follows: section 2 concerns the detection of clock skew in unsynchronized audio applications; section 3 describes the compensation algorithm to be applied when skew is perceived and it's performance on voice, popular, and classical music streams; section 4 presents some preliminary results illustrating the performance of these algorithms. A section on implementation issues concludes the contributions of this work. Finally, we summarize and discuss potential future work.

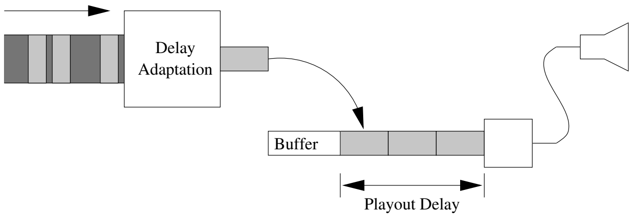

Figure 1: Playout buffering in a network audio application (from [5]).

2. SKEW DETECTION ALGORITHM

In systems with unsynchronized clocks, such as RTP [12] audio applications, an offset must be added to each arriving packet to map it from its source's time into local time. An additional delay, the playout delay, is added to compensate for arrival time variation (jitter) and for variations in host scheduling [4]. The playout delay and mapping offset are usually held constant over the duration of the packet stream to avoid interruption.

Detection of clock skew can be performed by maintaining a running estimate of the mapping offset, $\hat{m}_i$, and looking for divergence from the mapping offset assigned to the current packet stream, $\hat{m}_{Active}$. Assuming the i-th packet has a timestamp indicating its source time, s_i, and the receiver records its arrival time, a_i, these variables are calculated using:

$$m_i = a_i - s_i \quad (1)$$
$$\hat{m}_i = \alpha \hat{m}_{i-1} + (1-\alpha) m_i \quad (2)$$

with the initial $\hat{m}_{Active}$ being $\hat{m}_i|_{i=0}$. When $\hat{m}_i$ and $\hat{m}_{Active}$ have diverged sufficiently compensating action must be taken. We denote the divergence as $\delta = \hat{m}_{Active} - \hat{m}_i$.

Each value of m_i calculated includes a component attributable to the variable transit time between the sender and receiver. The responsiveness to these variations is determined by the parameter α in equation 2. There is a trade-off between being resilient to short term fluctuations and the delay introduced by the exponential weighting process. We have found by experimentation a value of

$\frac{31}{32}$ to represent a good compromise value with packet sizes of 20, 40, and 80ms.

It is desirable to limit the number of compensating actions to keep the computational cost low and to reduce sensitivity to the transient transit variations. High and low water marks, δ_L and δ_H, are therefore employed. The placement of the water marks is influenced by implementation issues that are deferred until section 5. We therefore arrive at the following formulation:

SAMPLESTOCORRECT($\hat{m}_{Active}$,$\hat{m}_i$)
$\delta \leftarrow \hat{m}_{Active} - \hat{m}_i$
if $\delta < \delta_L$ **or** $\delta_i > \delta_H$ **then**
    return δ
return 0

The returned value is passed to the skew compensation algorithm as the upper limit on the number of samples to add or remove from the stream. A positive divergence value indicates the source's clock is faster than the receiver and δ samples should be deleted from the stream. Conversely, a negative divergence value indicates the source's relatively slow clock rate and δ samples need to be inserted. When the compensation algorithm, in the next section, makes adjustments it changes the value of $\hat{m}_{Active}$ accordingly.

This algorithm suffices for symmetric distributions of transit variations. In reality, outliers exist in the distribution of transit times, particularly those arising from packet compression events [1] that adversely influence the mapping offset. An additional mechanism, derived from Ramjee et al's algorithm 4 [10], is therefore employed to identify periods of packet compression. During these periods the estimate of the mapping offset is not updated. Compression periods are typically of the order of 1 second in duration; a period short enough not to have repercussions at the observed skew rates.

The short-term memory of offset estimate means that variations in the mean end-to-end delay are indistinguishable from clock skew. In contrast to previous work [6, 8], our algorithm is not a robust skew estimator. Our goal is to maintain the buffer occupancy within a constrained region. Both clock skew and slow evolutions of the mean transit time manifest themselves as changes in the buffer occupancy, and it is these that the compensation algorithm caters for. Our estimator comes at a reduced cost, $O(1)$ compared to $O(n)$, owing to the simpler goal.

3. COMPENSATION ALGORITHM

When the skew detection algorithm indicates that the number of samples in the receiver's playout buffer requires adjustment, a compensation mechanism is applied. We have experimented with several schemes, and the mechanism we present here represents the most successful. Potentially simpler schemes, such as regular (and irregular) sample insertion and deletion, introduce audible distortion that is attributable to phase discontinuity.

The approach we adopt utilizes the temporal redundancy inherent in audio signals to identify segments of audio that may be repeated or cut from the stream to adjust the playout buffer occupancy. A simple and low-cost heuristic is used to locate repetitive segments within a frame:

$$E_i(t) = \frac{1}{L} \sum_{k=0}^{L-1} |s_i(w+k) - s_i(m+t)| \qquad (3)$$

where $s_i(t)$ represents the amplitude of sample t in frame i, w is position of the match window start, L is the comparison window width, and m position in the frame where the best match commences. match commences. When the smallest value of $E_i(t)$ for a frame is below a threshold, T, the portion of audio between the window and the match location is deemed an acceptable for repeating or cropping. The parameters T and L are chosen to be 1200 (using a 16-bit linear representation) and 8 samples respectively after aural testing. The match window and region searched do not overlap. The matching process is illustrated in figure 2.

Earlier work on repairing missing segments from packetized audio streams [2, 11, 13] scale the samples between the window and the search region. The heuristic we present selects only those segments with a relatively stationary signal gain. Whilst this reduces the number of segments available for cropping and repeating compared the earlier work cited, no costly gain scaling is required to mask the insertion and deletion operations.

The location of comparison window depends on whether a segment is to be inserted or deleted. When a segment is to be deleted the match window begins at the start of the frame. Conversely, when samples are to be inserted the match window is placed at the end of the frame. The insertion or deletion is applied and the transition is masked using a linear blending function. Frames in the playout buffer after the adjustment point have their playout time shifted to maintain continuity. The mapping offset used, $\hat{m}_{Active}$, is updated and subsequent packets have the updated offset applied. The window locations and operations are depicted in figure 3.

4. RESULTS

To evaluate the effectiveness of the compensation algorithm we applied artificial clock skew rates of $\pm0.5\%$, $\pm2\%$, and $\pm5\%$ relative to the intended playback rate to three sample streams comprising of voice, popular music, and classical music (described in appendix A). The skew compensated and original samples were then played through a high quality headset to 10 listeners. Skew compensation adjustments of $\pm0.5\%$ and $\pm2\%$ were not detected by any of the listeners. At $\pm5\%$ compensating adjustments were apparent to all listeners of the classical music piece. However, at $\pm5\%$ none of the listeners noticed the adjustments to the pop music track and only one listener commented on the brevity of the pauses between speech segments in the voice track.

The compensating algorithm has difficulty with audio sources containing prolonged periods without stationary repetitive segments. Sources with rich and wide ranging harmonic content, such as classical music, exemplify the problem. In the periods without stationary repetitive segments the number of samples above or below the threshold increases. When a passage with a stationary period arises many adjustments occur in close succession introducing a "warbling" effect. Sources like voice that have frequent low energy components and a high fraction of stationary repetitive segments are able to have adjustments made more frequently. A comparison of the number of samples in excess when adjustments are made for a faster source is shown in figure 4. The distributions of the size of adjustments (not shown) are near identical for each audio stream type, the adjustment sizes are evenly distributed across the available range.

We have also implemented the algorithm in an RTP audio application [3] and monitored it's behaviour with multicast sessions of terrestrial radio station programming and unicast tests between UCL and other institutions. We have observed skew rates of

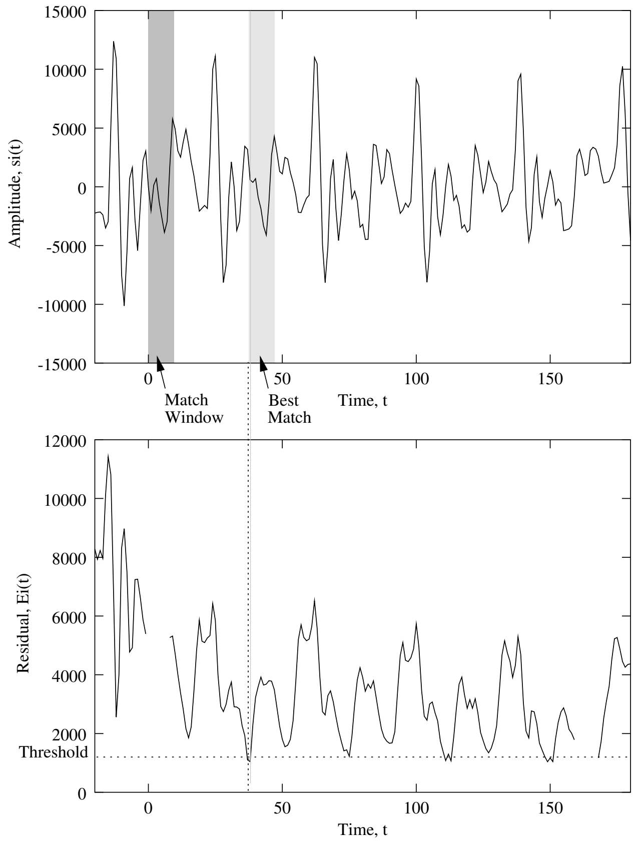

Figure 2: Identification of best matching audio segment within an audio frame of the voice test sample. The audio signal is depicted in the top graph together with the match window and the best matching segment. The residual is depicted lower graph.

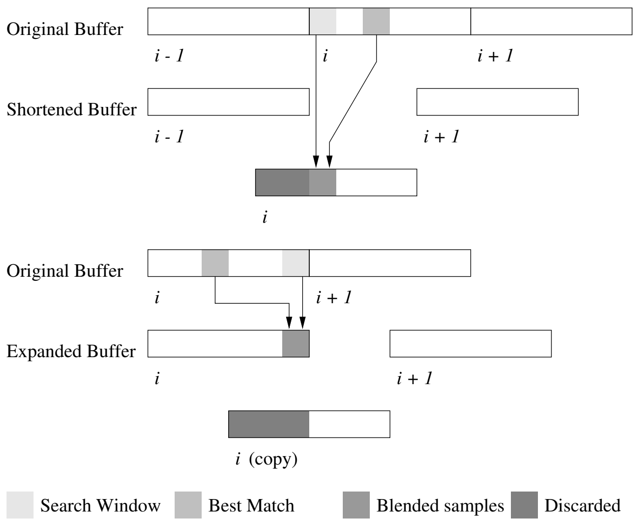

Figure 3: Deletion and insertion of repeated segments in the playout buffer. The arrows indicate audio segments that are blended to conceal insertion or deletion operations.

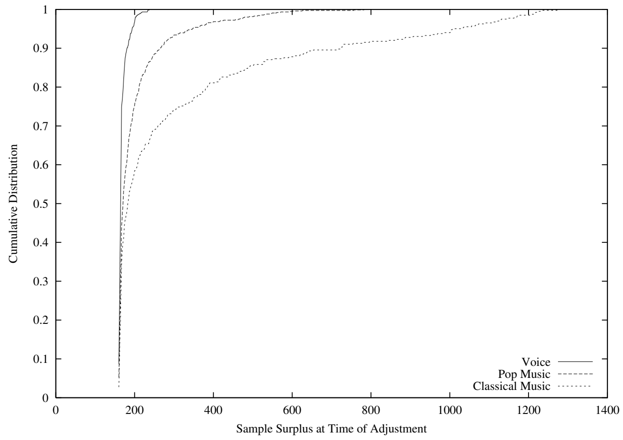

Figure 4: The cumulative distribution of the number of samples over those expected, from a faster clocked audio source source, before an adjustment is possible.

$\pm 0.5\%$ and have not observed any ill effects arising from compensating actions.

5. IMPLEMENTATION CONSIDERATIONS

We now consider the practical issues that implementing the detection and compensation algorithm present. The key issue is the selection of the low and high water marks that are used to determine an adjustment is necessary.

The low water mark is used to trigger sample insertion when the source is deemed to have a relatively slow clock. It is desirable to trigger sample insertion before the application runs out of audio to play, and has to invoke a last-chance loss concealment mechanism or playout silence. We have found that both concealment and silence substitution perform poorly when compared to the skew compensation algorithm presented. If it is not possible to perform skew compensation, loss concealment is potentially less damaging than silence substitution though its performance is heavily dependent on the signal content.

The low water mark is therefore determined by how many samples can be deficient before starvation effects occur. Our present implementation uses a fraction of the playout delay incurred due to network effects[1], but this is an ad-hoc measure, and further work is needed to refine its performance.

As a refinement, when it is determined that a source's clock is consistently slow, a receiver may add additional delay beyond the minimum necessary to ensure correct playout. This prevents interruption during periods that the skew compensation algorithm cannot function due to the lack of stationary segments in the audio stream.

The criteria for the high water mark is dependent on how much delay and memory consumption the receiver is prepared to tolerate. The high water mark needs to be placed a good distance from the low; sufficiently far that oscillations do not occur between positive and negative playout adjustments. Our implementation presently uses a fixed value of 200ms which is sufficiently large to

[1] As opposed to the delay introduced to compensate for scheduling variation in the host system [4].

handle observed network transit delay variations, and short enough that the additional playout delay which may be incurred is not an issue for interactive sessions.

6. SUMMARY AND FUTURE WORK

We have presented a low complexity algorithm to detect clock skew between remote audio applications, and an algorithm to compensate by adding or removing stationary audio segments. We have found these to work well in simulation and in a real application.

Our detection algorithm is affected by slow variations in end-to-end delay, such as those arising from demand driven variations in router queue lengths. These changes manifest themselves in a similar manner to clock skew, and our compensation algorithm also adapts to these changes. In practice, we have the found combination of the two algorithms to work well.

We believe our compensation algorithm could also be used to effect small changes in playout point, if needed. Applications are typically conservative and choose a large initial playout delay, since they have insufficient information about the transit time variation to be more aggresive. The compensation algorithm presented could make adjustments to the playout point for those streams which do not have natural breaks.

The use of stationary segments for the compensation is both a strength and weakness of this approach. It enables inaudible adjustments but at a cost of delay when a suitable segments are not present in the audio stream. In future work we intend to compare the compensation scheme with a resampling algorithm that may conceal phase shifts better than individual sample insertion and deletion schemes.

7. ACKNOWLEDGEMENTS

We thank the members of our research group who participated in the listening tests used to assess the effectiveness of the compensation algorithm. We are indebted to Jim Gemmell, Mark Handley, Isidor Kouvelas for feedback on this work, and to Mark Handley and Jitendra Padhye for accounts on their computing facilities.

Funding from British Telecommunications plc (ML72254) and the European Commission Telematics for Research project (RE4007) facilitated this work.

A. AUDIO SAMPLES

Two minute audio samples of the of the following pieces were used in evaluating the algorithms presented:

- "You've got to build bypasses", excerpt from Hitch-Hiker's Guide to the Galaxy, Douglas Adams, BBC Worldwide Ltd, 1978.
- "Imaginary Friends", from Dizzy Heights, The Lightning Seeds, Sony Music Entertainment, 1996.
- "Brandenburg Concerto No. 4 in B flat major", Johan Sebestian Bach, recording of the English Chamber Orchestra / Benjamin Britten, Decca Record Company, 1969.

B. REFERENCES

[1] Jean-Chrysostome Bolot. Characterizing end-to-end packet delay and loss in the Internet. *Journal of High Speed Networks*, 2(3):305–323, 1993.

[2] David J. Goodman, Gordon B. Lockhart, Ondria J. Wasem, and Wai-Choong Wong. Waveform substitution techniques for recovering missing speech segments in packet voice communications. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-34(6):1440 – 1448, December 1986.

[3] Orion Hodson and Colin Perkins. Robust audio tool (RAT) version 4. Software available online at http://www-mice.cs.ucl.ac.uk/multimedia/software/rat-4.0, December 1999.

[4] Isidor Kouvelas and Vicky Hardman. Overcoming workstation scheduling problems in a real-time audio tool. In *Proc. of Usenix Winter Conference*, Anaheim, California, January 1997.

[5] Isidor Kouvelas, Vicky Hardman, and Anna Watson. Lip synchronisation for use over the internet: Analysis and implementation. In *Proceedings of the IEEE Conference on Global Communications (GLOBECOM)*, London, England, November 1996.

[6] Sue Moon, Paul Skelly, and Don Towsley. Estimation and removal of clock skew from network delay measurements. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, New York, March 1999.

[7] Sue B. Moon, Jim Kurose, and Don Towsley. Packet audio playout delay adjustment: performance bounds and algorithms. *ACM/Springer Multimedia Systems*, 5(1):17–28, January 1998.

[8] Vern Paxson. On calibrating measurements of packet transit times. In *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pages 11–21, Madison, Wisconsin, June 1998.

[9] Colin Perkins, Orion Hodson, and Vicky Hardman. A survey of packet loss recovery techniques for streaming audio. *IEEE Network*, 12(5):40–48, September 1998.

[10] Ramachandran Ramjee, Jim Kurose, Don Towsley, and Henning Schulzrinne. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, pages 680–688, Toronto, Canada, June 1994. IEEE Computer Society Press, Los Alamitos, California.

[11] Henning Sanneck, Alexander Stenger, Khaled Ben Younes, and Bernd Girod. A new technique for audio packet loss concealment. In Jon Crowcroft and Henning Schulzrinne, editors, *Proceedings of Global Internet*, pages 48–52, London, England, November 1996. IEEE.

[12] Henning Schulzrinne, Steve Casner, Ron Frederick, and Van Jacobson. RTP: a transport protocol for real-time applications. Request for Comments (Proposed Standard) 1889, Internet Engineering Task Force, January 1996.

[13] Ondria J. Wasem, David J. Goodman, Charles A. Dvorak, and Howard G. Page. The effect of waveform substitution on the quality of PCM packet communications. *IEEE*

Transactions on Acoustics, Speech, and Signal Processing, 36(3):342–348, March 1988.