# ADAPTIVE DELAY CONCEALMENT FOR INTERNET VOICE APPLICATIONS WITH PACKET-BASED TIME-SCALE MODIFICATION

*Fang Liu, JongWon Kim, C.-C. Jay Kuo*

Integrated Media Systems Center and Dept. of Electrical Engineering - Systems
University of Southern California, Los Angeles, CA 90089-2564

## ABSTRACT

Much effort has been involved in packet-level error control and delay jitter concealment for Internet audio applications. A packet-based time-scale modification scheme for speech signals is developed to provide adaptive delay concealment at the receiver for an Internet voice session in this work. The adaptive playout algorithm strives to minimize packet droppings for late-arrival packets and premature packets while maintaining the end-to-end delay constraint. By stretching the length of voice segments and incorporating silence intervals, the proposed algorithm is able to accommodate fluctuating delays including delay spikes quickly. It is verified by experiments that the proposed adaptive playout algorithm improves the received speech intelligence under a tightly bounded average playout delay.

## 1. INTRODUCTION

Research on Internet audio streaming has focused on error control and delay concealment in the presence of delay jitter and packet loss recently [1, 2, 4, 6]. Given a fixed receiver buffer and a tight end-to-end delay bound, some packets sent to the receiver may still be discarded since the receiver buffer is adjusted to accommodate the average end-to-end delay. For example, late arrival packets, which arrive after its scheduled playout time, are discarded. Sometimes, a packet arrives too early for its scheduled playout time. The premature packet has to be discarded since there is no place to hold it in the buffer. A delay spike happens when several consecutive packets arrive at the receiver almost simultaneously. This happens when audio packets pile behind a large Internet load [3]. Packet droppings at the receiver caused by network delay jitter are added to increase the network packet loss rate, and thus result in degradation of audio/speech playout.

To recover from network packet loss, redundant Forward Error Correction (FEC) packets [4] and time-domain stretching [6] were considered before. Another approach is to adaptively adjust the silence length between talk spurts to reduce the jitter/loss effect [1, 2]. Most late-arriving packets can be salvaged instead of being thrown away. The adaptive playout based solely on the silence interval could work, if the network is relatively stable (which means the statistical estimation based on the previous talk-spurt could hold for the current) and the employed silence detection is effective. However, since the algorithm in [1] only uses the first packet within a talk-spurt to adapt to the delay, it is not effective if a delay spike happens in the middle of a talk spurt. The algorithm has to wait until the next spurt. Thus, the performance result depends on audio/speech contents and the network situation.

In this work, we extend the silence interval-based adaptive playout algorithm by exploiting the time-scale modification scheme. By applying a varying degree of stretching for each packet (although it is important to maintain the stretching factor within a talk-spurt), every packet could contribute in adapting to the network delay jitter/spike as well as packet loss. For time-scale modification, the synchronized overlap-and-add (SOLA) scheme is adopted in our work and modified into a packet-based version. The time-scale modification factor of each packet will be estimated for each packet depending on the delay constraint, delay statistics, and the number of late-arrival packets. This estimated stretching is then bounded by certain upper/lower bounds that are calculated based on speech contents. Finally, the performance of the proposed adaptive playout algorithm is demonstrated via an end-to-end evaluation with a modeled packet delay/loss behavior of the Internet.

## 2. PROPOSED ADAPTIVE PLAYOUT FRAMEWORK

The proposed adaptive playout framework that employs the fixed-interval packet-based (i.e. frame-based) speech is illustrated in Fig. 1. Chunks of voice inputs are generated at the sender for each fixed-size interval (e.g. 20ms) and then relayed to a content analysis block, where the short-time energy $E_n(i)$ and the zero crossing rate (ZCR) $Z_n(i)$ are calculated. With these, the sender classifies the partitioned input speech into several categories such as silence, transient and general segments. The classification rule adopted is as follows. If $E_n(i) < \overline{E_n(i)}/8$ and $Z_n(i) < 30$, this segment is classified as the silence segment, where $\overline{E_n(i)} = \gamma \underline{E_n(i-1)} + (1-\gamma) E_n(i)$ represents a weighted average of $E_n(i)$. Note that parameter $\gamma$ is introduced to give more weight to the most recent $E_n$. If $E_n(i)/E_n(i-1) > 1.6$ and $E_n(i) > \overline{E_n(i)} \times 2$, or $E_n(i)/E_n(i-1) < 1/1.6$ and $E_n(i-1) > \overline{E_n(i-1)} \times 2$, it is classified into a transient segment. Otherwise, it is classified to a general segment.

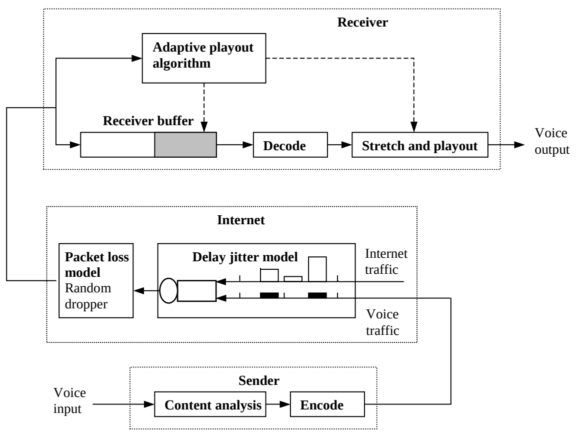


**Fig. 1**. The proposed adaptive playout framework.

Fig. 2 shows the timings associated with packet $i$. After encoding and packetization along with the content classification information, packet $i$ is sent out to the Internet at transmission time $t_i$. The silence part may be taken out from transmission. Then, a transmitted packet experiences network delay $n_i$, which consists of a fixed propagation delay $D_{prop}$ and the network queuing delay $v_i$. It arrives at the receiver with arrival time $a_i$ and waits for its processing with scheduling time $q_i$. It then goes through the processing delay $D_{calc} = D_{dec} + D_{TM}$, which includes decoding time $D_{dec}$ and processing time $D_{TM}$ for time-scale modification. The processing delay is assumed to be constant. The end-to-end delay for packet $i$ is denoted by $d_i$. The varying part of the end-to-end delay is denoted by $b_i$, which is the sum of the receiver buffer delay ($q_i$) and the network queueing delay ($v_i$). Finally, packet $i$ is played out at playout time $p_i$. Its original playout duration (i.e. without time-scale modification) and modified playout duration (i.e. with time-scale modification) are denoted by $l_i^{(O)}$ and $l_i^{(P)}$, respectively.

Let us examine typical models for the Internet traffic. As shown in the network traffic part of Fig. 1, we employ the FIFO queuing model as described in [3, 4]. In this work, we consider two categories of Internet traffic, i.e. light and heavy Internet traffics. The light Internet traffic corresponds to the situation given in [3] when packets almost never accumulate. In contrast, under heavy traffic, packets are piled up and experience consecutive packet delays called the delay spike. In Fig. 3, we show the delay correlation of consecutive packets of simulated heavy Internet traffic, where

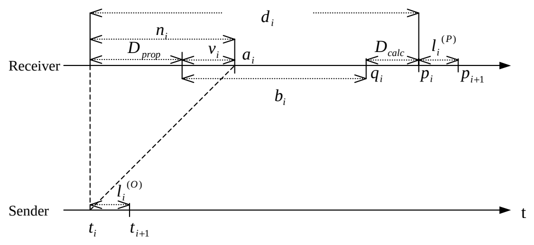


**Fig. 2**. Timings associated with packet $i$.

the horizontal and vertical axes represent the network queuing delays for packets $i$ and $i+1$, respectively. The high correlation in delays is expected under heavy traffic, since if $n_i$ is high, it is very likely that $n_{i+1}$ is also high. But, even if $n_i$ is low, $n_{i+1}$ can still be high. However, delay jitter statistics may not have strong correlation with delay loss [1]. Thus, we use a separate random dropper to emulate the packet loss effect of a certain percentage.

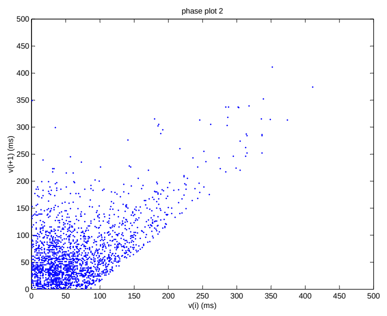
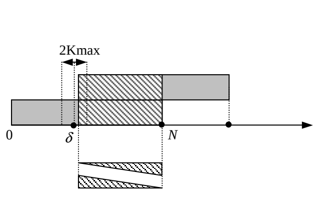


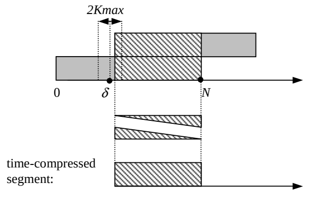
**Fig. 3**. Illustration of delay correlation of consecutive packets under heavy traffic.

## 3. TIME-SCALE MODIFICATION WITH MODIFIED SOLA

Let $x(n)$ be the original waveform and $y(n)$ the time-scale modified waveform with conventional SOLA (Synchronized OverLap-and-Add). Given a fixed-interval overlap frame size of $N$, the analysis segment of size $S_a$ ($S_a < N$) and the synthesis segment of size $S_s$ ($S_s < N$) are determined according to the time-scale modification factor $\alpha = S_s/S_a$. However, when applying SOLA to packetized speech, we may need SOLA to work on a packet-by-packet basis in some extreme cases. The packet-level SOLA leads to a degenerated version of SOLA with $S_a = 0$ and also results in more abrupt variation in time-scale modification. As a result, the stretching factor should be more conservatively selected (e.g. $50 \sim 150$ %) so that it does not deteriorate audio quality too much. If the playout length of each packet is $N$ (i.e., $l_i^{(O)}$), the time-scale modification task is performed as shown in Fig. 4.



(a) Time expansion



(b) Time contraction

**Fig. 4**. Time-scale modification with packet-based SOLA.

In SOLA, a local similarity search process is required, and normalized cross-correlation is often calculated to determine the similarity. In this work, a computationally efficient measure, called the short-time average magnitude difference (SAMD), is used for local similarity search. It is observed that SAMD just introduces a slightly higher mean average error (MAE) than the normalized cross-correlation.

## 4. ADAPTIVE PLAYOUT WITH TIME-SCALE MODIFICATION

To perform adaptive playout control, we employ a sliding-window approach for delay estimation of a packet under the dynamically changing network situation. This estimated delay is then utilized for the playout time calculation of packet $\hat{p}_{i+1}$. Basically, we borrow the estimation method given in Algorithm 4 in [1] or Algorithm 2 given in [2], which is referred to as the reference algorithm here. There are however several notation changes. For example, by considering the involved receiver processing time $D_{calc}$ compared to the PCM audio playback case of [1], $p_i$ in [1] actually corresponds to $q_i$ in our case. Also, since the reference algorithm concerns only the varying part of the network delay, $n_i$ and $d_i$ there correspond to $v_i$ and $b_i$ in our case, respectively. Thus, the notations $\hat{d}_i$ and $\hat{v}_i$ in [1] correspond to $\hat{b}_i$ and $\widehat{vb}_i$ here.

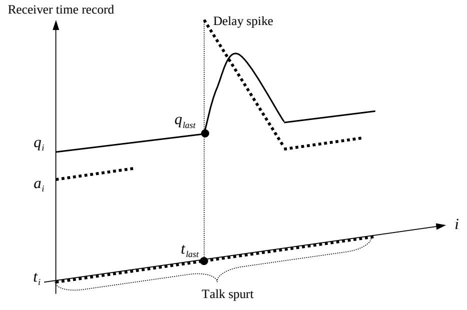


**Fig. 5**. Illustration of a delay spike occurring in the middle of a talk-spurt.

The reference algorithm estimates the end-to-end delay differently in cases of normal delay jitter (called the 'normal' mode) and delay spike (called the 'spike' mode). The goal is to detect a delay spike as soon as possible when it happens and traces the spike slope as shown in Fig. 5. For each received packet, we calculate $\hat{b}_i$ and $\widehat{vb}_i$ [1] and specify $q_i$ on the first packet of each talk-spurt, where

$$q_i = t_i + \hat{b}_i + 4\widehat{vb}_i. \quad (1)$$

Then, for all packets within a talk-spurt, the scheduling time follows the original inter-packet time-spacing at the transmitter

$$q_j = q_i + t_j - t_i. \quad (2)$$

Methods to determine $\hat{b}_i$ and $\widehat{vb}_i$ in the 'normal' mode and the 'spike' mode are different. In our test, the spike detection threshold is set to $2 \cdot |\widehat{vb_{i-1}}| + 200$.

Since the reference algorithm only uses the first packet within a talk-spurt to adapt to the delay, it is not effective if a delay spike happens in the middle of a talk spurt. In contrast, by using time-scale modification, we can adapt to the delay spike as soon as it happens. Fig. 5 illustrates the scenario. In the reference algorithm, all packets within one talk spurt has a constant $q_i - t_i$ value that is equal to $q_{last} - t_{last}$. Those packets that have $a_i > q_i$ will be dropped at the receiver. With our algorithm, the scheduling time $q_i$ will be delayed to catch up the spike. The playout length of spike packets are squeezed after the delay spike. That is, after we calculate $q_i$ based on the reference algorithm, $q_i$ is adjusted if we find that $a_i$ is higher than $q_i$. In the 'normal' mode, only if $a_i$ is within the threshold range of $q_i$, $q_i$ will be updated. The content-based stretching bound is enforced for this threshold. Within the 'spike' mode, we record $q_{last} - t_{last}$, where packet '$last$' is the adaptation starting point as shown in Fig. 5. We will continuously delay the scheduling time $q_i$ (and thus the playback time $p_i$ of packet $i$) until $q_i$ is high enough to catch a spike ($a_i < q_i$). Then, we decrease $q_i$. If $q_i - t_i$ is less than $q_{last} - t_{last}$, we stop decreasing $q_i$.

The procedure to determine $q_i$ and $\hat{\alpha}_i$ is summarized as follows.

1. For each packet received, we have $a_i = t_i + D_{prop} + v_i$ and calculate $\hat{b}_i$ and $\widehat{vb}_i$ according to the reference algorithm given in [1].
2. Calculate $q_i$ for the first packet and all other packets of each talk-spurt as shown in Eqs. (1) and (2).
3. Update $q_i$ as explained in the previous paragraph. In this case, it is assumed that the sender-generated content category, $class(i)$, is transmitted to the receiver in-band (i.e. within each packet).
4. Calculate the playout length $l_i^{(\hat{P})} = l_i^{(O)} + \hat{\delta}_i = q_{i+1} - q_i$ and the target stretching factor $\hat{\alpha}_i = l_i^{(\hat{P})}/l_i^{(O)}$.

5. At scheduling time $q_i$, if $a_i > q_i$, packet $i$ will be dropped. Proceed to packet $i+1$ and restart the algorithm with $i = i+1$.

6. Based on the final playout length $l_{i-1}^{(P)}$, update the playout time by $p_i = p_{i-1} + l_{i-1}^{(P)}$. (In fact, the scheduled playout time $p_i = q_i + D_{calc}$ for packet $i$ may also be used.)

7. Decode and perform time-scale modification based on $\hat{\alpha}_i$, and $l_i^{(P)}$ will be used for packet $i+1$.

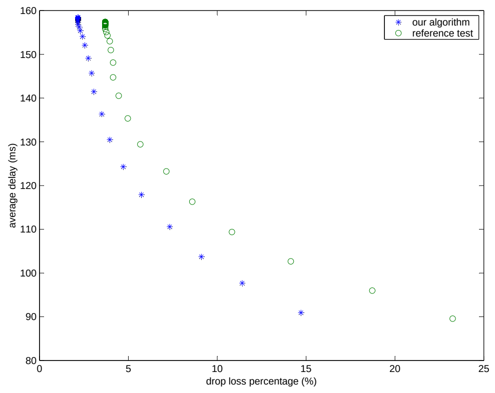
8. Proceed to packet $i+1$ with the same algorithm.

## 5. EXPERIMENTAL RESULTS

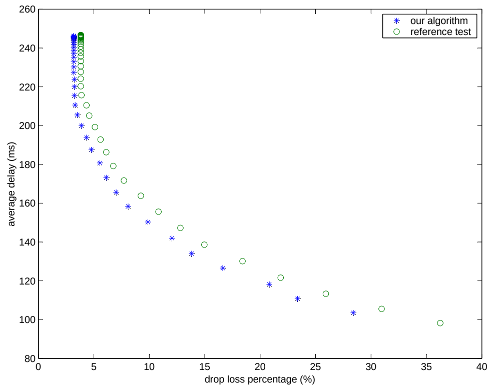
We conducted tests on speech sequences obtained by a sampling rate of 8kHz and 16 bits/sample precision. The interpacketization interval was 20ms (corresponding to 160 samples). Thus, $l_i^{(O)} = 20$ms. We evaluated the simulated network delay according to the description in Section 2 for light as well as heavy Internet traffics. We tested a speech segment consisting of 2000 packets in total, which corresponded to 40 seconds of speech. Since we could control the delay modeling parameters to generate the delay jitter statistics of our interest, the number of 2000 packets is enough to demonstrate the result. The delay correlation plot under heavy traffic is shown in Fig. 3. A content classification routine has been conducted on this speech sequence according to the description in Section 2. Thus, for packets that are categorized as silence, they will not be transmitted. For all other packets, we process them one by one by using the algorithm described in Section 4.

The performance of the reference algorithm was evaluated first. Then, we implement the proposed algorithm by incorporating time-scale modification. The receiver drop-loss percentage was simulated by varying the maximum receiver buffer size expressed in terms of time duration. The performance comparison of the two algorithms is shown in Fig. 6. Since playout scheduling is prediction-based, there is still some drop-loss even when the receiver buffer is large enough to prevent any buffer overflow. The proposed algorithm results in $1.5 \sim 8\%$ improvement over the reference algorithm depending on the maximum buffer size. The average buffering delay is also compared based on maximum buffer size. The proposed algorithm introduces around 1ms longer delay. However, by keeping the local search region smaller and by using the SAMD function, the latency increase could be kept negligibly small. Fig. 6(a) and (b) show the average delay and the drop loss rate for light and heavy traffic cases, respectively.

The resulting audio quality exhibits better intelligibility of the received voice sequence. Time-domain stretching introduce audio artifacts that is within an acceptable range of



(a) Light traffic.



(b) Heavy traffic.

**Fig. 6**. Comparison of adaptive playout algorithms.

quality. By using classification and content-adaptive stretching, we successfully preserve the pitch and the continuity of the original speech.

## 6. REFERENCES

[1] R. Ramjee, J. Kurose, D. Towsley and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide-area networks", in *Proc. IEEE INFOCOM*, 1994.

[2] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms", *ACM/Springer Multimedia Systems*, vol. 5, no. 1, pp. 17-28, Jan. 1998.

[3] J.-C. Bolot, "Characterizing end-to-end packet delay and loss in the Internet", *Journal of High-Speed Networks*, pp. 305-323, Dec. 1993.

[4] J.-C. Bolot and A. Vega-Carcia, "The case for FEC-based error control for packet audio in the Internet", *ACM Multimedia Systems*, 1993.

[5] S. Lee, H. D. Kim, and H. S. Kim, "Variable time-scale modification of speech using transient information", in *Proc. IEEE ICASSP*, pp. 1319-22, 1997.

[6] A. Stenger, K. B. Younes, R. Reng, and B. Girod, "A new error concealment technique for audio transmission with packet loss", in Proc. EUSIPCO, 1996.