Networked On-Line Audio Dilation

John S. Novak, III
University of Illinois at Chicago
Chicago, IL 60607
Jnovak5@uic.edu

Jason Leigh
University of Hawai'i at Manoa
Honolulu, HI 96822
leighj@hawaii.edu

Aashish Tandon
University of Illinois at Chicago
Chicago, IL 60607
Atando2@uic.edu

Robert V. Kenyon
University of Illinois at Chicago
Chicago, IL 60607
kenyon@uic.edu

UbiComp '14 Adjunct, September 13-17, 2014, Seattle, WA, USA
ACM 978-1-4503-3047-3/14/09.
http://dx.doi.org/10.1145/2638728.2638770

Abstract
Audio Dilation is a technique that slows down the tempo of audio signals without changing the pitch or otherwise distorting the sounds. Networked On-line Audio Dilation is an Android application that allows users to dilate audio while engaged in a full-duplex networked conversation. The capability can potentially be used to improve comprehension for hearing or cognitively impaired individuals, or for non-native speakers of foreign languages.

Author Keywords
Speech modification; Audio dilation; Human augmentics

ACM Classification Keywords
H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces; H.5.5. Information interfaces and presentation (e.g., HCI): Sound and Music Computing.

Introduction
Slower speech is often perceived to be more intelligible. Reduced speech rate is one factor of the "clear speech" produced when adults are asked to speak as though to a child, to a non-native speaker, or to a hearing-impaired person [7]. Slower speech rates are also shown to positively affect memory recall [3] and

sentence comprehension [6] and to improve comprehension in multi-talker environments [4].

Audio dilation, or time stretching, is already used in several applications, such as online educational environments which use lecture capture technology [8] and has been suggested as a feature of future hearing aids [2]. Existing algorithms that dilate audio signals are designed for off-line use, i.e., for the time-stretching of complete, pre-existing audio files. In contrast, on-line audio dilation processes signals as and when they are generated, and lend themselves to implementation on a variety of mobile, networked platforms such as smart phones. This expands its potential applications which can ease or aid communication in challenging environments.

Audio Dilation Algorithm

We accomplish audio dilation by adapting a phase vocoder technique from Ellis [1], which was designed to dilate a pre-existing audio file. Previously [5], we modified this algorithm to operate on data as it is generated, as from a microphone or other live source connected to a laptop computer. In this work the algorithm is adapted to a networked, smartphone environment, and supports full duplex communication and independent dilation ratio control.

We define a dilation ratio as the ratio of the amount of input data to output data, e.g., a dilation ratio of ½ or 50% stretches audio by a factor of 2. As in [5], when audio data arrives, it is processed with an on-going Short Term Fourier Transform (STFT) by dividing it into overlapping frames, applying a Hann window, transforming into the frequency domain, and storing in an input buffer indexed with integer values. Dilation is achieved by the construction of a new STFT data structure, with new frames corresponding to non-integer index values at multiples of the dilation ratio. These new frames are generated immediately prior to audio playback, as follows. The magnitude at each frequency for each new frame is constructed by interpolation using the corresponding magnitudes in the prior and subsequent integer-indexed frames. The phase at each frequency for each new frame is the sum of the phase of the previous frame, and the difference between prior and subsequent integer-indexed frames. If integer-indexed frames are required, their phase values are updated accordingly. (E.g., during frame creation, each frequency bin is considered to be a pure sine wave changing only in amplitude.) These frames

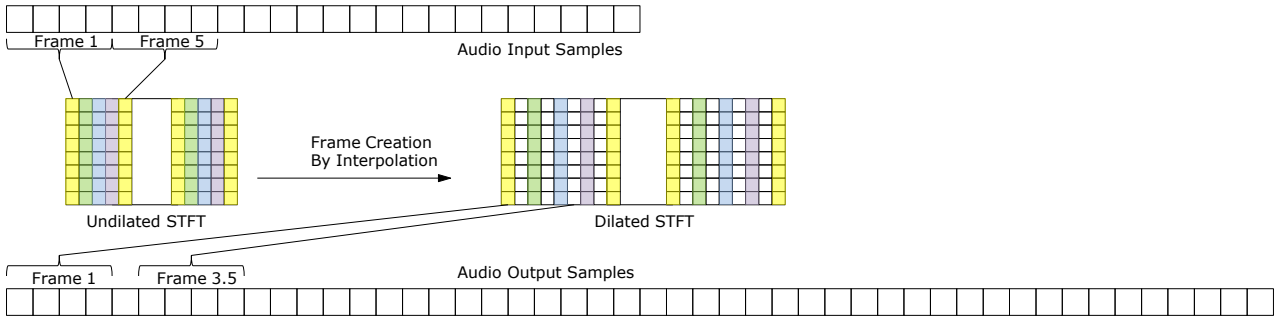

Figure 1: Dilation by 50%

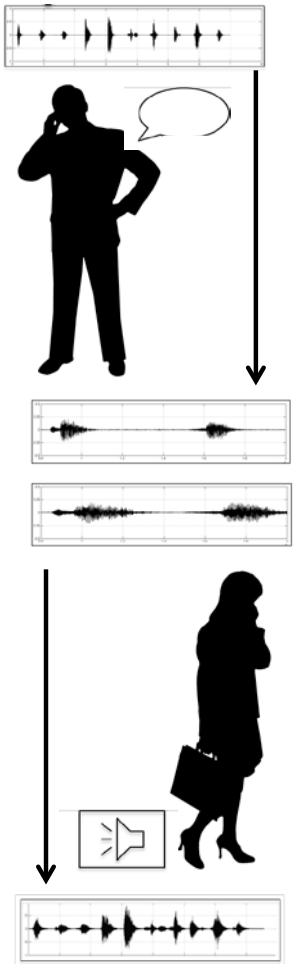

Figure 2

comprise the STFT of a dilated signal, which are then transformed back into the time domain with an inverse STFT, and may be played through speakers.

This process is illustrated in Figure 1.

Audio Dilation User Interface

The Audio Dilation application supports two functions: Initiating or terminating a call, and changing the dilation ratio of received audio. These functions are supported with a text box for the remote IP address, transmit/receive buttons, and a dilation ratio slider.

For our prototype system, each user first enters the IP address of the corresponding remote device into IP text box of the local unit, then presses the transmit and receive button. Each device then creates a pair of sockets to send and receive data. Once all sockets are created, each user may speak, and the application transmit voice data over the network to the other party, supporting full duplex communication. See Figure 2. (Derivative of Shutterstock imagery.)

At this time, each user may use the dilation ratio slider to adjust the amount of dilation for his local device, i.e., the rate at which received audio is played through the local unit. The amount of audio dilation may be changed, and the application will respond immediately.

Because each user may select a different dilation ratio (or none at all) some desynchronization of the conversation may occur. To minimize this effect, we have implemented a voice activation detection (VAD) algorithm, so that lengthy pauses (especially pauses as one conversation partner listens while the other is speaking) are neither transmitted nor dilated.

System Implementation

The Audio Dilation application's two basic functions (initiating or terminating a call, and dilating the received audio) are implemented as follows.

VoIP communication is supported by two datagram sockets (one to transmit and one to receive audio data) and implemented using several Android threads on each device.

Voice Activation and Transmission

All activities on the "transmit" side of the application, including data acquisition from the device microphone, packetization, and transmission are handled in a single thread. The microphone is sampled at 8 KHz, 16 bits per sample, and packaged into 256 sample sub-frames. (Although we consider a full frame of audio data to be 1024 samples, these frames overlap by 75% and it is convenient to send the data as sub-frames.) A simple voice activation detection algorithm discards consecutive sub-frames below an average audio threshold (i.e., "silent" sub-frames) so that pauses longer than 0.5 s are not transmitted. All remaining sub-frames are packaged into datagrams and sent, otherwise unprocessed, through the transmit socket to the remote device.

Reception, Dilation, and Playback

When datagrams are received from a remote source, they are processed in two threads. One thread unpacks the received datagrams into audio data sub-frames. Since the listener may elect to play the audio back at a slower rate than it was generated, the sub-frames are placed in a buffering queue until needed.

A separate thread performs dilation and audio playback: this thread assembles overlapping frames of audio data from the buffering queue as sub-frames become available. The thread then transforms the overlapping frames into the frequency domain and interpolates additional frames of frequency domain as described previously. Finally, the results of this process are transformed back into the time domain, re-assembled into audio sub-frames, and sent to an output buffer for audio playback.

Note that in order to support dynamic dilation ratio changes, the frequency domain interpolation is not performed in advance, but on-line; only when the output buffer is empty or nearly empty are new frames created. So long as frames can be created in less time than is required to play them, the audio is seamless. The dilation process thread monitors the user interface, and when a user changes the dilation ratio, the playback rate changes immediately.

Note also that each device performs all functions listed above, and the application supports full duplex communication with independent playback rates.

Conclusions and Future Work

We have successfully implemented a networked, on-line audio dilation algorithm on an Android platform. The application supports full duplex communication over VoIP, and allows users to control, in real time, the rate at which they experience incoming audio.

Currently, the computational burden restricts the application only to recent, high-end hardware releases. The application has been successfully tested on Samsung Galaxy s4 and Motorola Droid Razr M devices; however slower devices experience severe degradation in sound quality as the devices require more time to produce sound than to play it, resulting in rapidly stuttering sound.

We are developing a cloud-based version to shift the computational burden to server-class machines, thus allowing operation on smaller network-enabled devices with limited or non-existent processing abilities, such as headphones or hearing aids.

References

[1] D. Ellis, A Phase Vocoder in Matlab, http://labrosa.ee.columbia.edu/matlab/pvoc/

[2] E. W. Foo, and G. F. Hughes, "Hearing assistance system for providing consistent human speech," U.S. Patent Application No. 20,120,215,532, Aug. 2012.

[3] S. D. Goldinger et al, "On the nature of talker variability effects on recall of spoken word lists," Journal of Experimental Psychology: Learning, Memory and Cognition. vol. 17, no. 1, pp. 152-162, 1991.

[4] B. Gygi and V. Shafiro. "Spatial and temporal modifications of multitalker speech can improve speech perception in older adults." Hearing research 310 (2014): 76-86.

[5] J. S. Novak, III et al., "On-Line Audio Dilation for Human Interaction," Proc. Interspeech 2013, pp. 1869-1871.

[6] J. F. Schmitt, "The effects of time compression and time expansion on passage comprehension by elderly listeners," Journal of Speech and Hearing Research, vol. 26, no. 3, pp. 373-377, 1983.

[7] R. M. Uchanski, "Clear speech," in D. B. Pisoni and R. E. Remez. The Handbook of Speech Perception. Oxford, UK: Blackwell, pp 207–235, 2005.

[8] E. Zhu, E, and I. Bergom, "Lecture Capture: A Guide for Effective Use". Center for Research Learning and Technology," University of Michigan, 2010.