SpeechSkimmer: Interactively Skimming Recorded Speech

Barry Arons

Speech Research Group
MIT Media Laboratory
20 Ames Street, Cambridge, MA 02139
+1 617-253-2245
barons@media-lab.mit.edu

ABSTRACT

Skimming or browsing audio recordings is much more difficult than visually scanning a document because of the temporal nature of audio. By exploiting properties of spontaneous speech it is possible to automatically select and present salient audio segments in a time-efficient manner. Techniques for segmenting recordings and a prototype user interface for skimming speech are described. The system developed incorporates time-compressed speech and pause removal to reduce the time needed to listen to speech recordings. This paper presents a multi-level approach to auditory skimming, along with user interface techniques for interacting with the audio and providing feedback. Several time compression algorithms and an adaptive speech detection technique are also summarized.

KEYWORDS

Speech skimming, browsing, speech user interfaces, interactive listening, time compression, speech detection, speech as data, non-speech audio.

INTRODUCTION

This paper describes SpeechSkimmer, a user interface for skimming speech recordings. SpeechSkimmer uses simple speech processing techniques to allow a user to hear recorded sounds quickly, and at several levels of detail. User interaction through a manual input device provides continuous real-time control of speed and detail level of the audio presentation.

Speech is a powerful communications medium—it is natural, portable, rich in information, and can be used while doing other things. Speech is efficient for the talker, but is usually a burden on the listener [18]. It is faster to speak than it is to write or type, however, it is slower to listen than it is to read.

Skimming and browsing are traditionally considered visual tasks, as we instinctively perform them when reading a document or while window shopping. However, there is no natural way for humans to skim speech information because of the transient character of audio—the ear cannot skim in the temporal domain the way the eyes can browse in the spatial domain. The SpeechSkimmer user interface described in this paper attempts to exploit properties of speech to overcome these limitations and enable high-speed skimming of recorded speech without a visual display. Possible uses for such a system include reviewing a lecture, listening to a backlog of voice mail, and finding the rationale behind a decision made at a meeting recorded last year.

SpeechSkimmer explores a new paradigm for interactively skimming and retrieving information in speech interfaces. This work takes advantage of knowledge of the speech communication process by exploiting features, structure, and redundancies inherent in spontaneous speech. Talkers embed lexical, syntactic, semantic and turn taking information into their speech while having conversations and articulating their ideas [26]. These cues are realized in the speech signal, often as hesitations or changes in pitch and energy. Speech also contains redundant information; high-level syntactic and semantic constraints of English allow us to understand speech when severely degraded by noise, or even if entire words or phrases are removed. Within words there are other redundancies that allow partial or entire phonemes to be removed while still retaining intelligibility. This work attempts to exploit these acoustic cues to segment recorded speech into semantically meaningful chunks that are then time compressed to further remove redundant speech information.

When searching for information visually we tend to refine our search over time, looking at successively more detail. For example, we may glance at a shelf of books to select an appropriate title, flip through the pages to find a relevant chapter, skim headings until we find the right section, then alternately skim and read the text until the desired information is found. To skim and browse speech in an analogous manner the listener must have interactive control over the level of detail, rate of playback, and style of

In Proceedings of UIST '93: ACM Symposium on User Interface Software and Technology. ACM Press. Nov 3–5 '93. Atlanta pp. 187–196.

presentation. SpeechSkimmer allows a user to control the auditory presentation through a simple interaction mechanism that changes the granularity, time scale, and style of presentation of recorded speech.

A variety of user interface design decisions made while developing SpeechSkimmer are mentioned in this paper. These decisions were based on informal observations and heuristic evaluation of the interface [22] by members of the Speech Research Group. A more formal evaluation is planned for the near future.

This paper reviews related systems that attempt to provide browsing or speech summarization capabilities. The time compression and speech detection techniques used in SpeechSkimmer are described, including a review of the perception of pauses and time-compressed speech. The paper then details the interactive user interface to the system, considerations in selecting appropriate input devices, user feedback, and the system architecture.

RELATED WORK

A variety of predecessor systems relied on structured input techniques for segmenting speech. Phone Slave [41] segmented voice mail messages into five chunks[1] through an interactive dialogue with the caller. Skip and Scan [37] similarly required users to fill out an "audio form" to provide improved access to telephone-based information services. Hyperspeech [2] addressed navigation and speech user interface issues by using recorded interviews that were manually segmented. Degen's augmented tape recorder [9] requires a user to manually press buttons during recording to tag important segments. VoiceNotes [43] transparently shifts the authoring process to the user of the system, produces well-defined segments, and provides a mechanism for quickly scanning through the digitized speech notes. All these techniques provide accurate segmentation, but place a burden on the creator or author of the speech data. SpeechSkimmer automatically segments existing speech recordings based on properties of conversational speech.

Several systems have been designed that attempt to obtain the gist of a recorded message [21, 38] from acoustical information. These systems use a form of keyword spotting in conjunction with syntactic or timing constraints in an attempt to broadly classify the content of speech recordings. Similar work has recently been reported in the areas of retrieving speech documents [15] and editing applications [45]. Work in detecting emphasis [7] and intonation [44] in speech has begun to be applied to speech segmentation and summarization. SpeechSkimmer builds upon these ideas and is structured to integrate this type of information into an interactive interface.

There have been a variety of attempts at presenting hierarchical or "fisheye" views of visual information [12, 28]. These approaches are powerful but inherently rely on a spatial organization. Temporal video information has been displayed in a similar form [30], yet this primarily consists of mapping time-varying spatial information into the spatial domain. Graphical techniques can be used for a waveform or similar display of an audio signal, but such a representation is inappropriate—*sounds need to be heard, not viewed.* This work attempts to present a hierarchical (or "fish ear") representation of audio information that *only* exists temporally.

TIME COMPRESSING SPEECH

The length of time needed to listen to an audio recording can be reduced through a variety of time compression methods (see [3] for a review). These techniques allow recorded speech to be sped up (or slowed down) while maintaining intelligibility and voice quality. Time compression can be used in many application environments including voice mail, teaching systems, recorded books for the blind, and computer-human interfaces.

A recording can simply be played back with a faster clock rate than it was recorded at, but this produces an increase in pitch causing the speaker to sound like Mickey Mouse. This frequency shift results in an undesirable decrease of intelligibility. The most practical time compression techniques work in the time domain and are based on removing redundant information from the speech signal. In the *sampling* or *Fairbanks* method [10], short segments[2] are dropped from the speech signal at regular intervals (figure 1). Cross fading[3] between adjacent segments improves the resulting sound quality.

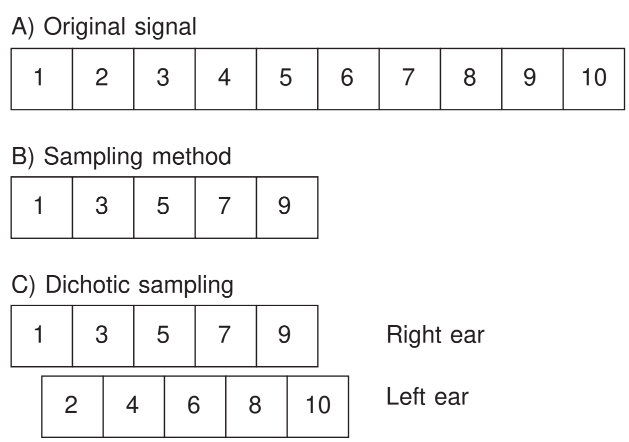

Figure 1. For a 2x speed increase using the sampling method (B), every other chunk of speech from the original signal is discarded (50 ms chunks are used). The same technique is used for dichotic presentation, but different segments are played to each ear (C).

[1]Name, subject, phone number, time to call, and detailed message.

[2]The segments are typically 30–50 ms; longer than a pitch period, but shorter than a phoneme.

[3]Ramping down the amplitude of one signal while ramping up the amplitude of the other.

The *synchronized overlap add method* (SOLA) is a variant of the sampling method that is becoming popular in computer-based systems [39]. Conceptually, the SOLA method consists of shifting the beginning of a new speech segment over the end of the preceding segment (see figure 2) to find the point of highest cross-correlation (i.e., maximum similarity). Once this point is found, the overlapping frames are averaged together, as in the sampling method. SOLA can be considered a type of selective sampling that effectively removes entire pitch periods. SOLA produces the best quality speech for a computationally efficient time domain technique.

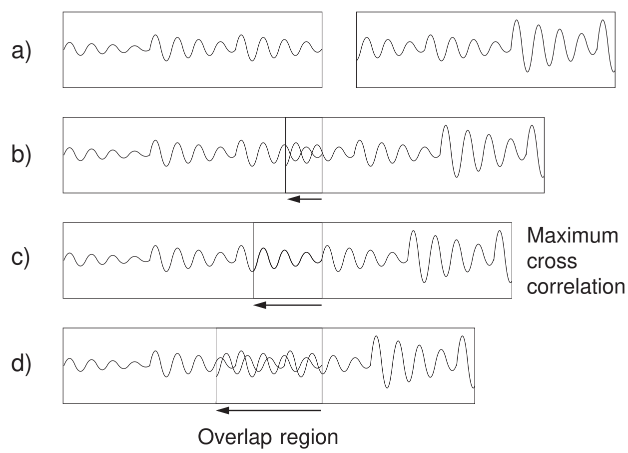

Figure 2. SOLA: shifting the speech segments (as in figure 1) to find the maximum cross correlation. The maximum similarity occurs in case c, eliminating a pitch period.

SpeechSkimmer incorporates several time compression techniques for experimentation and evaluation purposes. Note that all of these speech processing algorithms run in real-time on the main processor of the computer and do not require special signal processing hardware.

The current implementation of the sampling technique produces good quality speech and permits a wide range of time compression values. Sampling with dichotic[4] presentation is a variant of the sampling method that takes advantage of the auditory system's ability to integrate information from both ears. It improves on the sampling method by playing the standard sampled signal to one ear and the "discarded" material to the other ear [42] (see figure 1C). Under this dichotic presentation condition, both intelligibility and comprehension increase [14]. These time compression algorithms run in real-time on a Macintosh PowerBook 170 (25 MHz 68030).[5]

An optimized version of the synchronized overlap add technique called SOLAFS (SOLA with fixed synthesis) [20] is also used in SpeechSkimmer. This algorithm allows speech to be slowed down as well as sped up, reduces the acoustical artifacts of the compression process, and provides a minor improvement in sound quality over the sampling method. The cross correlation of the SOLAFS algorithm performs many multiplications and additions requiring a slightly more powerful machine to run in real-time.[6]

PERCEPTION OF TIME-COMPRESSED SPEECH

Intelligibility usually refers to the ability to identify isolated words. *Comprehension* refers to the understanding of the content of the material (obtained by asking questions about a recorded passage). Early studies showed that single well-learned phonetically balanced words could remain intelligible up to 10 times normal speed, while connected speech remains comprehensible up to about twice (2x) normal speed. Time compression decreases comprehension because of a degradation of speech signal and a processing overload of short-term memory. A 2x increase in speed removes virtually all redundant information [19]; with greater compression, critical non-redundant information is also lost.

Both intelligibility and comprehension improve with exposure to time-compressed speech. It has been reported on an informal basis that following a 30 minute or so exposure to time-compressed speech, listeners become uncomfortable if they are forced to return to the normal rate of presentation [5]. In a controlled experiment extending over six weeks, subjects' listening rate preference shifted to faster rates after exposure to compressed speech. Perception of time-compressed speech is reviewed in more detail in [3, 5, 11].

Pauses in Speech

Pause removal can also be used as a form of time compression. The resulting speech is "natural, but many people find it exhausting to listen to because the speaker never pauses for breath" [32]. In the perception of normal speech, it has been found that pauses exerted a considerable effect on the speed and accuracy with which sentences were recalled, particularly under conditions of cognitive complexity—"Just as pauses are critical for the speaker in facilitating fluent and complex speech, so are they crucial for the listener in enabling him to understand and keep pace with the utterance" [36]. Pauses, however, are only useful when they occur between clauses within sentences—pauses within clauses are disrupting. Pauses suggest the boundaries of material to be analyzed, and provide vital cognitive processing time.

Hesitation pauses are not under the conscious control of the talker, and average 200–250 ms. Juncture pauses are under talker control, usually occur and major syntactic boundaries, and average 500–1000 ms [31]. Note that there is a tendency for talkers to speak slower and hesitate more during spontaneous speech than during oral reading. Recent

[4] A different signal is played to each ear through headphones.

[5] All sound files contain 8 bit linear samples recorded at 22,254 samples/sec.

[6] Such as a Macintosh Quadra 950 (33 MHz 68040) that has several times the processing power of a PowerBook 170.

work, however, suggests that such categorical distinctions of pauses based solely on length cannot be made [34].

Juncture pauses are important for comprehension and cannot be eliminated or reduced without interfering with comprehension [24]. Studies have shown that increasing silence intervals between words increases recall accuracy. Aaronson suggests that for a fixed amount of compression, it may be optimal to delete more from the words than from the intervals between the words—"English is so redundant that much of the word can be eliminated without decreasing intelligibility, but the interword intervals are needed for perceptual processing" [1].

ADAPTIVE SPEECH DETECTION

Speech is a non-stationary (time-varying) signal; silence (background noise) is also typically non-stationary. Background noise may consist of mechanical noises such as fans, that can be defined temporally and spectrally, but can also consist of conversations, movements, and door slams that are difficult to characterize. Speech detection involves classifying these two non-stationary signals. Due to the variability of the speech and silence patterns, it is desirable to use an adaptive, or self-normalizing, solution for discriminating between the two signals that does not rely heavily on arbitrary fixed thresholds [8]. Requirements for an ideal speech detector include: reliability, robustness, accuracy, adaptivity, simplicity, and real-timeness without assuming *a priori* knowledge of the background noise [40].

The simplest speech detection methods involve the use of energy or average magnitude measurements combined with time thresholds; other metrics include zero-crossing rate (ZCR) measurements, LPC parameters, and autocorrelation coefficients. Two or more of these parameters are used by most existing speech detection algorithms. The most common error made by these algorithms is the misclassification of unvoiced consonants, or weak voiced segments, as silence.

An adaptive speech detector (based on [23]) has been developed for pause removal and to provide data for perceptually salient segmentation. Digitized speech files are analyzed in several passes. The first pass gathers energy[7] and ZCR[8] statistics for 10 ms frames of audio. The background noise level is determined by smoothing a histogram of the energy measurements, and finding the peak of the histogram. The peak corresponds to an energy value that is part of the background noise. A value several dB above this peak is selected as the dividing line between speech and background noise. The noise level and ZCR metrics provide an initial classification of each frame as speech or background noise.

Several additional passes through the sound data are made to refine this estimation based on heuristics of spontaneous speech. This processing fills-in short gaps between speech segments [16], removes isolated islands initially classified as speech, and extends the boundaries of speech segments so that they are not inadvertently clipped [17]. For example, two or three frames initially classified as background noise amid many high energy frames identified as speech should be treated as part of that speech, rather than as a short silence. Similarly, several high energy frames in a large region of silence should not be considered to be speech.

This speech detection technique has been found to work well under a variety of noise conditions. Audio files recorded in an office environment with computer fan noise and in a lecture hall with over 40 students have been successfully segmented into speech and background noise. This pre-processing of a sound file executes in faster than real-time on a personal computer.[9]

THE SKIMMING INTERFACE

Skimming Levels

While there are perceptual limits to conventional time compression of speech, there is a strong desire to be able to quickly skim a large audio document. For skimming, non-redundant as well as redundant segments of speech must be removed. Ideally, as the skimming speed is increased, the segments with the least information content are eliminated first.

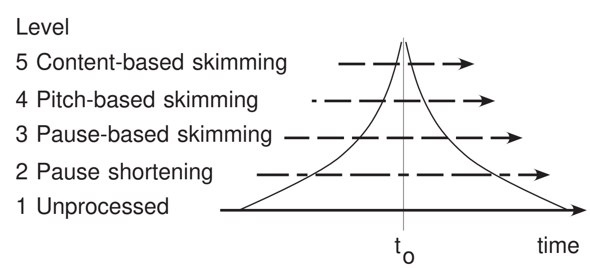

Figure 3. The hierarchical "fish ear" time-scale continuum. Each level in the diagram represents successively larger portions of the levels below it. The curved lines illustrate an equivalent time mapping from one level to the next. The current location in the sound file is represented by t_o; the speed and direction of movement of this point depends upon the skimming level.

A continuum of time compression and skimming techniques have been designed, allowing a user to efficiently skim a speech recording to find portions of interest, then listen to it time-compressed to allow quick browsing of the recorded information, and then slowing down further to listen to detailed information. Figure 3 presents one

[7] Average magnitude is used as a measure of energy [35].

[8] A high zero crossing rate indicates low energy fricative sounds such as "s" and "f." For example, a ZCR greater than 2500 crossings/sec indicates the presence of a fricative [33]. Note that the background noise in most office environments does not contain significant energy in this range.

[9] It currently takes 30 seconds to process a 100 second sound file on a PowerBook 170.

possible "fish ear" view of this continuum. For example, what may take 60 seconds to listen to at normal speed may take 30 seconds when time compressed, and only ten or five seconds at successively higher levels of skimming. If the speech segments are chosen appropriately it is hypothesized that this mechanism will provide a summarizing view of a speech recording.

Three distinct skimming levels have been implemented (figure 4). Within each level the speech signal can also be time compressed. The lowest skimming level (level 1) consists of the original speech recording without any processing. In level 2 skimming, the pauses are selectively shortened or removed. Pauses less than 500 ms are removed, and the remaining pauses are shortened to 500 ms.[10] This technique speeds up listening yet provides the listener with cognitive processing time and cues to the important juncture pauses.

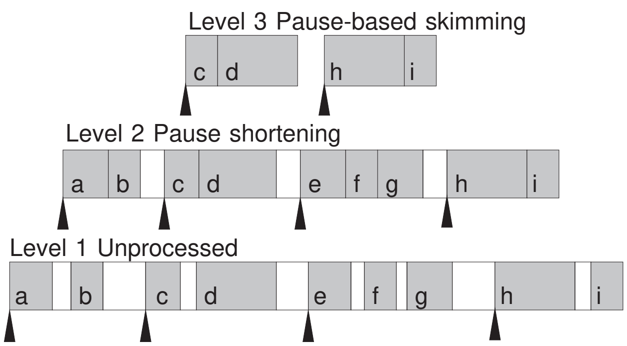

Figure 4. Speech and silence segments played at each skimming level. The gray boxes represent speech, white boxes represent background noise. The pointers indicate valid segments to go to when jumping or playing backwards.

Level 3 is the highest and most interesting skimming technique currently implemented. It is based on the premise that long juncture pauses tend to indicate either a new topic, some content words, or a new talker. For example, filled pauses (i.e., "uhh") usually indicate that the talker does not want to be interrupted, while long unfilled pauses (i.e., silences) act as a cue to the listener to begin speaking [26, 34]. Thus level 3 skimming attempts to play salient segments based on this simple heuristic. Only the speech that occurs just after a significant pause in the original recording is played. After detecting a pause over 900 ms, the subsequent 2 seconds of speech are played (with pauses removed). Note that this segmentation process is error prone, but these errors are partially overcome by giving the user interactive control of the presentation.

It is somewhat difficult to listen to level 3 skimmed speech, as relatively short unconnected segments are played in rapid succession. It has been informally found that slowing down the speech is useful when skimming unfamiliar material. When in this skimming mode, a short (600 ms) pure silence is inserted between each of the speech segments. An earlier version played several hundred milliseconds of the recorded ambient noise between segments, but this fit in so naturally with the speech that it was difficult to distinguish between segments.

In addition to the forward skimming levels, the recorded sounds can also be skimmed backwards. Small segments of sound are each played normally, but are presented in reverse order. When level 3 skimming is played backwards (considered level –3) the selected segments are played in reverse order. In figure 4, skimming level –3 plays segments h–i, then segments c–d. When level 1 and level 2 sounds are played backwards (i.e., level –1 and level –2), short segments are selected and played based upon speech detection. In figure 4 level –1 would play segments in the order: h–i, e–f–g, c–d, a–b. Level –2 is similar, but without the pauses.

Jumping

Besides controlling the skimming and time compression, it is desirable to be able to interactively jump between segments within each skimming level. When the user has determined that the segment being played is not of interest, it is possible to go on to the next segment without being forced to listen to each entire segment [2, 37]. In figure 4, for example, while listening at level 3 segments c and d would be played, then a short silence, then segments h and i. At any time while listening to segment c or d, a jump forward command would immediately interrupt the current audio output and start playing segment h. While in segment h or i, jumping backward would cause segment c to be played. Valid segments for jumping are indicated with pointers in figure 4.

Recent iterations of the skimming user interface have included a control that jumps backward a segment and drops into normal play mode (level 1, no time compression). The intent of this control is to encourage high speed browsing of time-compressed level 3 speech. When something of interest is heard, it is easy to back up a bit and hear the piece of interest at normal speed.

Interaction Mapping

A variety of interaction devices (i.e., mouse, trackball, joystick, and touchpad) have been experimented with in SpeechSkimmer. Finding an appropriate mapping between the input devices and controls for interacting with the skimmed speech has been difficult, as there are many independent variables that can be controlled. For this prototype, the primary variables of interest are time compression and skimming level, with all others (e.g., pause removal parameters and pause-based skimming timing parameters) held constant.

Several mappings of user input to time compression and skimming level have been tried. A two-dimensional

[10] Note that all speech and timing parameters are being refined as the skimming interface develops. The values listed throughout the paper are based on the current system configuration.

controller (e.g., a mouse) allows two variables to be changed independently. For example, the y-axis is used to control the amount of time compression while the x-axis controls the skimming level (see figure 5). Movement toward the top increases time compression; movement toward the right increases the skimming level. The right half is used for skimming forward, the left half for skimming backward.

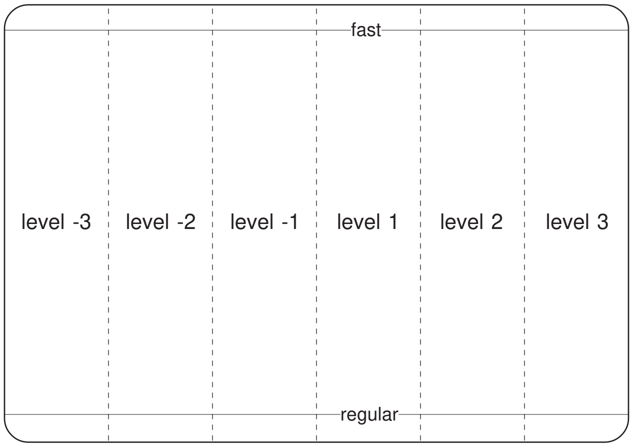

Figure 5. Schematic representation of two-dimensional control regions. Vertical movement changes the time compression; horizontal movement changes the skimming level.

The two primary variables can also be set by a one-dimensional control. For example, as the controller is moved forward, the sound playback speed is increased using time compression. As it is pushed forward further, time compression increases until a boundary into the next level of skimming is crossed. Pushing forward within each skimming level similarly increases the time compression (see figure 6). Pulling backward has an analogous but reverse effect. Note that using such a scheme leaves the other dimension of a 2-D controller available for setting other parameters.

One consideration in all these schemes is the continuity of speeds when transitioning from one skimming level to the next. In figure 6, for example, when moving from fast level 2 skimmed speech to level 3 there is a sudden change in speed at the border between the two skimming levels. Depending upon the details of the implementation, fast level 2 speech may be effectively faster or slower than regular level 3 speech. This problem also exists with a 2-D control scheme—to increase effective playback speed currently requires a zigzag motion through skimming and time compression levels.

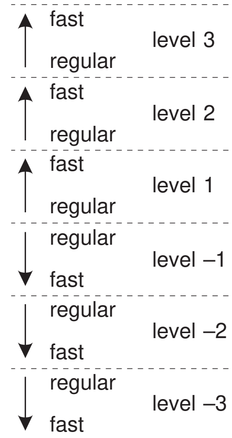

Figure 6. Schematic representation of the control regions for a one-dimensional interaction.

Interaction Devices

A mouse provides accurate control, but as a relative pointing device it is difficult to use without a display. A small hand-held trackball (controlled with the thumb) eliminates the desk space required by the mouse, but is still a relative device and is also inappropriate for a non-visual task.

A joystick can be used as an absolute position device. However, if it is spring-loaded (i.e., automatic return to center), it requires constant physical attention to hold it in position. If the springs are turned off, a particular position (i.e., time compression and skimming level) can be automatically maintained when the hand is removed. The home (center) position, for example, can be configured to play forward (level 1) at normal speed. Touching or looking at the joystick's position provides feedback as to the current settings. However, in either configuration, a off-the-shelf joystick does not provide any physical feedback when changing from one discrete skimming level to another and it is difficult to jump to an absolute location.

A small touchpad can act as an absolute pointing device and does not require any effort to maintain the last position selected. A touchpad can be easily modified to provide a physical indication of the boundaries between skimming levels. Unfortunately, a touchpad does not provide any physical indication of the current location once the finger is removed from the surface.

Touchpad Configuration

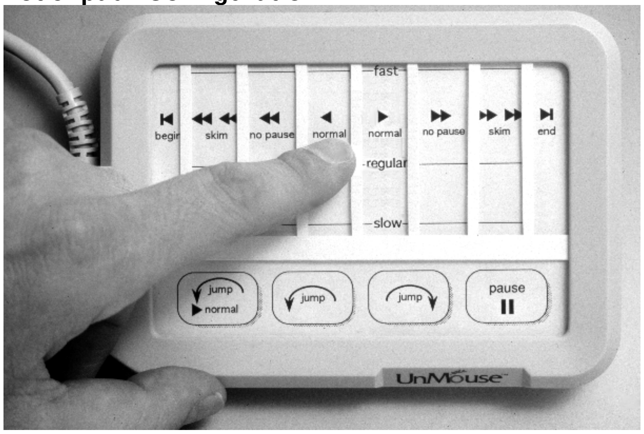

Figure 7. The touchpad with paper guide strips.

Currently, the preferred interaction device is a small (7x11 cm) touchpad [29] with the two-dimensional control scheme. as this provides independent control of the playback speed and skimming level. Thin strips of paper have been added to the touch sensitive surface to indicate the boundaries between skimming regions (see figure 7). In addition to the six regions representing the different skimming levels, two additional regions were added to go to the beginning and end of the sound file. Four buttons provide jumping and pausing capabilities (see figure 8).

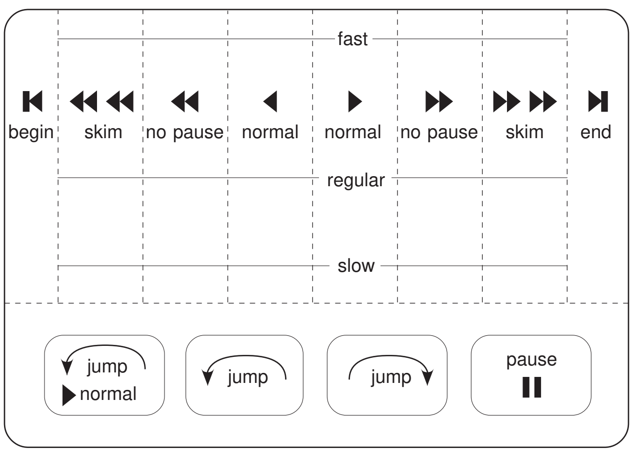

Figure 8. Template used in the touchpad. The dashed lines indicate the location of the guide strips.

The time compression control (vertical motion) is not continuous, but provides a "finger-sized" region around the "regular" mark that plays at normal speed (see figure 9). The areas between the paper strips form virtual sliders (as in a graphical equalizer) that each control the time compression within a skimming level.[11]

[11]Note that only one slider is active at a time.

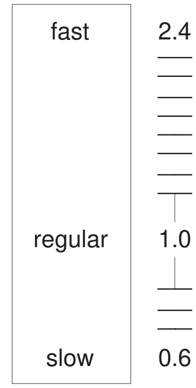

Figure 9. Mapping of the touchpad control to the time compression range.

Non-Speech Audio Feedback

Since SpeechSkimmer is intended to be used without a visual display, recorded sound effects are used to provide feedback when navigating in the interface [6, 13]. Non-speech audio was selected to provide terse, yet unobtrusive navigational cues [43].[12] For example, when playing past the end or beginning of a sound, a cartoon "boing" is played. When transitioning to a new skimming level, a short tone is played. The frequency of the tone increases with the skimming level (i.e., level 1 is 400 Hz, level 2 is 600 Hz, etc.). A double beep is played when changing to normal (level 1)—this acts as an audio landmark, clearly distinguishing it from the other tones and skimming levels.

No explicit feedback is provided for changes in time compression. The speed changes occur with low latency and are readily apparent in the speech signal itself.

Software Architecture

The software implementation consists of three primary modules: the main event loop, the segment player, and the sound library (figure 10). The skimming user interface is separated from the underlying mechanism that presents the skimmed and time-compressed speech. This modularization allows for the rapid prototyping of new interfaces using a variety of interaction devices. SpeechSkimmer is implemented using objects in THINK C 5.0, a subset of C++.[13]

The main event loop gathers raw data from the user and maps it onto the appropriate time compression and skimming ranges for the particular input device. This module sends simple requests to the segment player to set the time compression and skimming level, start and stop playback, and jump to the next segment.

[12]The amount of feedback is user configurable.

[13]Think C provides the object oriented features of C++, but does not include other extensions to C such as operator overloading, in-line macros, etc.

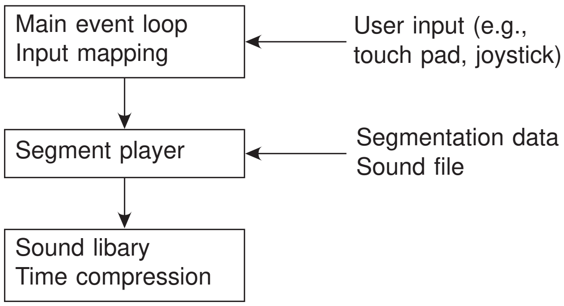

Figure 10. Software architecture of the skimming system.

The segment player is the core software module; it combines user input with the segmentation data to select the appropriate portion of the sound to play. When the end of a segment is reached, the next segment is selected and played. Audio data is read from the sound file and passed to the sound library. The size of these audio data buffers is kept to a minimum to reduce the latency between user input and the corresponding sound output.

The sound library provides a high-level interface to the audio playback hardware (based on the functional interface described in [4]). The time compression algorithms are built into the sound library.

FUTURE PLANS

The "sound and feel" of SpeechSkimmer appear promising enough to warrant continued research and development. Extensions and changes are planned in a variety of areas related to the underlying speech processing and segmentation, as well as to the overall user interface.

A user test is planned as part of this process to evaluate user search strategies, interaction preferences, and the skimming interface as a whole. There are tradeoffs, for example, between automatically skimming short segments of speech and interactively jumping between longer segments that need to be explored and evaluated.

Perceptually Salient Segmentation

Rather than developing additional techniques that fall within the range of skimming levels already explored, the emphasis will be on refining the existing techniques, and creating additional levels of skimming that embody higher amounts of knowledge.

The background noise level detection will be made to adapt to noise conditions that change over time (such as in an automobile). Additional knowledge about speech signals can be added to the algorithm so that speech can be differentiated from transient background sounds [27]. For example, speech must include breath pauses, and these occur with well known timing characteristics [25]. Such information could help distinguish a passing train from a short monologue.

It is possible to dynamically adapt the segmentation algorithm based on the content of the recording rather than using fixed parameters. For example, in determining the segments for level 3 skimming it may be better to analyze the actual pauses in a recording and pick a duration parameter that yields a desirable net compression rather than simply using a fixed pause length.

Prosodic information can be used to automatically extract emphasized portions of recordings [7] and to provide more reliable and informative segmentation. Pitch information combined with speech detection information should provide a better indication of phrase boundaries than using speech detection alone. For example, it has been found that a talker's pitch tends to rise before a grammatically significant pause, and fall before other pauses [34].

Since it is impractical to automatically create a transcript from spontaneous speech, word spotting could be used to classify parts of recordings (e.g., "play the part about pocket-sized computers"). Similarly, speaker identification [33] could be used filter the material presented by person (e.g., "only play what Lisa said"). These speech processing techniques can provide powerful high-level content information. However, to be used for skimming they need to be incorporated into an interactive framework that provides a hierarchical representation of the data, as is described in this paper.

Interaction

Other interaction devices and mappings will continue to be tried. For example, a shuttle wheel[14] with a form of a one-dimensional control may provide a more familiar and intuitive interface than the touchpad.

An absolute position control should be added to the interface. The ability to jump to the beginning and end of a recording are useful, but inadequate. For example, after attending a meeting, it may be desirable to confirm a detail that was discussed "a third of the way" into the recorded minutes.

CONCLUSION

Recorded speech is slow to listen to and difficult to skim. This work attempts to overcome these limitations by combining perceptually based segmentation with a hierarchical representation and an interactive listener control. SpeechSkimmer allows intelligent filtering and presentation of recorded audio—the intelligence is provided through the interactive control of the user.

SpeechSkimmer is not intended to be an application in itself, but rather a technology to be incorporated into any interface that uses recorded speech. Techniques such as this will enable speech to be readily accessed in a range of

[14] As found in video editing controllers and some VCRs.

applications and devices, enabling a new generation of user interfaces that use speech.

ACKNOWLEDGMENTS

Chris Schmandt and Lisa Stifelman participated in valuable discussions during the design of the system and assisted in the editing of this paper. Lisa taught me the inner wizardry of Macintosh programming, and along with Andrew Kass, developed the sound library. Don Hejna provided the SOLAFS implementation. Michael Halle provided imaging and visualization support. Thanks to George Furnas and Paul Resnick for their comments.

This work was sponsored by Apple® Computer, Inc.*

REFERENCES

[1] Aaronson, D., Markowitz, N., and Shapiro, H. Perception and Immediate Recall of Normal and Compressed Auditory Sequences. *Perception and Psychophysics 9*, 4 (1971), 338–344.

[2] Arons, B. Hyperspeech: Navigating in Speech-Only Hypermedia. In *Hypertext '91,* ACM, 1991, pp. 133–146.

[3] Arons, B. Techniques, Perception, and Applications of Time-Compressed Speech. In *Proceedings of 1992 Conference,* American Voice I/O Society, Sep. 1992, pp. 169–177.

[4] Arons, B. Tools for Building Asynchronous Servers to Support Speech and Audio Applications. In *UIST '92. Proceedings of the ACM Symposium on User Interface Software and Technology,* Nov. 1992, pp. 71–78.

[5] Beasley, D.S. and Maki, J.E. Time- and Frequency-Altered Speech. In *Contemporary Issues in Experimental Phonetics*. Academic Press, Lass, N.J., editor, Ch. 12, pp. 419–458, 1976.

[6] Buxton, W., Gaver, B., and Bly, S., *The Use of Non-Speech Audio at the Interface*, ACM SIGCHI, 1991, Tutorial Notes.

[7] Chen, F.R. and Withgott, M. The Use of Emphasis to Automatically Summarize Spoken Discourse. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* IEEE, 1992, pp. 229–233.

[8] De Souza, P. A Statistical Approach to the Design of an Adaptive Self-Normalizing Silence Detector. *IEEE Transactions on Acoustics, Speech, and Signal Processing ASSP-31*, 3 (Jun. 1983), 678–684.

[9] Degen, L., Mander, R., and Salomon, G. Working with Audio: Integrating Personal Tape Recorders and Desktop Computers. In *CHI '92,* ACM, Apr. 1992, pp. 413–418.

[10] Fairbanks, G., Everitt, W.L., and Jaeger, R.P. Method for Time or Frequency Compression-Expansion of Speech. *Transaction of the Institute of Radio Engineers, Professional Group on Audio AU-2* (1954), 7–12, Reprinted in G. Fairbanks. Experimental Phonetics: Selected Articles, University of Illinois Press, 1966.

[11] Foulke, E. The Perception of Time Compressed Speech. In *Perception of Language*. Charles E. Merrill Publishing Company, Kjeldergaard, P.M., Horton, D.L., and Jenkins, J.J., editors, Ch. 4, pp. 79–107, 1971.

[12] Furnas, G.W. Generalized Fisheye Views. In *CHI '86,* ACM, 1986, pp. 16–23.

[13] Gaver, W.W. Auditory Icons: Using Sound in Computer Interfaces. *Human-Computer Interaction 2* (1989), 167–177.

[14] Gerber, S.E. and Wulfeck, B.H. The Limiting Effect of Discard Interval on Time-Compressed Speech. *Language and Speech 20*, 2 (1977), 108–115.

[15] Glavitsch, U. and Schäuble, P. A System for Retrieving Speech Documents. In *15th Annual International SIGIR '92,* ACM, 1992, pp. 168–176.

[16] Gruber, J.G. A Comparison of Measured and Calculated Speech Temporal Parameters Relevant to Speech Activity Detection. *IEEE Transactions on Communications COM-30*, 4 (Apr. 1982), 728–738.

[17] Gruber, J.G. and Le, N.H. Performance Requirements for Integrated Voice/Data Networks. *IEEE Journal on Selected Areas in Communications SAC-1*, 6 (Dec. 1983), 981–1005.

[18] Grudin, J. Why CSCW applications fail: Problems in the Design and Evaluation of Organizational Interfaces. In *CHI '88,* 1988.

[19] Heiman, G.W., Leo, R.J., Leighbody, G., and Bowler, K. Word Intelligibility Decrements and the Comprehension of Time-Compressed Speech. *Perception and Psychophysics 40*, 6 (1986), 407–411.

[20] Hejna Jr., D.J. *Real-Time Time-Scale Modification of Speech via the Synchronized Overlap-Add Algorithm*, Master's thesis, Department of Electrical Engineering and Computer Science, MIT, Feb. 1990.

[21] Houle, G.R., Maksymowicz, A.T., and Penafiel, H.M. Back-End Processing for Automatic Gisting Systems. In *Proceedings of 1988 Conference,* American Voice I/O Society, 1988.

[22] Jeffries, R., Miller, J.R., Wharton, C., and Uyeda, K.M. User Interface Evaluation in the Real World: A comparison of Four techniques. In *CHI '91,* ACM, Apr 1991, pp. 119–124.

[23] Lamel, L.F., Rabiner, L.R., Rosenberg, A.E., and Wilpon, J.G. An Improved Endpoint Detector for Isolated Word Recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing ASSP-29*, 4 (Aug. 1981), 777–785.

[24] Lass, N.J. and Leeper, H.A. Listening Rate Preference: Comparison of Two Time Alteration Techniques. *Perceptual and Motor Skills 44* (1977), 1163–1168.

[25] Lee, H.H. and Un, C.K. A Study of on-off Characteristics of Conversational Speech. *IEEE Transactions on Communications COM-34*, 6 (Jun. 1986), 630–637.

[26] Levelt, W.J.M. *Speaking: From Intention to Articulation,* MIT Press (1989).

[27] Lynch Jr., J.F., Josenhans, J.G., and Crochiere, R.E. Speech/Silence Segmentation for Real-Time Coding via Rule Based Adaptive Endpoint Detection. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* IEEE, 1987, pp. 1348–1351.

[28] Mackinlay, J.D., Robertson, G.G., and Card, S.K. The Perspective Wall: Detail and Context Smoothly Integrated. In *CHI '91,* ACM, 1991, pp. 173–179.

[29] *UnMouse User's Manual*, Microtouch Systems Inc., Wilmington, MA.

[30] Mills, M., Cohen, J., and Wong, Y.Y. A Magnifier Tool for Video Data. In *CHI '92,* ACM, Apr. 1992, pp. 93–98.

[31] Minifie, F.D. Durational Aspects of Connected Speech Samples. In *Time-Compressed Speech*. Scarecrow, Duker, S., editor, pp. 709–715, 1974.

[32] Neuburg, E.P. Simple Pitch-Dependent Algorithm for High Quality Speech Rate Changing. *Journal of the Acoustic Society of America 63*, 2 (1978), 624–625.

[33] O'Shaughnessy, D. *Speech Communication: Human and Machine,* Addison-Wesley (1987).

[34] O'Shaughnessy, D. Recognition of Hesitations in Spontaneous Speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* IEEE, 1992, pp. I521–I524.

[35] Rabiner, L.R. and Sambur, M.R. An Algorithm for Determining the Endpoints of Isolated Utterances. *The Bell System Technical Journal 54*, 2 (Feb. 1975), 297–315.

[36] Reich, S.S. Significance of Pauses for Speech Perception. *Journal of Psycholinguistic Research 9*, 4 (1980), 379–389.

[37] Resnick, P. and Virzi, R.A. Skip and Scan: Cleaning Up Telephone Interfaces. In *CHI '92,* ACM, Apr. 1992, pp. 419–426.

[38] Rose, R.C. Techniques for Information Retrieval from Speech Messages. *The Lincoln Lab Journal 4*, 1 (1991), 45–60.

[39] Roucos, S. and Wilgus, A.M. High Quality Time-Scale Modification for Speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* IEEE, 1985, pp. 493–496.

[40] Savoji, M.H. A Robust Algorithm for Accurate Endpointing of Speech Signals. *Speech Communication 8* (1989), 45–60.

[41] Schmandt, C. and Arons, B. A Conversational Telephone Messaging System. *IEEE Transactions on Consumer Electronics CE-30*, 3 (Aug. 1984), xxi–xxiv.

[42] Scott, R.J. Time Adjustment in Speech Synthesis. *Journal of the Acoustic Society of America 41*, 1 (1967), 60–65.

[43] Stifelman, L.J., Arons, B., Schmandt, C., and Hulteen, E.A. VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker. In *Proceedings of INTERCHI Conference,* ACM SIGCHI, 1993.

[44] Wightman, C.W. and Ostendorf, M. Automatic Recognition of Intonational Features. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* IEEE, 1992, pp. I221–I224.

[45] Wilcox, L., Smith, I., and Bush, M. Wordspotting for Voice Editing and Audio Indexing. In *CHI '92,* ACM SIGCHI, 1992, pp. 655–656.