Using acoustic structure in a hand-held audio playback device

by C. Schmandt
D. Roy

This paper discusses issues in navigation and presentation of voice documents, and their application to a particular hand-held audio playback device, called NewsComm. It discusses situations amenable to auditory information retrieval, techniques for deriving document structure based on acoustical cues, and techniques for interactive presentation of digital audio. NewsComm provides a portable user interface to digitized radio news and interview programs, and it allows occasional connectivity to a networked audio server with disconnected playback.

igitized speech is an attractive and powerful medium for conveying and interacting with information; speech is rich and expressive, can be uttered faster than we type, and can be used while our hands and eyes are otherwise busy. However, speech is largely underutilized in our computing environments, although current computers routinely include speakers and microphones to support audio digitization. Networked digital audio is already practical, whereas digital video is still largely experimental, and the pervasiveness of cellular telephones has far outstripped the first generation of text-based personal digital assistants (PDAs). The barriers to digitized speech are limited neither by technology nor by our interests in talking and listening in a variety of situations.

In this paper we suggest that speech has not realized its full potential as a computer data type because it is difficult to manipulate and not presented so as to take full advantage of the human listening ability. We focus on three aspects of voice as a data type, i.e., as files of digitized sound, not text transcriptions. First is identifying those *situations* or uses in which speech is most attractive. Second is finding *structure* within an audio recording, the acoustic cues that mark important transitions and therefore help us navigate in a manner analogous to paragraph and section structure in a text document. Finally, in this paper, we explore techniques for the *interactive presentation* of audio recordings, utilizing structure in the recording to facilitate retrieval.

Research on the role of voice as a data type at the MIT Media Laboratory dates back to 1983, when a program called PhoneSlave attempted to provide audio "form filling" by asking a series of questions as an answering machine.[1] This paper summarizes relevant work across the intervening years based on the three themes described above, then follows a particular project, NewsComm, as a detailed case study. NewsComm is a hand-held digital audio playback device, designed for portable listening, selecting, and scanning of radio newscasts and other audio "programs." NewsComm is oriented toward portable listening, such as during commute time, and uses a model of occasional network connectivity to an audio server that selects recordings for each user and applies signal processing to audio files to derive their structure.

Although the bulk of this paper is about NewsComm, we wish to discuss the rationale for the work and place it in a larger context. Our approach to digital audio at the Media Lab has been to identify situations

0018-8670/96/$5.00 © 1996 IBM

or opportunities in which speech is valuable, use acoustic cues to provide a structure for navigation, and apply interactive techniques to facilitate browsing by using this structure. We discuss each of these points in turn and then offer examples of how we have explored them before we discuss NewsComm.

Roles of speech

Speech is richer and more expressive than text and uniquely capable of conveying subtle meanings important for persons working together.[2] Intonation and timing in speech convey importance and allow a speaker to emphasize appropriate utterances. A variety of acoustical cues reveals the emotional or affective state of the talker. Recognizing the identity of the talker imparts credence to a recording and may help involve the listener more intimately. Because it is usually easier to prepare a talk than write a paper, recordings of lectures may offer more timely and immediate access to technical information, much as radio news is always more up to date than news printed in newspapers.

Audio is also particularly attractive to an increasingly mobile population of computer users. The existing telephone networks provide effective and increasingly digital and wireless means of accessing information away from traditional office environments. Audio is attractive for mobile applications because it can be used while one's hands and eyes are busy performing other tasks, such as driving or walking. Because speech does not require a display, it may consume less power than text in a PDA and can be used in the dark.

With all these positive features, why are digital audio recordings not used more extensively in computing environments? Unfortunately, speech also suffers from a number of limitations that make it difficult to retrieve the information in a recording or to quickly evaluate its relevance.

Speed is a major factor in accessing audio recordings. Although we speak more rapidly than we are able to write or type, we can read much more quickly than any one of these. Although, as will be discussed below, we can partially compensate by compressing time in a recording, a related disadvantage of audio is our inability to skim or scan it quickly, as we can a printed page. In part this is due to the serial nature of audio, which is by definition perceived as variation in air pressure at the eardrum over time. Our eyes can move quickly to scan, review, and pick items for our attention from a visual display, but the transitory nature of audio usually interferes with performing analogous actions while listening.

Another area of concern is the homogeneity or amorphous structure of an audio recording. Authors use orthographic cues consisting of punctuation, capitalization, and double-spacing or initial tabs before new paragraphs to provide clues as to the meaning and internal structure of even informal communications; more formal reports and papers use major and minor section headings to further delineate their structure. Although speech, and especially conversation, indicates structure with emphasis, pauses, and turn-taking, these indicators are not immediately apparent in an audio recording.

This paper is about techniques to minimize the negative aspects of digitized speech by detecting structure in the speech signal and developing interactive techniques to use this structure to make the recording more accessible. But these techniques are still imperfect and only partially compensate for the limitations imposed by the audio medium. The reality of these limitations suggests that audio as a data type will be most valuable in situations of select use. Recordings are most useful for very timely information that has not yet been reduced to print, for information that never is translated to text, such as voice mail messages, and in situations where the listener is mobile or performing other tasks. These factors influenced the designers of NewsComm to focus on news as an information source and on portability—to allow use while commuting, exercising, or otherwise away from conventional computers.

Deriving structure

If audio data consist of small snippets of sound, as in telephone messages, calendar entries, or personal reminders, it may suffice to focus on the user interface for choosing the recordings, controlling their playback, and skipping between them quickly. But with longer recordings we require a means to navigate within the recording and to move rapidly between different portions while searching for the most interesting parts. It is also valuable to be able to summarize a recording, or quickly hear its main points, to determine whether we want to listen to it in its entirety.

Despite gains in speech recognition we are a long way from being able to automatically transcribe an unstructured recording, making it impossible to manipulate digitized audio at the lexical or semantic levels. However, in Reference 3 we presented the concept of *semistructured* audio, after Malone's use of the term for managing text messages with arbitrary descriptive fields.[4] In manipulating audio in this manner, we use acoustical evidence to derive various forms of structure from the recording. Although we have no certainty that this acoustically derived structure actually corresponds to the linguistic or logical structure of the discourse, in practice it often provides useful and scalable interaction hooks for enabling both random access and summarization.

A number of components can contribute to acoustically derived audio structure. The rhythm and intonation of a conversation or a monologue are structural cues as to the roles of those talking, the flow of topics, and the thought processes of the conversants. Pauses of varying duration serve different roles; shorter pauses (less than about 400 milliseconds) occur as a talker composes words "on the fly," whereas pauses of longer juncture usually occur at boundaries, such as when a speaker introduces new topics.

Talkers emphasize points by using an increased pitch range, and pitch range also sometimes indicates a new topic. Chen and Withgott describe a method for summarizing speech recordings by locating and extracting emphasized portions of the recording.[5] Hidden Markov Models (HMMs) are used to model regions of emphasis. The energy, delta energy, pitch, and delta pitch parameters are extracted from the speech recording and used as parametric input to the HMM. Training data were collected by manually annotating the emphasized portions of several speech recordings. These factors could be combined, as suggested in Reference 6, to provide a hierarchical acoustic analysis of discourse structure.

In a conversation the participants also take turns; speaker segregation can reveal this aspect of conversational structure. For example, Gish et al. have developed a method for segregating speakers engaged in dialog.[7] The method assumes no prior knowledge of the speakers. A distance measure based on likelihood ratios is developed to measure the distance between two segments of speech. Agglomerative clustering based on this distance measure is used to cluster a long recording by speaker. The method has been successfully applied to an air traffic control environment where the task is to separate the controller's speech from that of pilots. Wilcox et al. also use a Gaussian probability-based clustering algorithm to index speakers.[8] Additionally, they use a Hidden Markov Model to model speaker transition probabilities.

Different cues to structure are appropriate to different source material. Intonational cues may be strong in a lecture but weak in a newscast since newscasters tend to use heightened emphasis with considerably greater stress than that used in ordinary conversation. In our work, pauses alone turned out to be very reliable story boundary indicators in formal British newscasts, but much less valuable in commercial North American radio news. Speaker differentiation between a pair of talkers is very strong when summarizing an interview and less so in a recorded telephone conversation. The NewsComm approach assumes a centralized audio server that performs signal processing and segmentation on powerful computers, to be downloaded as structured audio into a less powerful hand-held device. The audio server could utilize different segmentation cues, depending on the program source.

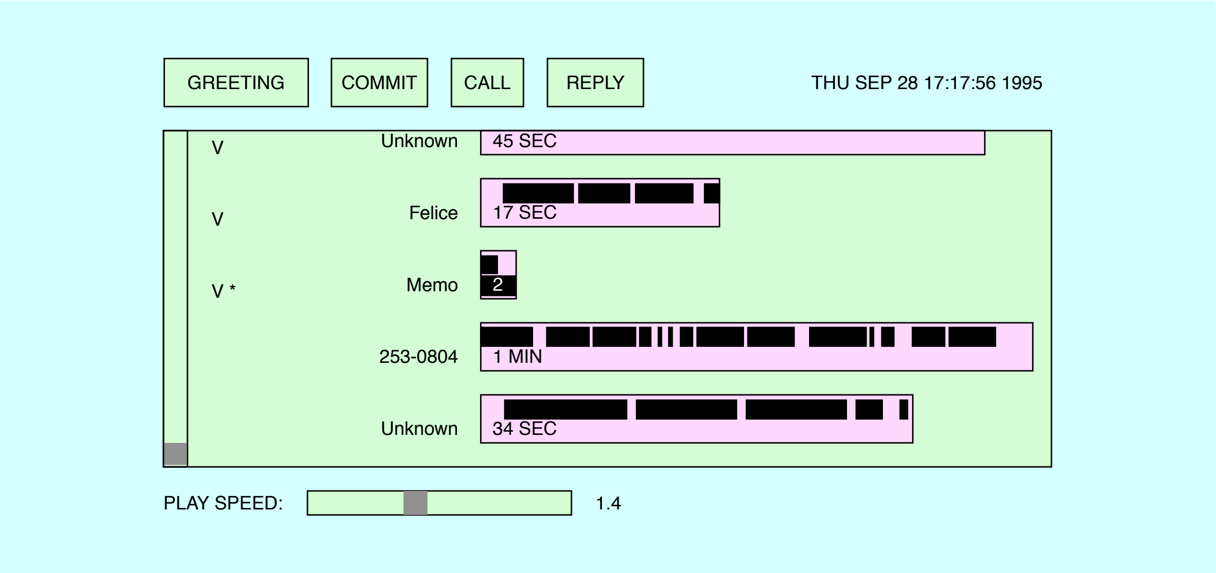
A number of components can contribute to acoustically derived audio structure.

The purpose of deriving the semistructured acoustical cues in an audio recording is to more effectively present the recording to a listener. In the extreme case, the structure can be used for automatic summarization of recordings, but the quality of such a summary is dubious. Because of the loose correlation between acoustical cues and semantic content, a concise summary may miss some important portions, whereas a lengthy summary will include extraneous speech. Therefore, structural cues are more apt to be successful when incorporated into interactive techniques, allowing a listener to control playback.

Presentation of digital audio recordings

If audio recordings consist of a number of short, independently recorded segments, interactive playback may entail simply being able to jump rapidly from segment to segment, using an appropriate input mech-

Figure 1 Graphical voice mail user interface, displaying speech and silence intervals



anism. One example to be discussed below is a graphical voice mail user interface that allows the user to quickly jump from message to message by use of a mouse-button click. This may be adequate to find a particular message among a number of old messages, because the first few seconds usually uniquely identify the recording; but this does not help with finding the important information within a message. Similar issues arise with the hand-held voice note taker, also described below, in which mouse buttons are replaced with push buttons on a hand-held device, and the audio data consist of personal memos.

For longer recordings, time-compression techniques can allow listeners to hear recorded speech in significantly less time than was required to originally speak it. Algorithms such as SOLA (Synchronous Overlap and Add) maintain reasonable intelligibility without shifting pitch, which happens when we speed up an analog audio or video tape. As greater degrees of time compression are used, increasingly large portions of the original signal are discarded; at some point whole phonemes vanish. Compression ratios of 1.3 to 1.5 are manageable by naive listeners, and Voor[9] demonstrated that fairly short adaptation times (minutes) increased comprehension. In fact, after adjusting to time-compressed speech, listening to recordings of normal speed can be discomforting.[10] Still, a rough upper bound of approximately twice the normal speed limits comprehensible time scaling. Beyond this, structural information must be employed to determine what larger regions of the recording can be skipped.

The goal of a structurally informed auditory user interface is to assist the user's attempts to randomly access the recording by suggesting or automatically enforcing selective "jump" points, attempting to summarize the recordings by extracting salient portions, or otherwise enhance the experience of listening to one or more possibly time-compressed audio streams. If a graphical interface medium is available, structure can be represented visually, and a mouse or other input device used to control playback and enable random access. For example, Degen et al. displayed sound with vertical elements and color to indicate amplitude and points at which a user had pressed a button during recording.[11] The Intelligent Ear was an early graphical editor displaying amplitude and keywords detected via speech recognition.[12] More recently Kimber et al. used speaker differentiation to identify talkers recorded in a meeting and displayed them on different horizontal "tracks."[13]

Another promising technique for audio browsing is simultaneous presentation of multiple audio streams. A number of acoustic cues allow the listener to separate a mix of sounds into distinct sources and selectively attend to any one of them; these include location, harmonics and frequency, continuity, volume, and correlation to visual events.[14] Spatial auditory presentation techniques currently being deployed in virtual reality applications also enable simultaneous presentation of multiple spatialized speech recordings. But does listening to three recordings simultaneously result in a threefold performance improvement? Unfortunately most of the experimental evidence suggests that relatively little information leaks through the secondary channels while attending to any one channel.[15] However, the experimental listening tasks are often very demanding, requiring shadowing or listening for a target phrase, and more typical comprehension measurements may not be so degraded for tasks such as finding interesting stories in radio newscasts.

Previous work

This section describes previous work in interacting with semistructured audio at the Media Lab. Through a number of independent projects, we have explored various aspects of representing and interacting with voice as a data type. Although this work includes graphical user interfaces, its emphasis is on nonvisual interactions. Among these projects, various ones were designed to meet specific user needs in specific situations, and for particular sources of audio. Several of them assume a greater amount of structural knowledge of the recordings than can ordinarily be expected, to allow greater emphasis on interaction techniques. But taken together they serve to illustrate a spectrum of interactive techniques, and NewsComm seeks to build upon them.

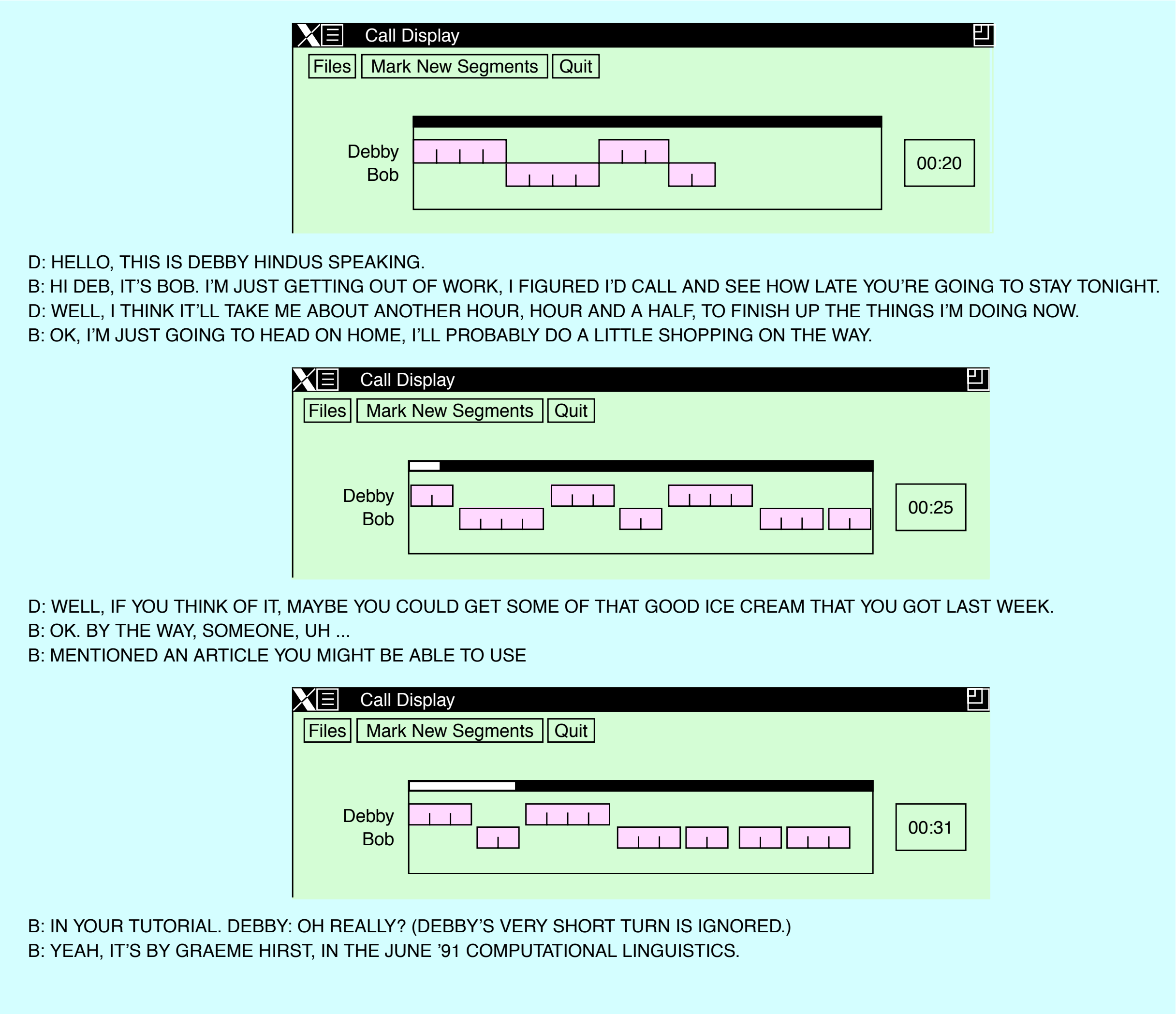
PhoneSlave was an early attempt at semistructured audio, where the structure was created on the fly by asking callers a series of questions such as “Who’s calling please?” and “At what number can you be reached?”[1] Although PhoneSlave did not understand any of the answers, it assumed that callers were cooperative, and so used these recorded audio snippets to allow the owner of the answering machine to query “Who left messages?” PhoneSlave included both a speech recognition user interface as well as a simple graphical interface in which recorded messages were displayed as a series of bars that changed color in synchronization with playback. PhoneSlave was more attractive in the early days of voice messaging before we all became accustomed to listening for a beep and then rapidly reciting our messages. Segmented messages benefit the listener, but conversational techniques quickly become tedious to the caller. Still, a method of asking the caller questions is appearing in some voice mail and call management products.

Voice messages are usually brief, and sophisticated navigation is not generally necessary, but a user interface should allow the recipient to rapidly jump between messages. A more recent approach to voice mail uses a graphical interface (Figure 1)[16] in which the bars of the SoundViewer widget represent periods of speech and silence, with limited user annotation in the form of “bookmarks.” Playback speed is controlled by a slider. The SoundViewer affords direct manipulation of the audio recording; a playback bar moves left to right to show current play position, and the user can click to cause playback to jump to any other point. The SoundViewer also allows the user to annotate the recording with “bookmarks” and cut and paste sound between audio-capable applications. A number of projects employed iterations on this playback controller, such as a personal calendar that includes audio entries.[17]

Hindus used a simple channel separation scheme to segment telephone conversations according to which party was speaking. A “retrospective” display (Figure 2) showed the recent past of the conversation as a scrolling window that a listener could use to mark sections to save as a recording after the call.[3] The moving stream of SoundViewers indicated both changes in speaker and spurts of speech by the same speaker and was designed to provide visual cues to enable selection of very recent portions of the conversation. A similar format was used during retrieval of a previously recorded conversation. This structured recording with graphical evidence of turn-taking was designed to facilitate recall but was not tested extensively in part due to privacy concerns.

As a step toward graphically managing larger quantities of loosely structured audio, Horner grouped both text and SoundViewers in a user interface to audio news, with text from the closed-captioned television channel.[3] Both text and sound portions were active, and clicking on either representation caused playback to jump to the correct region; they also scrolled in synchrony during audio playback (Figure 3). The SoundViewer was augmented with annotations to indicate topic category (national, international, business, sports, etc.) and expanded in a hierarchical manner to allow both coarse- and fine-grained manipulation of playback. The rich structural representation

Figure 2 A "retrospective display" showing recent turns in an ongoing telephone conversation



D: HELLO, THIS IS DEBBY HINDUS SPEAKING.
B: HI DEB, IT'S BOB. I'M JUST GETTING OUT OF WORK, I FIGURED I'D CALL AND SEE HOW LATE YOU'RE GOING TO STAY TONIGHT.
D: WELL, I THINK IT'LL TAKE ME ABOUT ANOTHER HOUR, HOUR AND A HALF, TO FINISH UP THE THINGS I'M DOING NOW.
B: OK, I'M JUST GOING TO HEAD ON HOME, I'LL PROBABLY DO A LITTLE SHOPPING ON THE WAY.

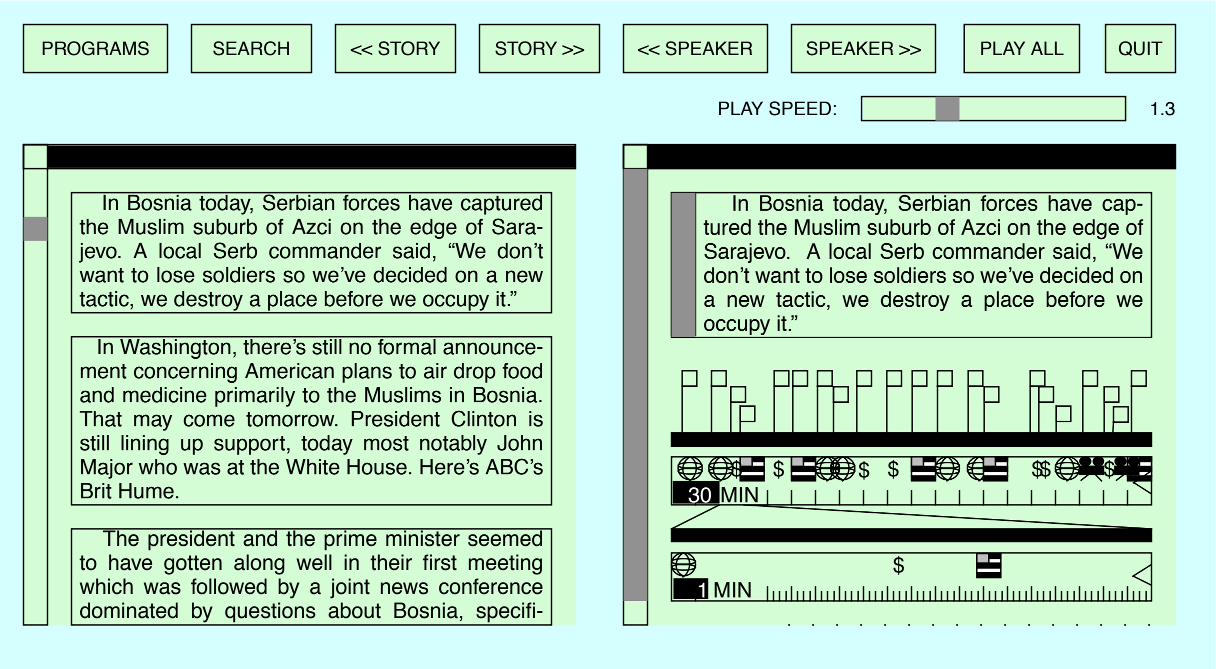
D: WELL, IF YOU THINK OF IT, MAYBE YOU COULD GET SOME OF THAT GOOD ICE CREAM THAT YOU GOT LAST WEEK.
B: OK. BY THE WAY, SOMEONE, UH ...
B: MENTIONED AN ARTICLE YOU MIGHT BE ABLE TO USE

B: IN YOUR TUTORIAL. DEBBY: OH REALLY? (DEBBY'S VERY SHORT TURN IS IGNORED.)
B: YEAH, IT'S BY GRAEME HIRST, IN THE JUNE '91 COMPUTATIONAL LINGUISTICS.

facilitated by the closed-captioned information is valuable, but the majority of the audio we wish to manipulate is not so captioned.

A graphical interface facilitates many aspects of audio interaction, enabling selection between sound files, navigation within a recording during playback, and display of attributes of the recording such as periods of speech and silence, and total duration. But graphical interfaces require displays, and recorded speech is most valuable in highly mobile environments in which the user's visual attention may be otherwise occupied. Stifelman explored nonvisual management of speech snippets in VoiceNotes,[16] a portable audio memo taker (Figure 4). VoiceNotes recorded memos into lists, or categories, and allowed navigation by button or speech recognition. It explored several navigation and user interface possibilities, using nonspeech auditory cues to help give a sense of "place" during playback. VoiceNotes also used time compression as a global playback parameter (using a volume control knob) as well as automatically while summarizing a list and to provide user feedback when the user deleted a memo; for example, VoiceNotes would confirm by saying "Deleting ..." and then play the memo to be deleted at a fast rate. Even though

Figure 3 Coordinated text and graphical audio display based on a closed-captioned newscast



In Bosnia today, Serbian forces have captured the Muslim suburb of Azci on the edge of Sarajevo. A local Serb commander said, “We don’t want to lose soldiers so we’ve decided on a new tactic, we destroy a place before we occupy it.”

In Washington, there’s still no formal announcement concerning American plans to air drop food and medicine primarily to the Muslims in Bosnia. That may come tomorrow. President Clinton is still lining up support, today most notably John Major who was at the White House. Here’s ABC’s Brit Hume.

The president and the prime minister seemed to have gotten along well in their first meeting which was followed by a joint news conference dominated by questions about Bosnia, specifi-

In Bosnia today, Serbian forces have captured the Muslim suburb of Azci on the edge of Sarajevo. A local Serb commander said, “We don’t want to lose soldiers so we’ve decided on a new tactic, we destroy a place before we occupy it.”

VoiceNotes are typically very short recordings, managing lists with a nonvisual interface proved to be challenging.

Arons’s Hyperspeech project[18] demonstrated a conversational interface to audio recordings, using speech recognition for input. Arons recorded interviews with four experts answering the same questions and then hand-segmented the recording and generated typed hypermedia links. This allowed the listener to ask questions such as “What did Minsky say about that?” and “What are the opposing views?” Some listeners enjoyed the conversational nature of the interaction, finding it a natural way to interact with the recordings. But this project was limited by the need to hand-segment the recordings in order to provide the typed links that related them to one another and enabled pragmatic questions about their content.

Despite the contributions of these projects, dealing with longer durations and less well-structured recordings is much more difficult, and hence leads to more speculative research. Arons’s SpeechSkimmer[19] explored both structuring techniques and user interaction for an audio context of recorded lectures. Arons

Figure 4 A hand-held voice memo taker

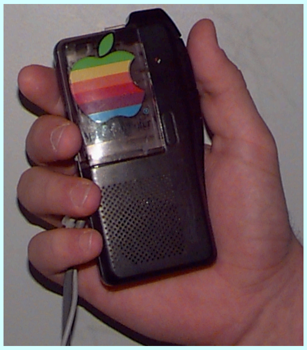
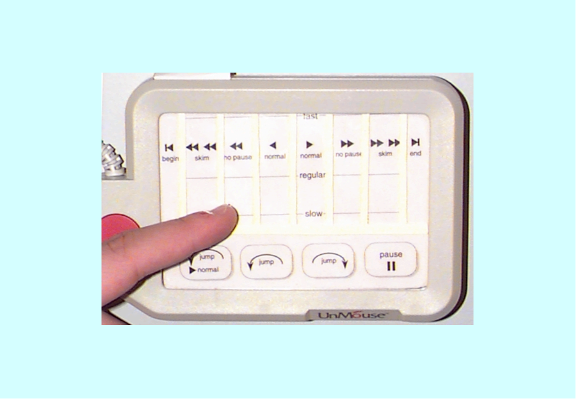


Figure 5 SpeechSkimmer hand-held controller



analyzed audio recordings for pause structure (energy) and intonation (pitch), building on work by Chen and Withgott[5] who used Hidden Markov Models to find portions of a recording that a speaker had emphasized by using an increased pitch range.

SpeechSkimmer provided playback at multiple levels of detail and with continuous control of playback speed for both forward and backward playback. It used a hand-sized touch tablet, divided into vertical "sliders"; each slider controlled speed for one of three playback modes (Figure 5). Playback could be of the entire recording, the recording with short pauses removed and long pauses shortened, or of just the emphasized portions. Playback of the emphasized portions resulted in an audio summary, playing only short portions of the sound; a button at the bottom of the tablet allowed the user to jump back and play the most recent portion in its entirety. Evaluations of SpeechSkimmer depended upon both the ability of the emphasis detection algorithm to find salient portions of the recording and the playback control afforded by the user interface. The dominant interactions by users consist of managing playback speed and random access within the sound, constrained by SpeechSkimmer's precalculated "jump points." This combination seemed useful, although subjects desired to navigate by absolute position within the sound, a common lack in nonvisual user interfaces.

SpeechSkimmer aimed for effective listening by using structure to determine which portions of sound to play selectively, with user control over playback speed. A rather different approach was taken by Mullins in AudioStreamer, which played multiple newscasts simultaneously for browsing.[20] AudioStreamer used spatially separated sound, with three sources in front of the listener separated horizontally by 60 degrees, to facilitate having listeners selectively attend to any one of the channels. (Selective attention is our ability to attend to a single sound source while surrounded by many—the "cocktail party effect.") AudioStreamer used noncontact head-position sensing[21] to enhance the listener's selective attention experience; moving the head toward one of the three sources caused it to become louder by 10 decibels (dB), allowing it to dominate. But since listener interest diminishes over time, the gain on the dominant channel decays as well. If the listener again attends to this channel, it gets still louder, and the decay time constant becomes longer (Figure 6). At the highest level of attention, the subordinate channels cease playing temporarily.

Although the model of interest for AudioStreamer allows the listener to rapidly switch attention between channels, it must also cope with the fact that little of the out-of-focus channel may be heard at all. It accomplishes this by detecting transitions in the audio stream based on pauses, speaker changes, and for some data, closed-captioned text transcriptions. At such a transition, a 400-hertz (Hz), 100-millisecond tone is inserted in the stream, and its gain is increased by 10 dB. Again, this increased gain rapidly decays, as shown in Figure 7.

NewsComm: Portable structured audio playback

The remainder of this paper is about NewsComm, the most recent project in this series exploring uses of voice as a data type. Based on the potential roles of recorded speech, NewsComm focuses on a mobile listener who may be busy performing other tasks at the same time as listening, and concentrates on timely data of news and radio interviews. NewsComm includes pause detection and a speaker segmentation algorithm to derive structure from acoustic cues. These are incorporated into the interactive presentation by suggesting jump points when the listener wishes to skip ahead in a recording, and for automatic summarization.

NewsComm is a hand-held device that provides interactive access to structured audio recordings. It is the first fully contained portable device built in this series of research projects. The device, shown in Figure 8, is meant for mobile use; it can be held and operated with one hand and does not require visual attention for

Figure 6 AudioStreamer enhances the gain on an in-focus audio channel, but the gain decays over time

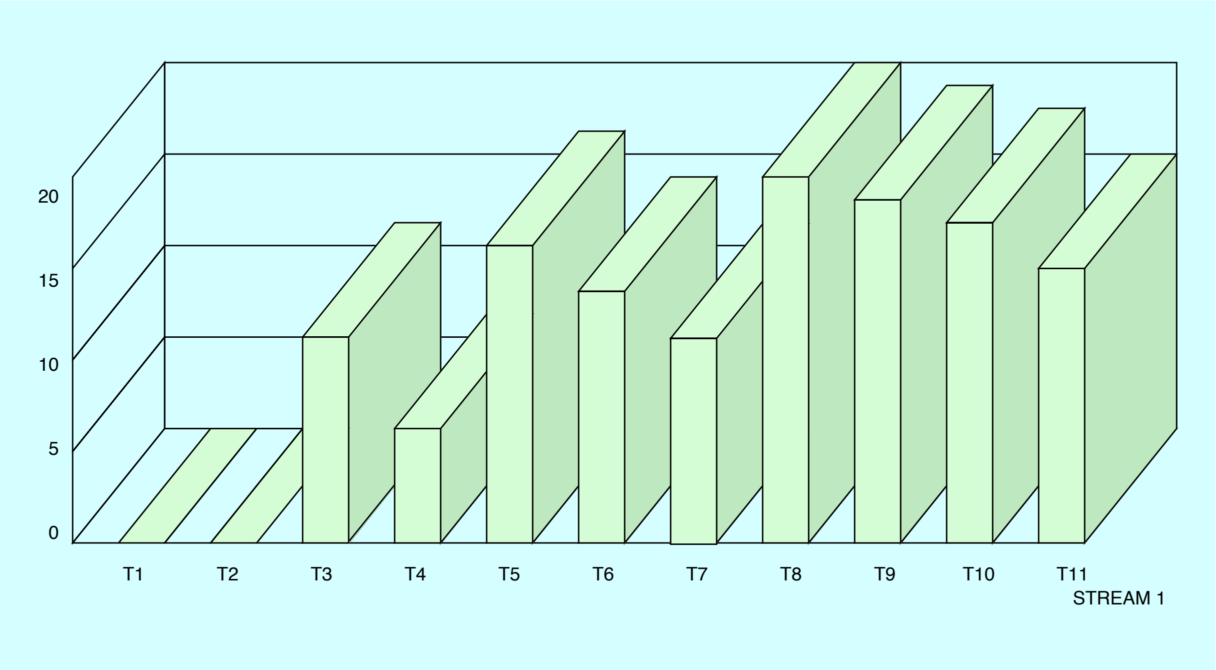


Figure 7 At transitions, each channel becomes momentarily louder

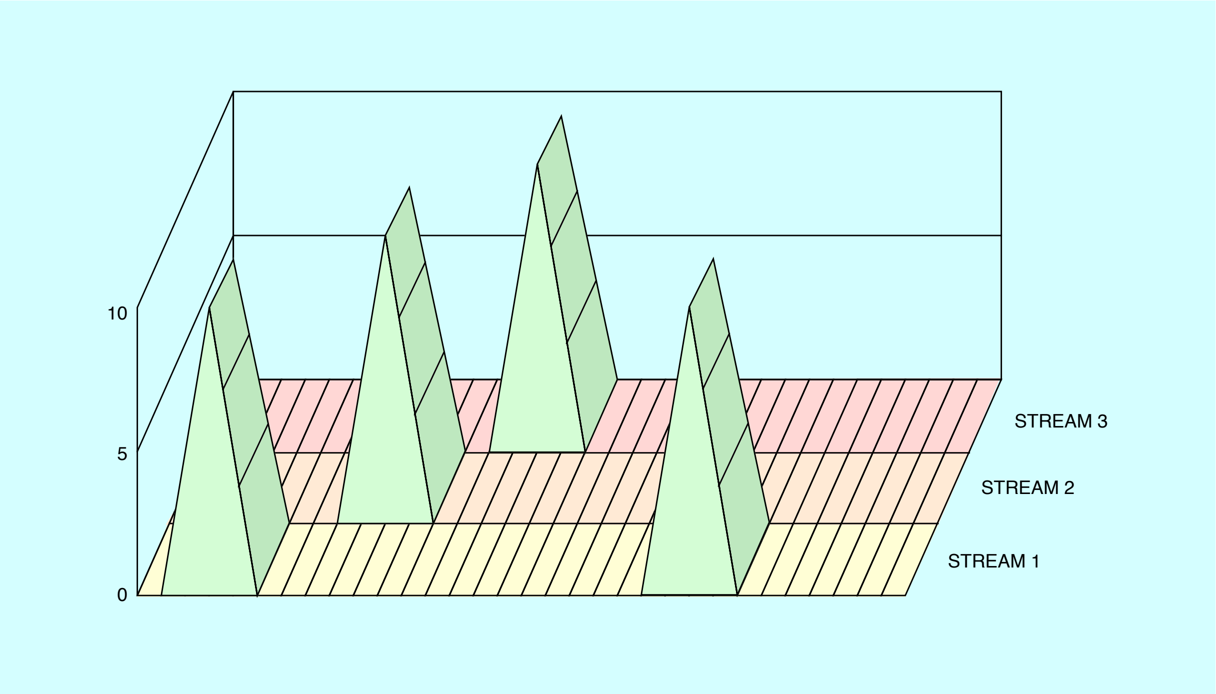
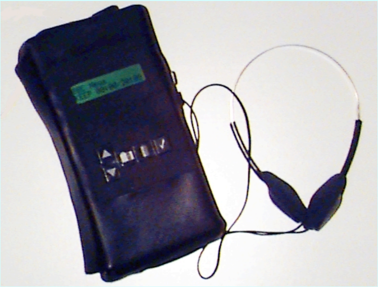


Figure 8 The NewsComm hand-held audio playback device with headphones



most operations. On top are a display and controls for selecting and managing recordings. The right side houses the navigation interface, which can be controlled with the thumb while holding the device.

Users intermittently connect to an audio server and receive personally selected audio recordings that are downloaded into the local random access memory (RAM) of the hand-held device. The user then disconnects from the server and interactively accesses the recordings off line.

Figure 9 gives an architectural overview of NewsComm. When the hand-held device connects to the audio server (top part of figure), the usage history and preferences are uploaded to the audio manager, and on the basis of this information, a set of filtered structured audio recordings are downloaded into the local audio memory of the hand-held device. The audio server collects and processes audio from various sources, including radio broadcasts, books and journals on tape, and Internet audio multicasts. Typical content might include newscasts, talk shows, books on tape, and lectures. If deployed, the hand-held device would download recordings from the audio server through intermittent high-bandwidth connections, such as overnight while at home or via enterprise networks while at work. As implemented, audio files, associated audio structure information, and usage history are exchanged with the server by docking the PCMCIA (Personal Computer Memory Card International Association) memory card of NewsComm into a server port. The server consists of two parts: a Sun Sparcstation** 10 for signal processing and a Pentium**-based PC for file management.

The audio processor module in the server automatically finds two types of features in each audio recording stored in the server: pauses and speaker changes. All audio in the server is structured by the audio processor and then stored in the audio library, a large network-mounted hard disk drive.

Users can download structured audio (audio data plus the list of associated features) from the server by connecting their hand-held device to the audio manager. The audio manager selects recordings based on a preference file that the user has previously specified, and also based on the recent usage history uploaded from the hand-held device.

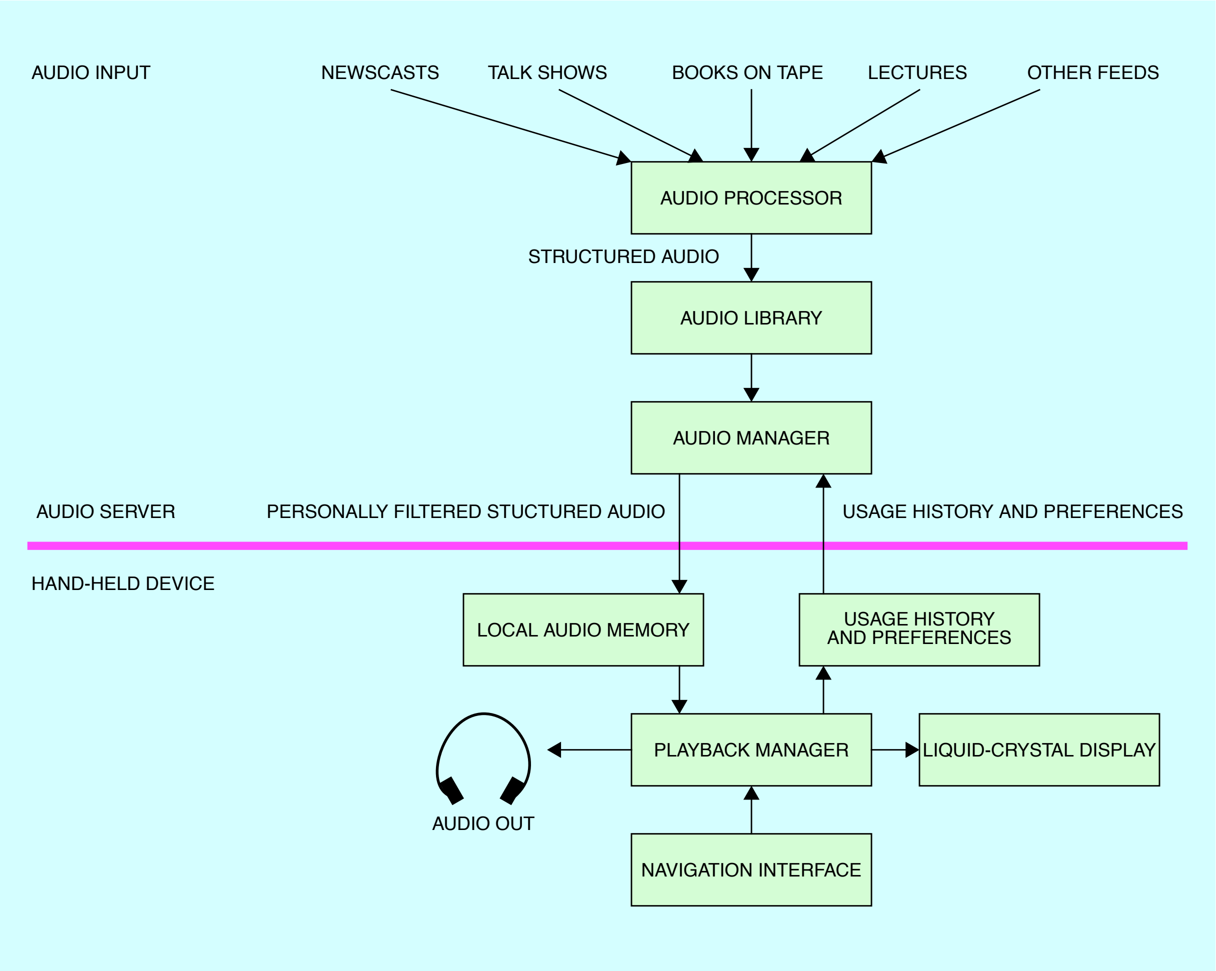
Once the download is complete, the user can disconnect from the server and interactively access the recordings using the navigation interface of the hand-held device. The playback manager in the hand-held device uses the structural description of the audio to enable efficient navigation of the recordings. It does this by ensuring that when the user wishes to jump forward or backward in a recording, the jump "lands" in a meaningful place rather than a random one. The structural description of each recording contains the location of all suitable jump destinations within the recording.

Acoustic structure

NewsComm uses two sources of evidence for structure in the audio recordings: pauses and speaker changes. These aspects of the audio programs are computed for each sound file by the server and then downloaded into the hand-held device together with the recording. This section describes how each feature is extracted.

Pause detection. The speech recording is segmented into speech and silence by computing the energy of 64-ms (millisecond) frames. Once the energy distribution has been computed, the 20 percent cutoff is found, and all samples of the recording that lie in the bottom 20 percent of the distribution are labeled as silence; the remaining 80 percent of samples are tagged as speech. This assumes a 4:1 ratio of speech to silence in the audio recording that has been found to be an acceptable approximation for other professionally recorded speech (including books on tape and newscasts from other sources) through empirical

Figure 9 An overview of the audio server and hand-held playback device



observations made during the development of the algorithm.

Once the 20-percent threshold has been applied, single-frame segmentation errors are corrected: any single-frame segments (i.e., a single frame tagged as speech surrounded by silence segments or vice versa) are removed.

Locating speaker changes. An algorithm called speaker indexing (SI) has been developed to separate speakers within a recording and assign labels, or indices, to each unique speaker. This is in contrast to the speaker identification task in which prior samples of each potential speaker are available. The current NewsComm system only uses locations of speaker changes and ignores speaker identity, although identity information may be used in the future. The SI algorithm is briefly described in this section (see Roy[22] for a more detailed description). Each speaker change boundary is located, and indices are assigned to each segment that are consistent with the original identities of the speakers. Since the SI system has no prior models of the speakers, it does not identify the speakers, but rather separates them from one another within the recording.

An important distinction between speaker indexing and conventional speaker identification is that there is no assumed prior knowledge about the speakers in the input recording. In speaker identification, a set of models of all possible speakers is created using training samples of each speaker. Identification of an unknown sample is performed by comparing the speech sample to each speaker model and finding the closest match. For the class of applications we are considering here, we cannot assume the *a priori* availability of training data for speakers. Thus, conventional speaker identification techniques cannot be directly applied.

The speaker indexing algorithm. The speaker indexing algorithm dynamically generates and trains a neural net to model each postulated speaker found in the recording. Each trained neural net takes a single vowel spectrum as input and outputs a binary decision indicating whether the vowel belongs to the speaker or not.

Signal processing. Audio is sampled at 8 kilohertz (kHz). A Fast Fourier Transform (FFT) of the input signal is computed using a 64-ms hamming window with 32-ms overlap. The resultant spectrum is passed through a mel-scaled filter bank that produces a 19 coefficient spectral vector. In the time domain, a peak picker estimates the location of vowels by picking peaks of the energy of the speech signal (vowels have relatively high airflow and thus a corresponding peak in the energy contour).

Only the mel-scaled spectra corresponding to each vowel are output to the neural network portion of the system. By discarding nonvowels, the possible set of sounds that must be modeled by the neural network is reduced to only English vowels, thus reducing the amount of training data required to train the neural network.

Although most vowels in the recording will occupy more than a single 64-ms frame, the current implementation only selects the single frame corresponding to the center of the energy peak.

Training the neural networks. The SI system employs back propagation neural networks to model each postulated speaker in the input recording. Back propagation neural networks are trained through a supervised process.[23] For a network with binary output, a set of positive and negative training examples is required. The examples are presented in sequence to the network. The weights of the network are adjusted by back-propagating the difference between the output of the network and the expected output for each training example to minimize the error over the entire training set.

If the positive training examples are a subset of the vowels spoken by some speaker X, and the negative examples are a subset of the vowels spoken by all the other speakers, we can expect the trained network to differentiate vowels generated by speaker X from all other speakers (including vowels that were not in the training set).

However, since there is no *a priori* knowledge of the speakers, training data must be selected automatically. This selection process begins by assuming that the first five seconds of the recording were spoken by a single speaker, speaker 1. The spectra of the vowels from this five-second segment comprise the positive training data for the first neural net. A random sampling of 25 percent of the remainder of the recording is used as negative training data. Note that the negative training set selected in this manner will probably contain some vowels that belong to speaker 1, leading to a suboptimal speaker model.

Once the neural network has been trained using this training set, the network is used to classify every vowel in the recording as either belonging to speaker 1 or not (true or false).

The resultant sequence of classification tags is then filtered to remove outliers. Filtering is accomplished by applying a "majority rules" heuristic. Let us define the term "sentence" in this context to be a segment of speech terminated at both ends by a pause. The minimum length of this pause is a manually set parameter. (We found 0.2 seconds to work well for broadcast news.) To filter the tags of a sentence, we count the number of occurrences of each tag in the sentence and then replace all of the tags with whichever tag occurred more often. This filtering process has two effects: (1) possible false-positive tags generated by the neural network are removed, and (2) vowels that were not recognized as speaker 1 are "picked up" in cases where the majority (but not all) of the vowels in a sentence were positively tagged. This filtering process partially compensates for errors in the training set.

A second filter is then applied, which ensures that any sequence of tags shorter than the *minimum speaker turn* is inverted. The minimum speaker turn is defined manually and depends on the nature of the audio being processed. We found a setting of five seconds appropriate for broadcast news since a speaker will rarely talk for less than five seconds. The setting would have to be lowered for conversational speech since speaker turns might be shorter.

Once the two levels of filters have been applied, the neural network is retrained. All the vowels that have been classified as speaker 1 (after filtering) are collected and comprise the new positive training set, and again 25 percent of the remaining vowels (randomly selected) comprise the negative training set. This entire training, tagging, and filtering cycle is repeated until no further positive training vowels are found.

Once the first speaker has been located, the audio corresponding to that speaker is removed from the input recording, and a new neural network (for speaker 2) is created and trained on the remaining audio using the same procedure. This cycle is repeated until all audio in the input recording has been indexed.

Experimental results

The accuracy of the speaker indexing algorithm has been tested on two sets of data. The first is a set of ten 20-minute British Broadcasting Corporation (BBC) newscasts recorded over a two-week period. Each recording contains about 15 unique speakers. The second test set contains six 15-minute clips of TechNation interviews.[24] Five of the TechNation clips contain two unique speakers, and the remaining clip contains three speakers. Speaker changes and indices were hand-annotated for each recording and used as references for measuring the accuracy of the automatic indexing. A set of test software has been written that runs the speaker indexing software in batch mode on all recordings in a test set and computes average accuracy scores across the entire set by comparing the output of the indexing program to the manual annotations.

Accuracy has been measured in three ways for each test set:

1. Speaker indexing: the number of frames of the recording that were indexed correctly as a percentage of the total number of frames
2. Speaker change hits: the percentage of speaker changes that were detected by the algorithm with an error of less than 1.0 second
3. False alarm percentage: the percentage of speaker changes detected by the algorithm that were not classified as hits

Table 1 Experimental results of the indexing algorithm (all values are percentages)

Test Set	Indexing Accuracy	Speaker Change Hits	False Alarms
BBC newscasts	64	50	55
TechNation	89	57	57

The results are shown in Table 1.

In the current nonoptimal implementation of the speech-processing algorithm, a 30-minute audio news program recording requires approximately three hours of processing time on a Sparcstation 10 workstation.

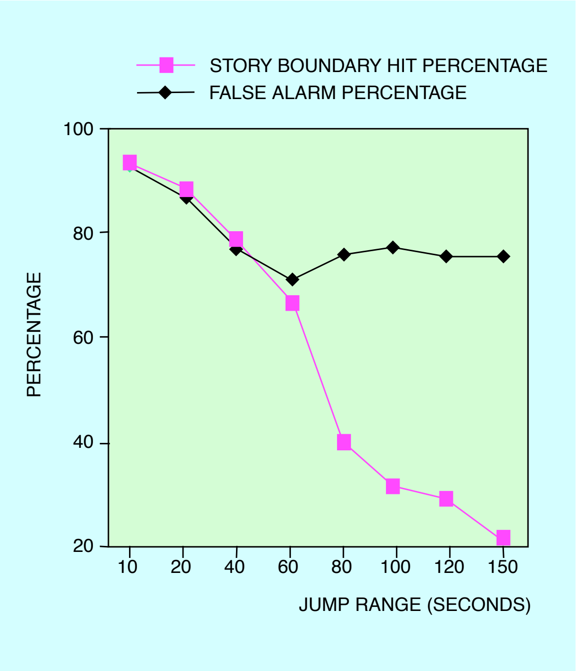
Discussion. The indexing algorithm has a relatively high error rate for all three measures. We believe that the main reason is the training initialization process that uses random selection of negative data for training the neural nets. Analysis of the algorithm shows that in many cases a poor choice of initial training vectors causes segments of a recording that belong to a single speaker to be fragmented and assigned multiple indices. This leads to a drop in indexing accuracy and a rise in the false alarm rate. Similarly, poor training data can also cause different speakers to be collapsed into one neural net model. This situation leads to a drop in speaker change hits and indexing accuracy.

It is important to note that although the error rates are high, the system does locate half or more of the speaker changes in recordings. The NewsComm interface has been designed with the assumption that the structural description of the audio has errors. Even with the given error rates, in practice the NewsComm hand-held device has proved to be an effective navigation device when speaker indexing output is combined with pause locations.

Annotation framework for user instruction

The goal of a structured representation is to have “handles” into a large media stream. If placed in meaningful or salient locations, these handles can be used to increase the efficiency of browsing and searching the stream. NewsComm chooses the location of these handles by combining information about pause and speaker change locations. Long pauses usually predict the start of a new sentence, a change of

Figure 10 A plot of the number of story boundaries in the BBC test newscasts versus the jump range of the annotation framework



topic, a dramatic pause of emphasis, or a change in speaker.[25] Speaker changes can be useful when listening to an interview, conversation, debate, or any other recording containing multiple speakers.

All iterations of the NewsComm interface design use the fundamental notion of jumping to facilitate navigation functions including skimming and searching. The framework of pause and speaker changes allows jump locations to be placed at any level of granularity. The jump locations can be used by applications to enable efficient access to the contents of the recording. Recordings can be skimmed by playing short segments following each jump. Recordings can be summarized by extracting and concatenating speech segments following each jump location. Note that the interface does not need to know how the jump locations were chosen, thus the design of the interface is isolated from the underlying annotations.

NewsComm defines a salience metric to order potential jump points within some range. Speaker changes by definition are assigned maximum salience. The salience of each pause is proportional to its duration; the longest pause is assigned a salience equal to a speaker change.

The salience measure is used by NewsComm to place jump locations. Given a position within a recording, the next jump location is chosen by finding the frame with the highest salience within the jump range. Thus the jump range controls average jump size within a recording. If the jump range is set to zero, every frame becomes a jump location. At the other extreme, if the jump range is set to the size of the entire recording, only one jump location will be selected: the frame with the highest salience across the entire recording.

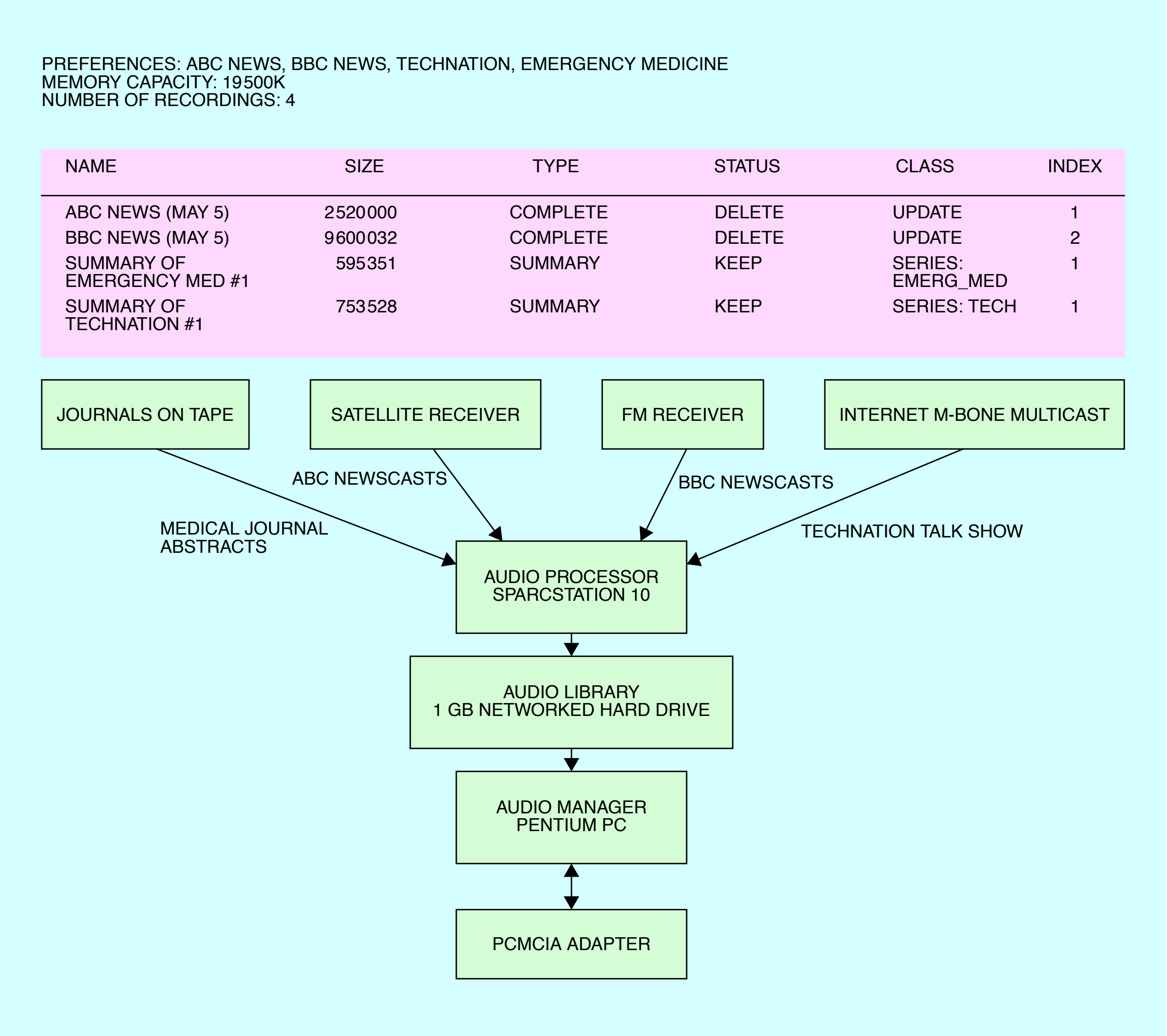
The jump location selection process may be thought of as sliding a window across the recording. We start with the window positioned so that the left edge of the window lines up with the start of the recording (the recording is laid out from left to right). The length of the window corresponds to the jump range. To select a jump location, we find the frame within the window with maximum salience. We then slide the window over so that the left edge lines up with the jump location we just selected. We repeat this process of picking the next jump location and sliding the window until the window reaches the end of the recording. To jump backward, the window is placed to the left of the play position instead of the right and slid to the left after each jump location is chosen.

The use of the jump range concept ensures even temporal coverage of the recording. Even if all of the most salient frames of a recording are located in the first half of the recording, the framework guarantees coverage of the second half as well.

The effect of jump granularity on story boundary detection in BBC newscasts. An experiment was conducted to study the effect of jump granularity on the number of story boundaries identified as jump locations by the framework. Story boundaries are desirable points to locate in a newscast since the user can browse the recording by jumping between stories. The locations of all story boundaries in four 20-minute BBC newscasts were manually annotated. A jump location is considered to coincide with (or hit) a story boundary if they occur less than 1.0 second apart.

Ideally the jump locations would coincide with only story boundaries. The assumption, based on empirical observations of the newscasts, is that speaker changes and long pauses usually coincide with story bound-

Figure 11 Components of the current NewsComm audio server



PREFERENCES: ABC NEWS, BBC NEWS, TECHNATION, EMERGENCY MEDICINE
MEMORY CAPACITY: 19500K
NUMBER OF RECORDINGS: 4

NAME	SIZE	TYPE	STATUS	CLASS	INDEX
ABC NEWS (MAY 5)	2520000	COMPLETE	DELETE	UPDATE	1
BBC NEWS (MAY 5)	9600032	COMPLETE	DELETE	UPDATE	2
SUMMARY OF EMERGENCY MED #1	595351	SUMMARY	KEEP	SERIES: EMERG_MED	1
SUMMARY OF TECHNATION #1	753528	SUMMARY	KEEP	SERIES: TECH	1

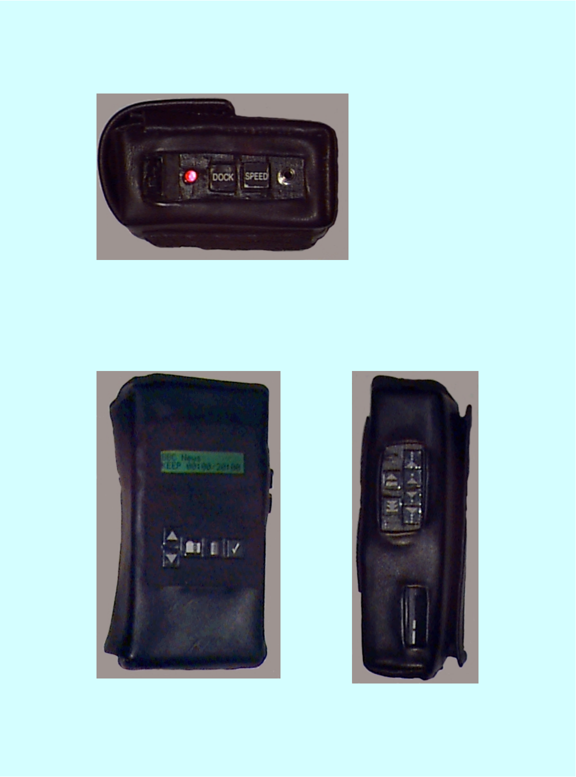
aries. Figure 10 shows the results on the four 20-minute BBC newscasts. The line marked with squares shows the percentage of story boundaries located as a function of the jump range. As expected, the two are inversely related. The line marked with diamonds shows the false alarm rate of the jump locations. The false alarm rate is the percentage of all jump locations that do not occur at story boundaries.

The false alarm rate dips at a jump range of 60 seconds. This is a reasonable jump range setting to use for accessing this type of recording since the false alarm rate is at a minimum (70 percent) and the story hit percentage is relatively high (67 percent).

The audio server

The audio server collects, structures, and stores audio recordings. When the hand-held device is connected to the server, the server receives the user's listening history and preferences, which are used to select the next set of recordings to download to the local memory of the hand-held device.

Figure 12 Details of the top, front, and right sides of the final hand-case, which incorporates Version 5 of the interface design



The user's preferences are represented as a list of preferred information sources. New content from those sources is given priority during downloads to the hand-held device. Longer recordings are automatically summarized using highlight selection. The listener can request complete recordings after hearing summaries of interest.

Figure 11 shows the components of the NewsComm audio server. The audio processor currently receives audio from four sources:

- A satellite dish receives newscasts from American Broadcasting Corporation (ABC) radio. Newscasts are received hourly and are five minutes long. Only one newscast per day is currently placed in the NewsComm server.
- A conventional FM radio receiver receives a daily 20-minute BBC newscast (rebroadcast from England by a local FM radio station).
- A series of medical journal abstracts was digitized from cassette tape and stored in the server. These journals-on-tape are commercially available and are typically purchased by physicians who listen to them while driving or performing other tasks.[26]
- TechNation is a weekly talk show available over the Internet.[24] This talk show is an hour long and is multicast over the Internet multicast backbone. Due to memory limitations of the hand-held prototype (40 minutes capacity), only the first 15 minutes of each show are stored in the server.

Audio recordings from each of these sources are processed by the algorithms described earlier. The structural descriptions (pause and speaker change locations) are encoded in ASCII files and stored with the corresponding recordings in the audio library. Summaries of the medical journals and TechNation talk shows have been generated and stored as separate recordings. An index file that lists all the available recordings in the library is generated. The audio manager uses this file to access the contents of the audio library. The library is a one-gigabyte networked hard disk drive.

The audio server supports two classes of recordings: *updatable* and *series*. Updatable recordings include newscasts, weather, and any other continuously updated information in which only the latest version of the information is usually of interest. In the current implementation, the ABC and BBC newscasts are classified as updatable. Series are ordered sets of recordings such as the chapters of a book on tape or a sequence of interviews from a talk show. TechNation and the medical abstracts are examples of series recordings. One-of-a-kind recordings are a special case of the series class with a set size of one.

Usage history, preference information, and an index of the recordings in the local memory of the hand-held device are all stored in a table of contents (TOC) file. The TOC is originally generated by the server and downloaded to the hand-held device along with a set of audio recordings.

The TOC file is read by the hand-held device after disconnecting from the server so that it knows what recordings are present in its local memory. When the hand-held device is next connected (docked), it will generate a modified version of the TOC containing updated usage information. The server then reads this TOC file and thus receives the updated usage information from the hand-held device.

Table 2 Icons, names, and functions of the buttons of Version 5

Flat Mount	Leather Case	Name and Function of Button
◀◀◀	•••◀	COARSE-JUMP-BACK: Jump back at coarse granularity; hold down to do a coarse-level backward skim.
◀◀	•◀	FINE-JUMP-BACK: Jump back at fine granularity; hold down to do a fine-level forward skim.
▶▶	▶•	FINE-JUMP-FORWARD: Jump forward at fine granularity; hold down to do a fine-level forward skim.
▶▶▶	▶•••	COARSE-JUMP-FORWARD: Jump forward at coarse granularity; hold down to do a coarse-level forward skim.
⏮	⏮	REWIND: Move the play position to start of the recording.
⏯	⏯	PLAY/PAUSE: Toggle between playing and stop.
SPEED	SPEED	SPEED: Designed to switch between three preset playback speeds, but not currently implemented.
△	▲	MENU-UP: Select the next recording in the menu.
▽	▼	MENU-DOWN: Select the previous recording in the menu.
KEEP	📁	KEEP: Keep the current recording at next connection with the audio server.
DEL	■	DELETE: Delete the current recording at next connection with the audio server.
REQ	✓	REQUEST: Download the full version of the current recording (this button works only when the current recording is a summary).
	DOCK	DOCK: Press this button before connecting the hand-held device to the audio server, and again after disconnecting.

The audio manager runs on a Pentium PC that has access to the audio library over a local area Ethernet network connection. The PC has a PCMCIA adapter attached so that it can write files onto the PCMCIA flash card from the hand-held device.

The NewsComm hand-held unit

Hardware implementation. The hardware implementation is based on the HP95LX palmtop computer. The HP95 was programmed in a combination of C and assembly language (for the audio drivers). Figure 12 shows a detailed three-way view of the device. The front panel houses the recording selection and management controls and a 2 × 16 character liquid-crystal display (LCD) screen, and the right side houses the navigation controls. The opening at the bottom of the right side provides access to the memory card eject slider. The case is made of soft leather and measures 7.5 inches (h) × 3.75 inches (w) × 2.0 inches (d). The icons presented in Table 2 have been printed onto the button controls on the side and face of the interface.

The interface was implemented in two versions of hardware (the first was mounted flat, the second was in a leather case). The icons for each implementation are listed in the two columns on the left side of the table.

Not shown in Figure 12 is the slot at the bottom of the device for inserting and removing a PCMCIA flash RAM memory card. Audio is transferred to the hand-held device by removing this card from the device and inserting it into a card writer attached to the audio server.

A basic design decision for all the hardware designs was to put local memory in the hand-held device for audio storage since it is relatively simple to implement. Alternatively the device could be a cellular-telephone-type device that uses a two-way point-to-point wireless connection to relay audio from the server and send navigation commands back. This wireless model is more difficult to implement but is certainly a feasible alternative for realizing NewsComm. In fact, in a commercial version it may be cheaper to use the wireless model if there are enough users.

A sample interaction using the hand-held device interface. This section presents the transcript of a sample interaction using the NewsComm interface. Button presses are indicated in italics. In this example, the user first listens to portions of an ABC news broadcast and then switches to an interview. Prerecorded prompts that announce the names of recordings are underlined.

<PLAY/PAUSE> ABC News: “In Bosnia today there was more fighting ...”

<FINE-JUMP-FORWARD> “President Clinton says the health care bill will ...”

<COARSE-JUMP-FORWARD> “In other news, New Jersey Senator Bill Bradley says he will not run ...”

<FINE-JUMP-BACK> “In other news, New Jersey Senator Bill Bradley says he will not seek a fourth term, handing the Democrats another blow to their fast-fading hopes of reclaiming control of the Senate in next year’s elections ...”

<MENU-DOWN> TechNation March 15, 1995 “This is TechNation with Moira Gunn. Today we are talking with Bill Gates ...”

<MENU-DOWN> TechNation, May 10, 1995 “This is TechNation with Moira Gunn. Today we speak with Richard Dawkins, author of the Selfish Gene ...”

<COARSE-JUMP-FORWARD> “I first got interested with Darwin’s theories while in a local pub at Oxford ...”

<PLAY/PAUSE>

Interface design. The interface for the hand-held device was developed through an iterative design and testing process. To establish the initial set of functions for the device we had a round-table discussion with a group of journalists who are experienced with the problems of accessing audio from conventional tape recorders. The main problems they complained about were the inability to search and browse efficiently. Their specific comments were taken into consideration when we designed the initial interfaces.

Five versions of the hand-held interface were designed over a five-month period. The first design was implemented in hardware as a proof of concept (mainly to test hardware components of the system for the hand-held device). Because of the cost of building custom hardware, the next three iterations were implemented and evaluated using a graphical interface built in Tcl/Tk (an interpreted rapid prototyping language and environment). Although one of the goals was to design a nonvisual interface, the on-screen Tcl/Tk simulations of the interface allowed us to explore different potential functions that might become part of the final interface design. The final Tcl/Tk interface was then “ported” to a hardware implementation that could be used with one hand and minimal visual attention. For details on the interface design process see Reference 27.

To summarize the interface design process, the authors found a tendency to expose increasing control to the underlying indexing abilities of the system, but usability studies consistently showed that a simple interface retaining only basic indexing control was preferred. Casual observations have shown that users can much more easily understand and use the final interface than the initial interface.

For example, early versions of the interface include a skimming mode that automatically plays only selected highlights of a recording. Additional controls enable the user to adjust parameters of the skimming mode. Although users generally agreed that the skimming mode is useful, they consistently preferred a simpler interface with no second modality; the users could achieve the same effect of skimming by manually pressing a jump forward button after hearing each highlight. Although this method of skimming requires more input from the user, it reduces the confusion of having a hidden mode, and it gives direct navigation control to the user.

The final interface consists of four navigation buttons for skipping at coarse and fine levels in forward and reverse, a button to start and stop playing, and a button to "rewind" to the beginning of a recording. These six buttons are mounted on the side panel of the device and can be operated with the user's thumb. The top panel of the hand-held device contains several buttons for selecting and specifying preferences for each file. Table 2 lists the controls of the final interface and describes the function of each. Although we wished to support audio time compression in the NewsComm unit, the processor in the HP95LX was not fast enough to support it.

Conclusions

NewsComm is an example from a series of projects and ongoing work seeking to make audio recordings with voice as a data type more accessible and useful to listeners. NewsComm focuses on *portability* and timeliness to justify the use of audio recordings over other media. It defines a structure for interacting with these recordings, based on pauses and speaker changes. NewsComm implements a user interface based on physical buttons in a hand-held device to allow nonvisual interactive control of playback, suitable for mobile users.

The model of acoustical structure for NewsComm incorporates both speaker changes and pauses. The speaker indexing algorithm is able to detect about half of the speaker changes in speech recordings. Several potential causes for the errors have been identified including poor initial selection of training data and a suboptimal representation of the audio signal. Even with the present error rates, the speaker indexing algorithm was successful in enabling efficient navigation when combined with pause locations.

The framework for combining annotations was successfully used to combine the output of the speaker indexing algorithm and pause detection. The framework was used to provide jump locations for five different interface designs demonstrating the separation of interface design from the underlying representation of the media.

Implementing the hand-held design in hardware was a useful reminder of the inseparable relationship of function and form. Although the interface was first implemented in software, its usefulness in this form is limited since it relies on a visual display for output; in this environment it is more efficient to visually skim and read information than to listen. The hardware version demonstrates a truly portable device that provides a function that a visual display system does not: interactive access to information for mobile users using a nonvisual interface.

Acknowledgments

We received help and useful comments from Jeff Herman, Sandra Bunch, Nicola Shaw, Josh Bers, Sara Elo, Nicolas Saint-Arnaud, Walter Bender, Bob Donnelly, John Makhoul, Jack Driscoll, Sumit Basu, Vadim Gerasimov, Lisa Stifelman, and Yin Yin Wong. Lena Davis helped edit the manuscript.

This work was supported by the MIT News in the Future consortium.

**Trademark or registered trademark of Sun Microsystems, Inc. or Intel Corp.

Cited references

1. C. Schmandt and B. Arons, "A Conversational Telephone Messaging System," *IEEE Transactions on Consumer Electronics* **CE-30**, No. 3, xxi–xxiv (August 1984).
2. B. Chalfonte, R. Fish, and R. Kraut, "Expressive Richness: A Comparison of Speech and Text as Media for Revision," *Proceedings of the Conference on Computer Human Interaction*, ACM (April 1991), pp. 21–26.
3. D. Hindus, C. Schmandt, and C. Horner, "Capturing, Structuring, and Representing Ubiquitous Audio," *ACM Transactions on Information Systems* **11**, No. 4, 376–400 (October 1993).
4. T. W. Malone, K. R. Grant, K. Y. Lai, R. Rao, and D. Rosenblitt, "Semi-Structured Messages Are Surprisingly Useful for Computer-Supported Coordination," *TOIS* **5**, No. 2, 115–131 (April 1987).
5. F. Chen and M. Withgott, "The Use of Emphasis to Automatically Summarize a Spoken Discourse," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Vol. 1 (1992), pp. 229–232.
6. L. J. Stifelman, "A Discourse Analysis Approach to Structured Speech," *AAAI 1995 Spring Symposium Series*, Palo Alto, CA (March 1995).
7. H. Gish, M. Siu, and R. Rohlicek, "Segregation of Speakers for Speech Recognition and Speaker Identification," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Vol. 2 (1991), pp. 873–876.
8. L. Wilcox, D. Kimber, and F. Chen, "Audio Indexing Using

Speaker Identification," Xerox PARC ISTL Technical Report No. ISTL-QCA-1994-05-04 (1994).

9. J. B. Voor and J. M. Miller, "The Effect of Practice upon the Comprehension of Time-Compressed Speech," *Speech Monography* **32**, 452–455 (1965).
10. D. S. Beasley and J. E. Maki, "Time- and Frequency-Altered Speech," *Contemporary Issues in Experimental Phonetics*, N. J. Lass, Editor, Chapter 12, Academic Press, New York (1976), pp. 419–458.
11. L. Degen, R. Mander, and G. Salomon, "Working with Audio: Integrating Personal Tape Recorders and Desktop Computers," *CHI*, OCHI, New York (1992), pp. 413–418.
12. C. Schmandt, "The Intelligent Ear: A Graphical Interface to Digital Audio," *Proceedings of the IEEE Conference on Cybernetics and Society* (October 1981), pp. 393–397.
13. D. Kimber, L. Wilcox, F. Chen, and T. Moran, "Speaker Segmentation for Browsing Recorded Audio," *CHI '95 Conference Companion*, Denver, CO (May 7–11, 1995), pp. 212–213.
14. A. S. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*, MIT Press, Cambridge, MA (1990).
15. D. A. Norman, *Memory and Attention: An Introduction to Human Information Processing*, John Wiley & Sons, New York (1976).
16. L. J. Stifelman, "Not Just Another Voice Mail System," *Proceedings of the 1991 Conference*, American Voice I/O Society (September 1991), pp. 21–26.
17. C. Schmandt, "Caltalk: A Multi-Media Calendar," *Proceedings of the 1990 Conference*, OAVIOS (1990), pp. 71–75.
18. B. Arons, "Hyperspeech: Navigating in Speech-Only Hypermedia," *Hypertext '91 Proceedings*, ACM (December 1991), pp. 133–146.
19. B. Arons, *SpeechSkimmer: Interactively Skimming Recorded Speech*, Ph.D. thesis, MIT Media Laboratory, Cambridge, MA (1994).
20. C. Schmandt and A. Mullins, "AudioStreamer: Exploiting Simultaneity for Listening," *CHI '95 Conference Companion*, Denver, CO (May 1995), pp. 218–219.
21. T. G. Zimmerman, J. R. Smith, J. A. Paradiso, D. Allport, and N. Gershenfeld, "Applying Electric Field Sensing to Human-Computer Interfaces," *CHI '95 Conference Proceedings*, Denver, CO (May 1995), pp. 280–287.
22. D. Roy, *NewsComm: A Hand-Held Device for Interactive Access to Structured Audio*, M.Sc. thesis, MIT Media Laboratory, Cambridge, MA (1995).
23. D. Rumelhart, G. Hinton, and R. Williams, "Learning Representations by Back-Propagating Errors," *Nature* **323**, 533–536 (1986).
24. Information available by sending e-mail to *technation@usfca.edu.*
25. D. O'Shaughnessy, "Recognition of Hesitations in Spontaneous Speech," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing* (1992), pp. 1521–1524.
26. *Emergency Medical Abstracts*, G. Hasapes, Executive Editor, Center for Medical Education, Harleysville, PA (1995).
27. D. Roy and C. Schmandt, "NewsComm: A Hand-Held Interface for Interactive Access to Structured Audio," *CHI '96 Conference Proceedings*, Vancouver, Canada (April 1996), pp. 173–180.

Accepted for publication June 14, 1996.

Chris Schmandt *MIT Media Laboratory, 20 Ames Street, Cambridge, Massachusetts 02139-4307 (electronic mail: geek@media.mit.edu).* Mr. Schmandt received the B.S. and M.S. degrees from MIT, where he has been building speech systems since 1979. He is the director of the Speech Interfaces Group at the Media Laboratory, a position he has held since the creation of the Lab. His current research focuses on user interfaces and applications of speech processing technology, voice as a data type on workstations and hand-held computers, and computer-mediated telephony.

Deb Roy *MIT Media Laboratory, 20 Ames Street, Cambridge, Massachusetts 02139-4307 (electronic mail: dkroy@media.mit.edu).* Mr. Roy received the B.A.Sc. degree in computer engineering from the University of Waterloo in 1992, and the M.Sc. degree from the Media Laboratory in 1995, where he is currently a Ph.D. candidate. His research interests include automatic language acquisition, multimodal interfaces, speech and speaker recognition, and audio structuring and retrieval.

Reprint Order No. G321-5617.