Chapter 5

Voice Communication with Computers*

Christopher Schmandt
Media Laboratory,
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

INTRODUCTION

Speech is an attractive channel for a range of communication tasks with computers. (Beek, Neuberg, & Hodge, 1977; Lea, 1980; Martin, 1976; Ochsman & Chapanis, 1974). Speaking and listening are universal abilities, whereas typing remains a skill which must be learned and practiced. Voice interaction may be less intimidating to the computer naive population, allowing them greater machine access. Voice is powerful in hands-and-eyes-busy work environments, or locations where a convenient terminal or keyboard may be impractical. Voice input and output can allow remote database access via ordinary telecommunication channels.

To the extent a computer interface can employ natural language, voice is the obvious command channel; indeed, people often talk to machines even when they have no expectation of being understood! Speech synthesis, which is much easier to incorporate into an application environment than speech recognition, allows for conversational systems. Merely adding speech input/output to an application does not make a "user-friendly" interface, however. Voice interaction may be conversational and forgiving, or terse, staccato commands and replies. That the latter is in fact far more common is a reflection of the capabilities of current technology, and in part explains the very limited number of voice systems in actual use.

*Portions of the work reported in this chapter have been supported by the Defense Advanced Research Projects Agency, the Office of Naval Research, Atari, Inc., and NTT, the Nippon Telegraph and Telephone Company.

The author wishes to credit Eric Hulteen, Walter Bender, and Barry Arons for joint participation in the work described herein. It has been a pleasure working with them.

To participate in a dialogue, a conversational interface requires natural language understanding and high recognition accuracy as input, and intelligent, as well as intelligible, response capacity. What is required is not as much recognition, that is, the correct *identification* of particular spoken words, as understanding, determining the *meaning* or *intent* of those words.

This chapter considers design implications and performance criteria for a conversational speech interface. Such an interface is presented as a semi-autonomous intermediary between the human user and whatever application software is being accessed. In arguing the utility of such a model, a brief review of speech technologies will be given to indicate hardware weaknesses which must be compensated for. Next, system constraints, that is, design limitations of the interface which hamper dialogue, will be briefly examined. This serves as a prelude to discussion of three projects, implemented in the author's laboratory at the Massachusetts Institute of Technology, which attempt to overcome aspects of both hardware and system design limitations.

1 UNDERSTANDING IN THE INTERFACE

Human speech processing is aided by knowledge and constraints. Language has structure, an underlying syntactic form (Chomsky, 1957, 1965) which both limits the meaningful combinatorial arrangements of a collection of words, and also allows some prediction of expected word patterns. The topic of a conversation limits the domain of relevant utterances; discrimination of acoustically similar speech sounds is aided by our expectation of what the speaker may be saying. Prosodics, including stress and intonation, convey both overall syntactic information and also subtle shades of meaning (Brown, Currie, & Kenworthy, 1980; Lea, Medress, & Skinner, 1975). Knowledge of the topic allows the listener to make assumptions about a speaker's intent, filling in misunderstood words or expanding acronyms. Additionally, a variety of interactive techniques (such as quizzical expressions, eye movements, and interruption) allow for an undercurrent of information flow concerning the degree of mutual understanding (Brown & Yule, 1983).

The fact that current speech recognizers make many more mistakes than human listeners only underscores the need for a *systemic* or *holistic* approach to the conversational interface. A speech recognizer is not another "black box" device to replace a keyboard in a particular program by plugging it into the otherwise unchanged i/o routines. Likewise, the slower speed and variable intelligibility of synthesized speech indicate that direct substitution for a CRT screen is not practical for output.

It is useful to distinguish two general approaches to speech understanding and its relationship to word recognition. The more promising method, in the long run, integrates syntactic analysis into the word recognition process, so that knowledge of both sentence structure and the domain of conversation are used in the process of analyzing acoustical information for word detection. Although successful systems have been demonstrated for some time, perhaps the most well known being the DARPA Speech Understanding Project (Klatt, 1977), generalizable real time systems have yet to appear. Although some progress is being made in incorporating these constraints into commercial recognizers, current implementations are still difficult to program and rather inflexible.

The second, a less integrated approach, takes whatever output is provided by the recognizer, including such data as runner-up word matches and relative scores, and then applies understanding algorithms to this, as a separate, higher level of processing. Such an approach is more realizable, as it is "device independent," and not part of the internals of a specific recognition algorithm; hence it may be compatible with any of a number of already available recognizers.

Although the separation of acoustic and syntactic processing may not be optimal for word recognition accuracy, significant improvements are still possible (Levinson, 1978; Levinson, Rosenberg, & Flanagan, 1978) and the resulting program modularity aids system design. Most important for *interactive* systems is that such separation allows understanding tasks to operate at a higher level in the system, with options for communication and feedback through other devices, such as voice response.

This chapter discusses some design aspects of a model of interactive systems based on the second of these two approaches, applying analysis techniques after recognition. At issue is the linkage between speech processing hardware, either for input or output or both, the dialog management components of the human interface, and the application software or database being accessed.

Limitations in the voice channel preclude a direct connection between voice hardware and application software. Some of these limitations, to be discussed in the next several sections, are specific to current hardware, for example, the accuracy of speech recognition or the intelligibility of synthesized speech. Others are inherent in the channel, for example, the relatively slow rate of speech as contrasted with reading speeds and CRT baud rates.

The model to be discussed includes a range of functionality; it is a conversational intermediary between speech hardware and the application, and acts as a filter between them. It includes hardware drivers, as low level timing information or the ability to interrupt output hardware is extremely important. Language understanding algorithms in this module have access to the application database to aid in context-sensitive semantic analysis.

This module is responsible for all user feedback and prompting short of actually performing the requested task. It modulates information transfer between the user and the application; voice input is buffered and analyzed, perhaps with some assumptions made, or perhaps stimulating a dialogue, until a complete and coherent command can be issued to the application. For output, the interface module may adjust speech rate, suspend buffered speech output in request to a new input, or re-order text to facilitate intelligibility and minimize short term memory loads.

Rather than discuss this model in an abstract framework, within which it has never been formulated, this chapter will illustrate its utility and evolution through several examples drawn from work developed at the Architecture Machine Group of M.I.T. These projects are presented as examples of specific techniques to be embedded in the interface. A brief overview of the speech technologies will be presented, as it is their limitations which in large part necessitate this work. A summary of interface design flaws follows, as interface design is crucial for overcoming these limitations.

2 SPEECH RECOGNITION

Speech recognition provides the capability to monitor speech and identify words from some small known vocabulary. From the point of view of interface design, a number of different classes of recognizers may be differentiated. Recognition may be accomplished by a variety of methods, (Rabiner & Levinson, 1981; Reddy, 1976), but all approaches manifest similar *types* of errors.

Speech recognizers may be classified by functionality in several ways. These include speaker dependent versus independent, connected versus discrete speech, and trainable versus fixed vocabulary templates.

SPEAKER INDEPENDENT recognizers are designed to understand any reasonably articulate native speaker of the language, much as one may be expected to understand a stranger asking for directions. A SPEAKER DEPENDENT recognizer is more particular, able to understand only the individual who trained it. In general, speaker independence is much harder to achieve, and these recognizers exhibit smaller vocabulary and/or poorer recognition. Note, however, that as vocabulary size becomes very large (e.g., an automatic transcribing typewriter), speaker independence becomes necessary, as training several thousand words could be a very tedious task!

Speaker dependence is often considered an unfortunate limitation of the technology, but it may in fact be beneficial in some circumstances. In group interactions, the vocabulary of a single recognizer may be partitioned, with each section trained by a separate person. Later, a word match reveals not only what was said, but also the speaker. A measure of security is provided by speaker dependence, as individuals will most likely experience rather poor performance unless they have trained the templates.

DISCRETE SPEECH recognition requires each word to be spoken separately, with a slight pause inserted between, as "*go . . . to . . . the . . . door.*" Note, though, that a *word* can really be a short, quickly spoken phrase, such as "*answer the phone.*"

CONNECTED SPEECH recognition takes longer *sentences,* or groups of words spoken naturally, and identifies each individually. Connected speech is clearly more natural and faster; discrete speech hardware exists only as a manageable simplification of the problem, as it is much easier to recognize isolated words.

Several acoustical problems make connected speech recognition difficult. The first is segmentation of the input into word units. Acoustically, this may be very difficult, as the pauses between syllables of a single word are often longer than the pauses between words in a sentence. The second, strongly related, is coarticulation; the initial and final pronunciations of words are heavily modified by the sounds that precede and follow them (Klatt & Stevens, 1973; Oshika, Zue, Weeks, Nue, & Aurback, 1975). Hence, longer words are easier to recognize in connected speech, as more information remains invariant in various acoustic contexts.

Reference templates for recognition may be TRAINABLE OR PREDETERMINED. Trainable templates, a characteristic feature of speaker dependent recognizers, are formed from one or more spoken samples of the word to be recognized. Training may be single pass, during which the word is spoken once, captured and converted into some internal representation (see below). Alternatively, multiple passes may be utilized, in an attempt to emphasize features that remain constant among them to improve recognition (Rabiner & Wilpon, 1980). Some metric of acceptable variance of sounds within the word may be incorporated into the template.

Most systems provide for loading and restoring trained templates, either to a host computer or on internal storage media, such as floppy disks or bubble memory. For a given speaker, a well-trained template set is likely to change little over time.

Other recognizers use predetermined reference templates. This tends to be the case in any speaker independent system, as the templates must reflect the range of allowable pronunciations or perhaps key acoustic features in an utterance. Such templates are usually created in laboratory environments. A special class of such fixed template recognizers are experimental systems which detect phonemes or acoustic feature information (Chen & Zue, 1984), and compare it with phonemic descriptions of utterances; these descriptions may be considered to be templates predetermined by the language.

2.1 Recognizer Components

Despite the variety of methods and algorithms for speech recognition, several basic components can be identified in all recognizers: an internal representation of the audio signal, reference templates in this representation for comparison, and a metric of similarity between the templates and a speech sample.

An INTERNAL REPRESENTATION of the speech signal is required for comparison with the vocabulary templates. This may be a series of time samples from the output of digital or analog filter banks or it may be derived directly from the digitized audio signal, for example, zero crossing spacing. The signal may be analyzed into coefficients of a Fourier or other transform, or Linear Predictive Coding parameters (see below) may be extracted. The presence of significant acoustic events, such as stops, plosives, and voicing, may be employed.

REFERENCE TEMPLATES, against which speech input will be matched, must be created before recognition can be performed. Templates are the universe of words to be recognized, as encoded in whatever internal representation of the speech signal is utilized.

As mentioned above, speaker dependent templates are trained by a particular individual. Speaker independent templates are generally produced in a laboratory, as it is difficult to extract the key features which remain invariant across a large speaker population. Phonetic based recognizers utilize a template description somewhat akin to the phonetic spelling of a word.

Finally, a METRIC is needed to quantify the differences between captured speech and the set of stored templates (Itakura, 1975), and determine whether the minimal error found is less than a rejection threshold. This will probably involve power normalization, and often involves some form of *dynamic programming* (also referred to as *dynamic time warping*) (Nakagawa, 1983; Sakoe, 1979), to stretch or compress the length of the spoken utterance to compensate for variable speech rates. Some configurations allow the templates to be divided into subsets, and this algorithm restricted to matching against various of these subsets over time, as indicated by the host.

2.2 Recognizer Limitations

Recognizer errors may be of three types. A word in the trained vocabulary may be seemingly properly spoken, and no match within tolerable limits found by the recognizer. An acceptable word may be spoken and erroneously matched against some other word. An extraneous word, perhaps just the clearing of the throat or a heavy breath, may be mistakenly matched against the known words.

Due to the variety of error modes, accuracy may be difficult to quantify (Pallet, 1982). Such statistics are meaningless if not standardized, and even then are probably useful only for ranking recognizers, not for prediction of performance in actual use. Indeed, performance will vary as a function of the similarity between words in the vocabulary, and one must be aware of the confusability of words when selecting an input set (Rosenberg, 1983). Acoustic context such as background noise level or even microphone placement may alter performance dramatically (Dautrich, Rabiner, & Martin, 1983; Rabiner & Levinson, 1981). Performance will vary greatly from speaker to speaker, depending in large part on one's willingness to accommodate speech style to recognizer capability, by speaking slightly more slowly and consistently.

It is certainly safe to expect that recognizers will never be more accurate than the human ear, which we know to be error prone. For real-world applications, except perhaps in very small vocabulary and acoustically controlled situations, accuracy is far less than a system designer would like. It is significant to note that although recognizers have become less expensive, smaller, commercially available, and supportive of larger vocabularies, there has not been any dramatic breakthrough in recognition algorithms over the last several decades. Errors must be anticipated in human–computer interface design, and some means provided to enhance recognizer performance.

3 SPEECH PRODUCTION

Voice output provides the capability for a machine to respond to some action or situation vocally. Although the term "speech synthesis" is often used, this in fact properly describes only some of the technologies available. There are as many techniques of creating voice output as there are ways of storing and defining speech sounds.

3.1 Output Technologies

Speech output may consist of playback of previously recorded speech, or synthesis-by-rule from unrestricted text. The former may be stored as a direct recording, or compressed by waveform or parametric encoding.

The simplest means to provide speech output for a computer is to directly *record* some human speaker saying all the possible responses the system can make, and allow the machine to play back whichever segment is appropriate. This analog recording could be stored on a variety of media: magnetic tape, magnetic disk, optical videodisc, or digital audio discs. Although high quality may be obtained, these approaches tend to be costly, simply because of the mechanical components involved.

The same audio may be DIGITIZED and stored in computer memory as a series of samples of the speech waveform, with reproduction through filters and digital-to-analog converters. This approach, PCM (Pulse-Code Modulation), may be less expensive than analog, as no mechanical components are involved, and will in general be easier to interface to a digital computer. The main drawbacks are memory requirements and data rates; one minute of toll telephone quality speech requires nearly a half megabyte of storage.

Storage requirements may be reduced through a variety of data compression schemes (Flanagan, Schroder, Atal, Crochiere, Jayant, & Tribolet, 1979; Rabiner & Schafer, 1978). The simplest approaches, WAVEFORM CODERS, use alternative descriptions of the speech waveform, such as the difference between samples, (Delta Modulation), pause removal (Maxemchuk, 1980), or run-length encoding similar periods of the signal. In general, these techniques take advantage of the relatively slowly varying nature of speech for data compression. Some schemes, such as CVSD (Continuously Varying Slope Delta modulation), are common enough to be readily available as integrated circuits.

Greater data compression may be obtained using PARAMETRIC CODING, or analysis/synthesis techniques. This broad range of frequency domain approaches is characterized by use of an alternative, reversible description of the signal, often based on a model of the vocal tract. Parameters are extracted and later used as input to the inverse operation, to reproduce the spectrum of the original, rather than matching the signal sample by sample. Some signal quality is necessarily lost; the parameterized representation of the waveform is based on a model of speech production that is itself only an approximation to real speech.

An example of such analysis techniques is the Fourier transform, with varying number of coefficients stored or transmitted. Linear Predictive Coding, in a number of variations (Markel & Gray, 1979) produces other coefficients, describing the sound as a linear function of previous samples. Data rate is a function of not only how many bits per parameter, but how many parameters per sample. As should be expected, for any given technique, a higher bit rate may be associated with higher quality reproduction. As these techniques tend to be developed specifically for speech coding, bizarre outputs may be produced from non-speech inputs (coughs, music, etc.).

In general, the analysis or parameter generation stage is significantly more demanding than the synthesis operation. Until recently, LPC and related coding schemes were used in low cost applications, such as speaking games, with parameter generation done carefully and synthesis done by inexpensive chips. Recent developments in VLSI signal processing chips (Secrest, Arjmand, & Ni, 1982), however, have made possible inexpensive analysis as well as synthesis.

SYNTHESIS-BY-RULE, a very different technique, produces speech directly from ASCII text strings. An algorithm containing rules of pronunciation for the particular language must be included to generate *phonemes* (the various speech sounds making up a language) in terms of duration, energy, and voice resonances or *formants*. In English, this is far from simple, and a dictionary of algorithm exceptions is often included (Allen, 1976).

A model of the vocal tract as a series of filters and resonators (Klatt, 1977) is excited by the pronunciation algorithm. More advanced approaches also attempt to model clause and sentence level prosody, including pauses, pitch contours, and stress (Anderson, Pierrehumbert, & Liberman, 1984; Pierrehumbert, 1981).

This purely synthetic speech requires the least storage and lowest transmission bandwidth, as it may be stored as ASCII text strings and data rates of 300 baud or less can produce continuous speech. The outstanding advantage of text-to-speech conversion is its utility in situations where one may not know in advance what text is to be spoken, such as an electronic newspaper, reading electronic mail, or remote database access.

3.2 Synthetic Speech Limitations

Four main problems are encountered with synthetic speech employed for computer-generated dialogue; its overall quality, word intelligibility, the speed of speech, and the fact that speech is serial or time sequential in nature.

In general, speech QUALITY across the compression schemes deteriorates as data rates are reduced. As quality degrades, artifacts may be noticed in the speech, such as a wavering or buzzing, and it becomes difficult to recognize the particular speaker. Synthesized speech bears its own distinct acoustic personality; in fact, all presently available synthesis-by-rule hardware sounds remarkably similar.

There are times when this mechanical-sounding speech (either from synthesis or the lower rate coding schemes) may be useful; for example, a pilot may be speaking over the radio to humans, but it would be clear that a synthetic voice interrupting is a message from the plane itself. In other cases, it is important for people to realize that a machine is speaking as part of an automatic process, to which they would respond differently if a human were present.

The primary concern with voice output must be INTELLIGIBILITY, which may prove difficult to quantify (Miller, Heise, & Lichten, 1951). Modified Rhyme Tests (House, Williams, Hecker, & Kryter, 1965), a common metric, play back a single word, and subjects are asked to identify it from a list of similar sounding words; this is particularly useful for evaluating the correctness of text-to-speech rules. But scores on such a test may not accurately reflect average intelligibility, as one may miss several words of a sentence and still understand the whole. Listeners are very likely to have difficulty understanding unfamiliar words or acronyms reproduced under lower bit rate speech.

As a listener is exposed to a particular synthetic speech peripheral, and becomes accustomed to it, misunderstanding errors decrease significantly, much as one improves in ability to understand a regional or foreign accent (Slowiaczek & Pisoni, 1982). Studies indicate, however, that even though word-by-word understanding may become fairly high, this takes some effort, such that a listener is less likely to comprehend the meaning of the sentence or paragraph being spoken (Luce, Feustel, & Pisoni, 1983). This suggests short answers of several words, to allow maximum effectiveness.

Of particular note in applications using synthesis is the necessity for clause and sentence level prosody generation in the synthesizer (McPeters & Tharp, 1984). If voice messages exceed one or two sentences, the monotonous pacing of less sophisticated synthesizers inhibits syntactic perception, which in turn limits one's ability to use syntactic context to improve understanding. The least expensive synthesizers are barely intelligible to previously unexposed listeners.

An important consideration for the use of speech output is its SPEED. Speech is quite slow as compared to computer terminal baud rates and human reading speeds. In situations requiring rapid response, voice output must be kept terse. In applications necessitating reading a large amount of text (e.g., accessing a database by telephone) any extraneous information should be removed before it is spoken to the user.

Finally, in applying speech synthesis to applications which formerly used CRT screens, the TEMPORAL NATURE of speech must be kept in mind. Although one may display a number of menu options simultaneously on a screen, they must be spoken sequentially under voice access. If there are a number of choices, the first may well be forgotten before the last is spoken, especially in light of the concentration required for understanding synthetic speech. Such a list of recited options is necessarily serial, whereas the eye can (and does) skip from place to place on the page or screen scanning for the required information.

4 HUMAN–COMPUTER COMMUNICATION

Although the claim has been made that speech has widespread potential for human–computer communication, the previous sections indicated that the technology is far from flawless. Most noteworthy are the high error rates of recognizers and intelligibility as well as speed constraints imposed by synthesizers. Equally important are constraints which may be caused by improper design of the interface between this hardware and an application. A poorly designed interface can render error-prone hardware useless; conversely, the interface can counteract errors and actually make the hardware appear more accurate.

This section critiques the interface from the point of view of *communication*. Communication is a minimal prerequisite for interactive systems; the person must realize what the computer has decided it is supposed to do, and the computer must be able to learn what information is of any interest to the user. Any barrier to this communication detracts in some degree from the effective utilization of these resources as an aid to the user.

Four aspects of the interface will be discussed: language, accuracy, intelligence, and failure modes. Each of these constrains communication but, alternatively, suggests desirable system behaviors for improved interaction. Subsequent sections will offer examples of this design philosophy in actual use.

4.1 Language Constraints

This use of the term "language" includes both the universe of active *words* as well as the rules of syntax for concatenating them into meaningful *sentences*. The most obvious constraint on voice communication with a computer is its small VOCABULARY. Although current speech recognition hardware is of course limited to a finite vocabulary, there is no reason why the content of that vocabulary should not be user-selected, or task-dependent, and perhaps even dynamically and transparently reconfigurable in the midst of the speech-driven application.

From the system designer's point of view, it is desirable to choose a vocabulary set which would maximize recognizer performance. Words which may be easily confused would be avoided, as well as short words which contain less acoustic information and are therefore harder to identify. From the user's view, however, such choices may sound unnatural (e.g., "affirmative" for "yes") or prove difficult to remember (e.g., "new sequence" for "clear").

SYNTAX is another language constraint. The leap from intuitive natural English syntax to that required by any computer language is one of those incredible barriers separating the "naive user" from the computers he or she may be using. Voice interactive systems should strive for as natural a grammar as possible. Every syntax rule of a command language is a limit which only helps emphasize the fact that the user is talking to a limited machine. The use of an appropriate word with multiple meanings should not be prevented, but rather the proper meaning should be derived by the machine from the context in which it is used.

4.2 Accuracy Constraints

The accuracy or error rate of speech recognition hardware is the primary constraint to voice input. Undetected or misrecognized words are a serious limit to the effectiveness and acceptance of speech input technologies. As discussed earlier, speech recognizers are prone to a variety of errors, and such hardware errors will not vanish in the near future, manufacturers' claims to the contrary.

The critical factor is the error rate *as it is perceived by the user*, which may be very different from a raw percentage recognition score. If a recognizer performs superbly on every word except the one needed to enter each command string, any user will quickly feel constrained to the point of removing the microphone. Conversely, context-sensitive software may be able to resolve or interpolate around some of the hardware errors, dramatically enhancing system performance in the speaker's perception.

Similarly, pronunciation errors by text-to-speech synthesizers, or poor intelligibility of low bit rate speech coding, hamper communication from computer to the user. In contrast to speech input, however, the user is aware of these errors and can elicit a repetition, perhaps spoken more distinctly, to correct it. End-to-end communication may be enhanced by providing a means of interrupting and clarifying, much like a conversation.

4.3 Intelligence Constraints

A third major component is intelligence, a minimal prerequisite for communication. Intelligence is manifest in responsiveness or feedback, through which the interface indicates what it has understood or needs to know, and without which the user can have only marginal confidence that the system is even listening.

An intelligent system is capable not only of indicating what it has heard, but, equally important, what it needs to know in order to understand. This implies sensitivity to context, an awareness of the task at hand, and inputs and responses which are germane to it. Context can

become a powerful tool to assist recognition, as well as providing the guidelines to direct dialog toward task completion. In order to do its job well, the interface must understand the job it is trying to do.

Intelligence is also important in making real-time decisions about the outward flow of data on a voice channel. While the image of reams of paper spewing out of a line printer is amusing, listening to a synthesizer recite the same output is not! Text must be sorted and filtered by the interface, which must also remain responsive to redirections of requests in progress in response to further user queries or commands.

4.4 Failure Constraints

There will always be recognition errors, and some of those errors will not be recoverable; what happens then? The simplest and least communicative failure mode is no response; the system and user remain in a loop, the system not reacting until the right word is recognized, and the frustrated user repeating this magic word until by some coincidence it is recognized. The user may or may not even realize what the system is waiting for.

More encouraging are helpful responses, which in some manner coax the user to supply missing information on the most direct path to completing plausible commands. A clever system may even realize from the timing or pattern of inputs that the user is confused, and provide spontaneous coaching. Failure may not even be irritating in a chatty interaction, being so much absorbed into the personality implied by dialogue.

Three systems implemented at the Architecture Machine Group will be described in the following sections. Each was designed to explore penetration into various combinations of these areas of constraint. The first is the most encompassing, dealing with many aspects of the conversational interface; indeed the interface *is* the research. The second and third systems deal respectively with context-driven vocabulary selection for recognition and database query organized for speech output.

5 PUT THAT THERE

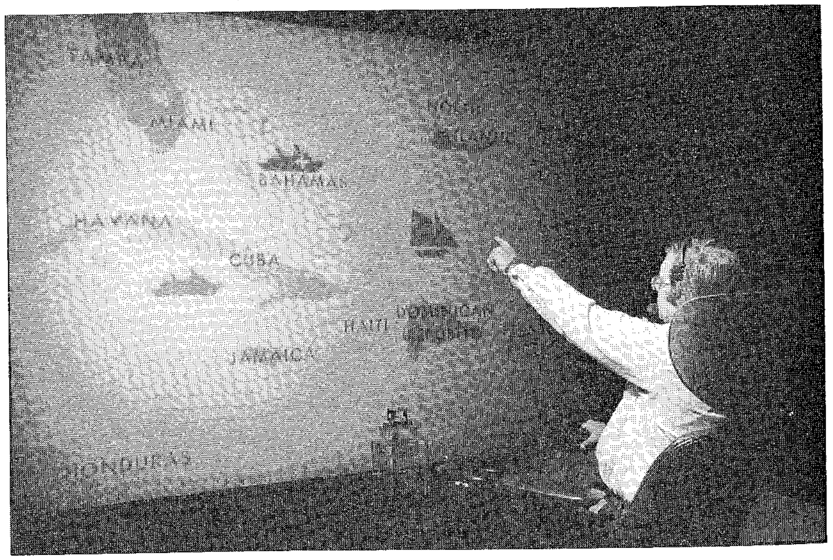
Put That There (Bolt, 1980; Schmandt & Hulteen, 1982) began as an exploration of interactivity based on multi-modal (voice and gesture) input, and rapidly "digressed" into an exploration of a conversational user interface. This change was driven by the necessity of coping with error-prone speech recognition in a complex command and control environment. The goal was to provide as natural an interface as possible under the circumstance of hardware constraints. It was through this work that the model of a distinct interface module, with intelligence and a life of its own, was first developed.

The user sits in a comfortable chair before a thirteen-foot diagonal rear-projected video screen to run Put That There (Figure 1). Using a connected speech recognizer (Kato, 1980) and pointing with a magnetic digitizer having six degrees-of-freedom (Rabb, Blood, Steiner, & Jones, 1979), one moves ships around on a computer-generated map of the Caribbean. One can create, move, change color, copy, or delete the ships, as well as name them. A relational geographic database allows directional specification of ships, that is, one can *"create a green freighter north of the red oil tanker."* The underlying map can be selected from a small library, or annotated upon, and the system vocabulary can be expanded "on the fly" with spoken synonyms.

Of note is not so much the particular application as the style of interaction, and its interplay with the speech technologies. The user and the machine converse, with the machine engaging the user in a conversation after an initial request, according to its own syntactic and semantic model of the shipping world. An example of such an interchange:

"Move the red cruise ship. . ."
  "Which object?"
"cruise ship"

Figure 1. "Put That There" in the Media Room



  "Where?"
"north of Jamaica" ⟨action takes place⟩

Dialogue is the key element. With early versions of the system, before speech output, one would talk to a microphone, waiting for something to happen. If no action occurred, there were few cues as to which words were misrecognized, leading to frequent repetitions, which sometimes resulted in very ambiguous syntax. Underlying the dialogue are a number of techniques which result in an apparently higher recognition accuracy then the hardware actually accomplishes; software takes several steps, perhaps simple, but nonetheless effective, towards speech understanding. The following section will analyze these techniques and their interplay.

5.1 Implementation Techniques

Inherent in the design of this interface is a communication module, involving both hardware access and software, and clearly sensitive to the context of the particular application. This module *mediates* between the user and the application (to which it may be thought of as a "front end"), manifesting intelligence to the user and passing control to the application only when fully satisfied that the user's intent is well understood. Such an interface component will clearly sustain a dialog with the user, and may well also query the application on its current state or history, to aid its understanding of the speech.

5.1.1 Speech Analysis. The first component of this mediating interface is syntactic and semantic analysis of whatever speech is recognized. Although perhaps an intimidating prerequisite, the task is vastly simplified by the limited domain of a relatively small vocabulary and the known context of the application task. A wide range of work has been done on syntactic analysis (Winograd, 1983); though it tends to be too general-purpose to be cost effective for a small application, many simplifications would be made to provide robust systems for small computers.

SYNTACTIC analysis in Put That There is accomplished by mapping each word into a class, and an instance of that class, for example, the class may be *command* and the instance *move.* The syntactic parser scans incoming speech, building a data structure that not only describes which parts of speech are present, but also their syntactic relationship, for example to differentiate the role of the ships in *"Move the freighter north of the sailboat."* In this sense, it is reminiscent of some of the earliest natural language processing systems (Bobrow, 1967; Green, Wolf, Chomsky, & Laughery, 1963; Lindsay, 1973) which took advantage of domain specificity to limit complexity of analysis; the syntactic structure for Put That There would be inappropriate for another application.

SEMANTIC analysis is action-driven, that is, each verb, or command, has a corresponding subroutine, to test the syntactic structure for completeness. Once a unique command is found, this lower level specifies what information, if any, is missing, and may call other routines to look for alternate sources of information. At this stage, second guess information and confidence scores from the recognizer can be invoked (Levinson & Shipley, 1980). This approach is very amenable to the addition of new commands without disturbing existing semantic analysis routines.

If information is still needed, a spoken request—for example, "*Which object?*" or "*Where?*"—is spoken, and semantic analysis suspended. Syntactic analysis is re-entrant, by virtue of the syntax data structure; the parser simply updates it when additional input is available. Hence control may pass back to the top level where more speech input is gathered and parsed into the structure, and semantic analysis begun anew. This is vital for error recovery, and is a significant step toward graceful failure, as the interface takes an *active* role in detecting and correcting errors.

The syntactic analysis structure is a compact representation of the current state of knowledge about the user's input. The presence or absence of specific word classes, for example, *command, location* or *directional modifier,* can be used to initiate additional stages of analysis, as described below. Syntactic content also provides the ability to distinguish the meanings of the command *make* in "*make an oil tanker there*" versus "*make that red.*"

5.1.2 Redundant Input Channels. An auxiliary source of information to aid speech processing is input which may be arriving on other perhaps redundant and also noisy channels. It is for this reason that speech is seen as an addition, rather than a replacement, for other modes. The semantic analyzer of Put That There utilizes gesture as a second source of input; pointing to an object, an explicit, intentional motion, serves to select it even if the spoken "that" was never recognized. Not only is it useful to be able to balance workload by selecting between input channels, it also increases the probability of a correct transaction.

5.1.3 Knowledge-Based Assumptions. More of the missing speech may be deduced by the semantic analyzer from constraints imposed by the particular application. A knowledge-based system may pare down the universe of syntactically possible commands dramatically, and indeed may even be able to make reasonable assumptions of the user's intent. A simple example is the awareness that the only action one can do to a nonexistent object is create it, and Put That There assumes such when a command is missing. More sophisticated analysis incorporates knowledge of interaction history to aid object selection or placement.

5.1.4 Non-vocal Feedback. An essential component of a responsive interface is *feedback.* As speech is often to be used in "hands and eyes busy" situations, the visual or tactile media may be considered a *data channel* with audio as a *control channel.* Different styles of feedback are apropos to each function. In Put That There, an example of data channel feedback is a ship changing color whenever the system has concluded the user has selected it. The purpose of this style of feedback is to communicate whatever conclusions the interface has made with respect to the application as soon as possible, at the locality of the user's attention.

Feedback should always be immediate. Even if the system will take some time to perform a task, some response should be the first step. The earlier feedback is provided, the sooner errors can be detected or corrected by the user. But, equally important, any feedback gives the user confidence of being heard, and motivation to continue the dialogue.

5.1.5 Voice Response. It is natural for a computer to which one speaks to talk back. Clearly the best feedback is to actually do the task; the worst is to say nothing, or nearly as bad, respond with "*I didn't understand.*"

The syntactic data structure indicates which words are missing or ambiguous, and can be used to guide the dialogue toward those components. Questions are, as a result, direct and avoid generalities, indicating as much as possible about system understanding. For example, "*What object?*" conveys a missing operand, while "*Which one?*" shows that more than one object matching the description has been found.

A noteworthy benefit for a connected speech environment is gained by this approach of querying for single-word replies. As noted earlier, connected recognition is much more difficult than discrete, due to coarticulation and segmentation difficulties, so a single-word response to a specific question is much more likely to be recognized. This is significant for user satisfaction; the worst time to make a recognition error is immediately after making a recognition error!

5.1.6 Dynamic Vocabularies. A fact of life in speech recognition is limited vocabulary size; in some cases it may be desirable for performance to purposely limit size even further. Intelligence can make this constraint felt as less of an intrusion on the user. As already suggested, syntactic analysis is a tool allowing synonyms as well as multiple meanings for the same word as a function of context, thereby loosening vocabulary constraints. Dynamic vocabularies allow the flexible addition of different words, which may reflect changes in the task environment, such as the appearance of a new data object, or merely accommodate user speaking style idiosyncracies.

Of course, it is always possible to change the vocabularies or retrain templates offline. The desire here is to modify them in real time, in the midst of and under the operation of the application. This can be done with recognition hardware which has two features: templates must be trainable in a single pass, and recognizer functions must be computer controlled. In Put That There two commands allow the creation of new templates. The *name* command trains an utterance which is then associated with a particular object. The *designate* command—as in *"Designate move . . . transpose"*—allows user definable synonyms for any word in the vocabulary.

5.1.7 Transaction Memory. System responsiveness and cooperativeness can be improved by keeping track of the transactions it has performed. In the ideal case, if either the system or the user has made an error, the user can request the computer to *"Undo that."* Commands can even be stored on a stack, and popped off until the proper one is found. Memory in Put That There is object-oriented; each ship remembers its prior location and attributes, so it can be put back or restored in a single command.

6 VIRTUAL VOCABULARY

Speech recognizers are limited in vocabulary size. The price of a larger vocabulary is not just an increase in template memory, which is relatively inexpensive, but the corresponding loss in recognition speed, as a greater number of patterns need to be compared for each utterance. Search time is compounded with connected speech, as an utterance consists of a combination of words, each of which must be compared with the complete list of words during pattern matching.

Error rates are also linked to vocabulary size. As the number of words to be recognized increases, it becomes more likely that several words will have similar features (Dixon & Silverman, 1981; Rabiner, Rosenberg, Wilpon, & Keilin, 1982; Rosenberg, 1983). The mean distance between words, by whatever scoring metric is used, must decrease as vocabulary size grows; hence, it may be useful to limit the number of words in order to improve performance.

Many applications for which voice would be a useful channel have potentially very large vocabularies. In some of these applications, however, only a relatively small subset of the known words need be available at

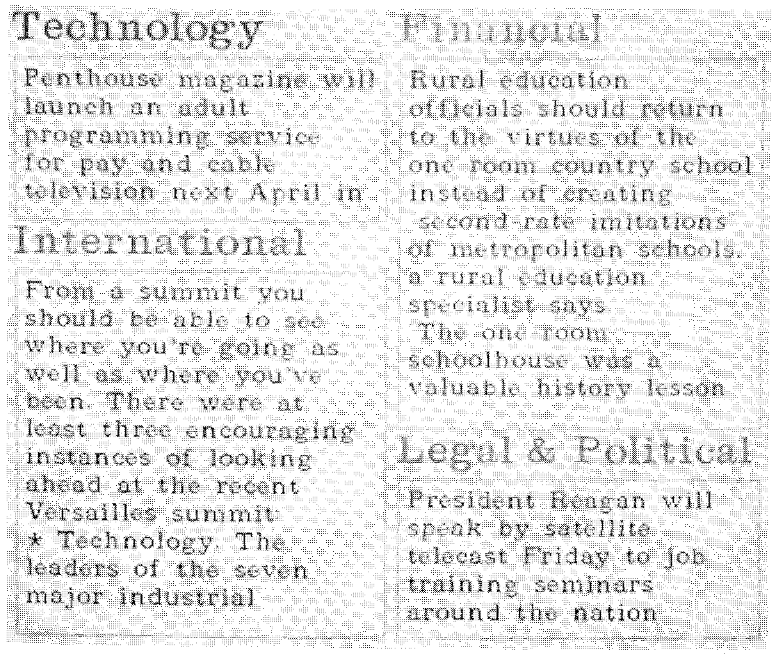
any time. This will be the case when the application involves clearly differentiable subtasks or topic areas; while one subtask has been invoked, there will be a *locality of reference* such that words germane to that subtask can be expected, while others may be ignored. Of course, there must be a global vocabulary, consisting at least of commands to move from one task to another, active at all times. If one could change resident templates rapidly under control of the application, the *virtual* recognizer could, under ideal circumstances, appear to have an unlimited vocabulary.

This concept has been applied to a very large vocabulary speech-accessed electronic newspaper, NewsPeek, in another Architecture Machine Group effort. Hardware for the virtual recognizer consists of a personal computer with modified, programmable recognition routines and disk storage for templates, all interfaced to a larger host computer.

6.1 The Electronic Newspaper

NewsPeek is a personal electronic newspaper being developed to explore new techniques for browsing and analysis of news information. Virtual vocabulary speech recognition is used in NewsPeek to manage a

Figure 2. A page from the electronic newspaper



Technology

Penthouse magazine will launch an adult programming service for pay and cable television next April in

International

From a summit you should be able to see where you're going as well as where you've been. There were at least three encouraging instances of looking ahead at the recent Versailles summit: * Technology. The leaders of the seven major industrial

Financial

Rural education officials should return to the virtues of the one room country school instead of creating second-rate imitations of metropolitan schools, a rural education specialist says The one room schoolhouse was a valuable history lesson

Legal & Political

President Reagan will speak by satellite telecast Friday to job training seminars around the nation large vocabulary of single-word utterances (Pathe, 1983; Schmandt & Bender, 1983). The recognition scheme employs knowledge of both the application and the user's interaction with the system.

NewsPeek combines a remote database search system with local computing, storage, interaction, and display. A computer surrogate directs search operations within a large database and creates a display with layout similar to a daily newspaper (Figure 2). Included are wire services such as AP and UPI, magazines, newspapers, journals, and the Encyclopedia Britannica, all accessed digitally by telephone. The computer takes on the editing tasks normally associated with news information and perusal, using knowledge of the reader's interests and prior transactions to create a personalized news report.

This is an attractive domain for virtual vocabulary segmentation, as there are clearly separable sub-vocabularies derived from each of the visible stories and some follow-on stories currently invisible. A correlation database exists as a means of mapping from visible to other, archived, stories. Vocabulary subsets are triggered by the same process that selected the text to be typeset.

6.2 Implementation Issues

Much like virtual memory computer systems, a virtual vocabulary recognizer has two requirements: SECONDARY STORAGE for templates not in active use, and an ALGORITHM to select which templates should be swapped into the scarce resource of active vocabulary memory.

In speech tasks, one is most likely to think of the host as the site of SECONDARY STORAGE, that is, to store the virtual vocabulary templates there, and download pieces of it to the recognizer over whatever interface is provided. Note that words are read into memory from secondary storage much more often than they need be written out, as this is necessary only when the data have been modified after a training session.

The weak link in this configuration is the interface between the host and the recognizer, which is often too slow to transfer large amounts of vocabulary data in real-time. This data transmission also occupies the host, which has other tasks to do meanwhile; in the electronic newspaper, this consists of searching internal databases for relevant material and typesetting articles on the graphics screen. Swapping excessive amounts of vocabulary information would detract significantly from the newspaper's responsiveness.

To avoid communication overhead, a virtual recognizer was programmed on a personal computer. Floppy disks are used for secondary vocabulary template storage, and recognition is done with a modified recognizer board that plugs into the processor memory bus. The microprocessor manipulates filter bank hardware, performs pattern matching, and monitors a serial interface; this bi-directional line is used to report back word recognitions as well as initiate template swaps. A high level protocol simplifies host communication, responding to powerful commands such as block transfer of templates between disk and memory, and initiation of memory compaction.

The second requirement of a virtual memory system is the SWAPPING ALGORITHM to control the mechanics of memory management. This algorithm, implemented on the host, provides for both the segmentation of the virtual vocabulary into logical blocks and the rules for assigning priority to blocks. It can be optimized by close integration with the host application task and knowledge of the user; this specificity and personalization are the points of departure from standard computer memory management schemes.

The NewsPeek application suggests several natural lines for segmentation of the virtual vocabulary. The local library of stories is organized by topic, providing a convenient way of associating groups of words with groups of stories; each word also has an associated pointer to related topics. Priority assignments for blocks are generated by the NewsPeek algorithms which sort the news for browsing. The vocabulary predicter assigns each block a priority by rules dependent upon the current state of the system to maximize the likelihood of anticipating user requests.

Typical blocks are the basic NewsPeek command set, the last five words selected by the user, trained words found in the current story, trained words found in related stories, and trained words found in stories read previously. Blocks are swapped out by a weighted least-recently-used algorithm, but many possible refinements are currently being researched. Temporal, statistical, and page layout information maintained by the NewsPeek system is available to the swapper. Frequency and periodicity of words can be an additional weighting factor. Is this the first appearance of a word in the paper? Does this word appear only on Mondays? Is the word a keyword or in the lead paragraph?

Not only is this virtual vocabulary segmentation effective, it further suggests that dynamic personalization of large vocabulary sets may enable higher quality recognition by successful limitation of active template set sizes. Again, it is the integration of hardware, algorithms, and the user's workload which can make speech recognition an attractive interface component.

7 VOICE ACCESS TO TEXT DATABASES

In the two previous interface examples, emphasis was on speech recognition and the associated constraints of accuracy and vocabulary size. This section focuses on speech output, as used for remote (telephone) access of text databases. Four problem areas need be addressed by interface design. As already discussed, both INTELLIGIBILITY and SPEED of synthesized speech interfere with data transfer. Speech is necessarily SERIAL, which hampers menu presentation and the conceptual framework within which data are presented to the user. With telephone access, a further consideration is that the single channel must be MULTIPLEXED FOR BOTH DATA AND CONTROL functions.

7.1 Voiced Mail

A goal of much speech work is easier *access* to information. The telephone system provides a ubiquitous network optimized for voice communication. The combination of the two provides extremely powerful computer access.

The Voiced Mail system at M.I.T. allows users of an electronic mail system to call in on a telephone and have their messages read by a text-to-speech synthesizer. The primary motivation for such a system is convenience of access; electronic mail is gaining popularity in part because it enables quick and reliable communication between parties or groups in disparate locations and perhaps on differing schedules. To use conventional electronic mail systems one must carry a terminal and modem to read and send messages. A telephonic voice interface removes this restraint.

This project is actually a component of a wider research activity in telephone access to a variety of databases and multi-media message systems in a telecommunications environment. Major portions of the Voiced Mail system have been incorporated as is into a *personalized* text and voice message storage and answering machine (Arons, 1984; Schmandt & Arons, 1984a, 1984b).

To use Voiced Mail a user calls in and gives a unique identifer (home phone number) and a password by Touch-Tones, much like using an automatic bank teller. Messages are sorted by source, and by default, played sequentially; the caller may interact with the system by either Touch-Tones or spoken commands, to jump to the next message or next sender, obtain more information about the sender, repeat a sentence, or make a reply (Figure 3). Touch-Tones are used as a backup to speech recognition due to occasionally noisy phone connections.

Several types of replies may be generated. The caller may send back electronic message of the form "I read your message about ⟨*subject line*⟩ and the answer is **"yes,"** or **"no,"** or **"please call me at** ⟨*a telephone number*⟩," which is then keyed in. A recent addition allows the reader to record a voice message for the sender, with mail sent back explaining how to get access to this reply, including a personal code for security.

Figure 3. "Voiced Mail" Telephone keypad command layout.

1 Next Message	2 Previous Message	3 Repeat
4 Next Sender	5 Previous Sender	6 More Info
7 Yes	8 No	9 Reply
* Cancel	0 Pause/ Continue	# Quit

7.2 Interface Issues

The initial issue to be faced in the design of such a system is organizational, directed by limitations caused by exclusive use of the speech channel. A conventional mail system uses a text screen formatted for particular fields in constant positions, such as message headers (which may be quite lengthy) and text body. The reader can scan the message in fairly random order, for example, look in more detail at the header if the "postmark" on the message is important, or re-read a sentence, or skip ahead a screenful. With spoken messages this must all be done sequentially, with more explicit commands.

The key to managing this slow serial interface is to reduce the amount of information transmitted, and organize the presentation such as to minimize short-term memory requirements on the listener. Messages are accessed not in conventional time-sequential order, but rather according to sender, with all the messages from a single source treated as a group. Commands, such as *next* or *previous*, can operate on a single message or a group.

Most header information, including the sender's full name and mailing address as well as message data, is not transmitted by voice, although it may be accessed by a *more info* key. The data is spoken in a natural form, for example, "this morning at 10:30" or "Monday at 8:10 P.M.," rather than the computer-standard "month-date-year hour-minutes-seconds" format.

Another organizational aid is a *pause/continue* command, allowing speech to be stopped at any moment; it is resumed from the beginning of the current sentence. Note that this time-to-text coordination requires both an interface which is always listening as well as two-way communication between the synthesizer and host, as there is a processing delay from receipt of the text by the synthesizer and the corresponding speech output.

Speed issues are relevant to any system transmitting a large amount of information by voice. Speech is slow enough that it quickly becomes tedious to listen to long passages of irrelevant information. A certain amount of filtering can be of assistance; for each message the user is told the subject and queried as to whether the whole message should be played; a very short message, however, is just played. Another helpful cue is provided by the synthesizer warning *"This is a rather long message"* or *"This is a very long message."* These features are not cute options, but in fact necessary components of a robust system in actual use.

It is vital to be able to interrupt the output at any time and advance to the next message, repeat the current message, or back up if the caller desires. This requires the system to be listening at all times, and allow itself to be interrupted, as there is not any other means of feedback available. Similarly, while speaking a greeting message, which includes instructions for entering an ID and password, or any such *control* function, the system should also be listening. The first digit entered cuts off the explanatory introduction, allowing an experienced user a fast transaction while a novice can receive extended assistance. Such a feature requires the ability to inform the synthesizer to flush text remaining unspoken in its own internal buffer.

Several aids can improve speech intelligibility. A lexicon of frequently used and mispronounced words, with "corrected" spellings, can be stored either in the host or synthesizer. It is also helpful to provide some means of repeating a poorly understood passage. The Voiced Mail system has a single *repeat* command; with its first invocation, the current sentence is repeated more slowly or, if already in slow mode, spelled out. This allows a convenient balance between speed and intelligibility, which improves with slower speech.

8 CONCLUSIONS

This chapter has espoused a design philosophy for a voice interactive interface, based on intelligence and conversational ability. This interface has a life of its own, and functions as an *intermediary* between speech peripherals and whatever application tasks they may be driving.

Several reasons for this approach to interface development have been identified. The driving force has been the limitations discovered in speech hardware, many of which are inherent in the speech channel itself, and some of which are simply reflections of the current state of a nascent technology.

An underlying theme, however, is that such a style of interaction not only renders this error-prone hardware functional, it also results in a higher level of interface utility. Rather than the interface being a simple path to some database, it can begin to act as the user's *agent,* discussing a request or making assumptions about what the user *really* wants, in light of knowledge about the user as well as the database. At this stage, computers transcend the role of tools, and begin to become assistants.

That this work is still in its early stages is an explanation, if not a justification, of its ad hoc nature. Hence, it has been presented largely in terms of three selected projects, chosen to illustrate various perspectives on the interface problem. On the one hand, these examples may be interpreted as a shotgun approach to device integration. Alternatively, a *style* of human–machine interaction may be gleaned as the theme of such a collection of techniques. The latter is the spirit in which this work has been presented, and hopefully will be the touchstone against which future developments will be judged.

REFERENCES

Allen, J. (1976). Synthesis of speech from unrestricted text. *Transactions of the IEEE, 4,* 433–442.

Anderson, M. D., Pierrehumbert, J. B., & Liberman, M. Y. (1984). Synthesis by rule of English intonation patterns. *International Conference on Acoustics, Speech and Signal Processing.* New York: IEEE.

Arons, B. (1984). The audio-graphical interface to a personal integrated telecommunications system. Master's thesis, June, Massachusetts Institute of Technology.

Beek, P., Neuberg, E. P., & Hodge, D. C. (1977). An assessment of the technology of automatic speech recognition for military applications. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 25*(4), 310–322.

Bobrow, D. (1967). *Natural language input for a computer problem solving system.* Cambridge, MA: M.I.T. Press.

Bolt, R. A. (1980). Voice and gesture at the graphics interface. *Proceedings, SIGGRAPH '80,* 262–270.

Brown, G., Currie, L., & Kenwothy, J. (1980). *Questions of intonation.* London: Croon Helm.

Brown, G. & Yule, G. (1983). *Discourse analysis.* Cambridge, England: Cambridge University Press.

Chen, F. R., & Zue, V. W. (1984). Application of allophonic and lexical constraints in continuous digit recognition. *International Conference on Acoustics, Speech and Signal Processing.* IEEE.

Chomsky, N. (1957). *Syntactic Structures.* The Hague: Mouton.

Chomsky, N. (1965). *Aspects of the Theory of Syntax.* Cambridge, MA: M.I.T. Press.

Dautrich, B. A., Rabiner, L. R., & Martin, T. B. (1983). The Effects of Selected Signal Processing Techniques on the Performance of a Filter-Bank-Based Isolated Word Recognizer. *The Bell System Technical Journal, 62*(5), 1311–1336, May–June, 1983.

Dixon, N. R., & Silverman, H. F. (1981). What are the significant variables in dynamic programming? *International Conference on Acoustics, Speech and Signal Processing,* 728–731. IEEE.

Flanagan, J. L., Schroder, M. R., Atal, B. S., Crochiere, R. E., Jayant, N. S., & Tribolet, J. M. (1979). Speech coding. *IEEE Transactions on Communications 27*(4) April, 710–737.

Green, B., Wolf, A. K., Chomsky, C., & Laughery, K. (1963). *BASEBALL: An automatic question answerer.* New York: McGraw-Hill.

House, A. S., Williams, C. E., Hecker, M. H. L., & Kryter, K. D. (1965). Articulation testing methods: consonantal differentiation with a closed response set. *Journal of the Acoustical Society of America, 37,* 158–166.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics Speech and Signal Processing, 23,* 67–72.

Kato, Y., (1980). Words into action: a commercial system. *IEEE Spectrum, 29,* June.

Klatt, D. H., & Stevens, K. N. (1973). On the automatic recognition of continuous speech: Implications of a spectrogram-reading experiment. *IEEE Transaction on Audio and Electroacoustics, 21,* 210–217.

Klatt, D. H. (1977). The ARPA speech understanding project. *Journal of the Acoustical Society of America, 62*(6), 1345–1366.

Klatt, D. H. (1980). Software for a cascade/parallel formant synthesizer. *Journal of the Acoustical Society of America 67*(3), 971–995.

Lea, W. A. (1980). *The value of recognition systems.* Englewood Cliffs, NJ: Prentice-Hall.

Lea, W. A., Medress, M., & Skinner, T. (1975). A prosodically-guided speech understanding strategy. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 23,* 30–38.

Levinson, S. E. (1978). The effects of syntax analysis on word recognition accuracy. *The Bell System Technical Journal, 57*(5), May–June, 1627–1644.

Levinson, S. E., Rosenberg, A. E., & Flanagan, J. L. (1978). Evaluation of a word recognition system using syntax analysis. *The Bell System Technical Journal, 57*(5), May–June, 1619–1626.

Levinson, S. E., & Shipley, K. L. (1980). A conversational-mode airline information and reservation system using speech input and output. *The Bell System Technical Journal, 59*(1), January, 119–137.

Lindsay, R. (1973). In defense of ad hoc systems. In R. C. Schank & K. M. Colby, *Computer Models of Thought and Language.* San Francisco, CA: Freeman.

Luce, P. A., Feustel, T. C., & Pisoni, D. B. (1983). Capacity demands short-term memory for synthetic and natural word lists. *Human Factors, 25*(1), 17–32.

Markel, J. D., & Gray, A. H. (1979). *Linear prediction of speech.* New York: Springer-Verlag.

Martin, T. B. (1976). Practical applications of voice input to machines. *Proceedings of the IEEE, 64,* April, 487–501.

Maxemchuk, N. F. (1980). An experimental speech storage and editing facility. *The Bell System Technical Journal, 59*(8), 1383–1395.

McPeters, D. L., & Tharp, A. L. (1984). The influence of rule-generated stress on computer synthesized speech. *International Journal of Man-Machine Studies, 20*(2), 215–226.

Miller, G. A., Heise, G. A., & Lichten, W. (1951). The intelligibility of speech as a function of the context of the test materials. *Journal of Experimental Psychology, 41,* 329–335.

Nakagawa, S. (1983). A connected spoken word recognition method by O(n) dynamic programming pattern matching algorithm. *International Conference on Acoustics, Speech and Signal Processing,* pages 296–299. New York: IEEE.

Ochsman, R. B., & Chapanis, A. (1974). The effects of 10 communications modes in the behavior of teams during cooperative problem solving. *International Journal of Man-Machine Studies 6,* 579–619.

Oshika, B., Zue, V. W., Weeks, R. V., Nue, H., & Aurbach, J. (1975). The role of phonological rules in speech understanding research. *IEEE Transactions on Acoustics Speech, and Signal Processing, 23*(1) 104–112.

Pallet, D. S. (Ed.), (1982). *Proceedings of the Workshop of Standardization for Speech I/O Technology.* National Bureau of Standards, 1982.

Pathe, P. (1983). A virtual vocabulary speech recognizer. Master's thesis, Massachusetts Institute of Technology, June.

Pierrehumbert, J. (1981). Synthesizing intonation. *Journal of the Acoustical Society of America, 70*(4), 985–995.

Rabb, F. H., Blood, E. B., Steiner, T. O., & Jones, H. R. (1979). Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace and Electronic Systems, 15*(5), 709–718.

Rabiner, L. R., & Schafer, R. W. (1978). *Digital processing of speech signals.* Englewood Cliffs, NJ: Prentice-Hall.

Rabiner, L. G., & Wilpon, J. G. (1980). A simplified, robust training procedure for speaker trained, isolated word recognition systems. *Journal of the Acoustical Society of America, 68*(5) 1271–1276.

Rabiner, L. R., & Levinson, S. E. (1981). Isolated and connected word recognition—theory and selected applications. *IEEE Transactions on Communications, 29*(5), 621–659.

Rabiner, L. R., Rosenberg, A. E., Wilpon, J. G., & Keilin, W. J. (1982). Isolated word recognition for large vocabularies. *The Bell System Technical Journal, 6*(10), 2989–3005.

Reddy, R. (1976). Speech recognition by machine: a review. *Proceedings of the IEEE, 64,* 501–531.

Rosenberg, A. E. (1983). Probabilistic model for the performance of speech recognition systems. *International Conference on Acoustics, Speech and Signal Processing,* pages 1057–1060. IEEE.

Sakoe, A. (1979). Two level DP-matching dynamic programming based on pattern matching algorithms for connected word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 27*(6), 588–595.

Schmandt, C., & Hulteen, E. A., (1982). The intelligent voice interacive interface. *Proceedings, Human Factors in Computer Systems,* pages 363–366. ACM National Bureau of Standards.

Schmandt, C., & Bender, W. (1983). A programmable virtual vocabulary speech recognition peripheral. *Proceedings, Voice Data Entry Systems Conference.* Chicago American Voice Input Output Society, Palo Alto, CA 1983.

Schmandt, C., & Arons, B. (1984a). A conversational telephone messaging system. *IEEE Transactions on Consumer Electronics,* CE-30(3), xxi–xxv.

Schmandt, C., & Arons, B. (1984b). Phone Slave: A Graphical Telecommunication Interface. *Digest of Technical Papers.* SID International Symposium, San Francisco.

Secrest, B., Arjmand, M, & Ni, M. (1982). Speech analysis and synthesis become practical on a μC chip. *Electronic Design,* May 27, 129–136.

Slowiaczek, L. M., & Pisoni, D. B. (1982). Effects of practice on speech classification of natural and synthetic speech. *Journal of the Acoustical Society of America,* 71, Suppl. 1, 596–597.

Winograd, T. (1983). *Language as a Cognititive Process.* Reading, MA: Addison-Wesley.

Hartson,
Editor

ADVANCES IN
HUMAN–COMPUTER INTERACTION, VOLUME 1

ADVANCES IN HUMAN–COMPUTER INTERACTION

VOLUME 1

H. Rex Hartson

editor

Advances in Human–Computer Interaction

Volume 1

H. REX HARTSON, Editor

Virginia Polytechnic Institute and State University

ABLEX PUBLISHING CORPORATION
NORWOOD, NEW JERSEY