THE EFFECTIVENESS OF SPEECH RECOGNITION AS A USER INTERFACE FOR COMPUTER-BASED TRAINING

DISSERTATION

Presented to the Graduate Council of the

University of North Texas in Partial

Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Wayne E. Creech, B.G.S., M.A.

Denton, Texas

August, 1995

THE EFFECTIVENESS OF SPEECH RECOGNITION AS A USER INTERFACE FOR COMPUTER-BASED TRAINING

DISSERTATION

Presented to the Graduate Council of the

University of North Texas in Partial

Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Wayne E. Creech, B.G.S., M.A.

Denton, Texas

August, 1995

Creech, Wayne, The effectiveness of speech recognition as a user interface for computer-based training. Doctor of Philosophy (Applied Technology, Training, Development), August, 1995, 140pp., 7 tables, 12 illustrations, references, 55 titles.

Some researchers are saying that natural language is probably one of the most promising interfaces for use in the long term for simplicity of learning. If this is true, then it follows that speech recognition would be ideal as the interface for computer-based training (CBT). While many speech recognition applications are being used as a means for a computer interface, these are usually confined to controlling the computer or causing the computer to control other devices.

The user input or interface has been the recipient of a strong effort to improve the quality of the communication between man and machine and is proposed to be a dominant factor in determining user productivity, performance, and satisfaction. However, other researchers note that full natural interfaces with computers are still a long way from being the state-of-the art with technology. The focus of this study was to determine if the technology of speech recognition is an effective interface for an academic lesson presented via CBT. How does one determine if learning has been affected and how is this measured? Previous research has attempted quantify a learning effect when using a variety of interfaces. This dissertation summarizes previous studies using other interfaces and those using speech recognition. It attempted to apply a

framework used to measure learning effectiveness in some of these studies to quantify the measurement of learning when speech recognition is used as the sole interface. The focus of the study was on cognitive processing which affects short-term memory and in-turn, the effect on original learning (OL).

The methods and procedures applied in an experimental study were presented.

ACKNOWLEDGEMENTS

My deepest gratitude to my wife, Marlis, for her love, sustained support, and encouragement that helped me through seven long years. Her positive attitude and willingness always to go the extra mile made it possible to achieve our goal. I am also grateful to my children, Rick, Laura, and Wayne, Jr., for their understanding of the "quality time" we missed, and their responsiveness to any and all requests for help when needed.

My special thanks to Brad Eck, software engineer, who demonstrated an outstanding ability to understand the intricacies of speech recognition and apply it to an instructional application. His cheerful enthusiasm and technical expertise were indispensable in converting one authoring system to another while programming the speech recognition application. He volunteered many hours of his personal time to assist me.

I would also like to thank my advising professor, Dr. Jerry Wircenski, for his professional and thoughtful guidance throughout my program, and Doctors Roger Ditzenberger, Randall Schumacker, Donald Staples, and Jon Young for their constructive critiques and contributions to this dissertation.

TABLE OF CONTENTS

TABLE OF CONTENTS--Continued

TABLE OF CONTENTS--Continued

LIST OF TABLES

LIST OF ILLUSTRATIONS

Figure Page

CHAPTER I

INTRODUCTION

This chapter reviews the development of speech recognition technology and its potential as a natural interface for the computer. Several interfaces are explored and the equipment that makes up a generic speech recognition system is discussed. The statement of the problem, purpose of the study, and research questions are outlined. The hypothesis is stated and the potential significance of the study is discussed, as well as the delimitations and limitations. The chapter concludes with a list of terms defined.

Background of the Study

Instructional software has been developed in the United States since the late 1950s (Karrer, 1988) and among others, has evolved under such names as Computer-based Instruction (CBI), Computer-assisted Instruction (CAI), Computer-assisted Learning (CAL), Computer-based Training (CBT), and Computer-managed Learning (CML). More recently the names have acknowledged the concept of artificial intelligence: Intelligent Computer-assisted Instruction (ICAI) and Intelligent Tutoring System (ITS) (Burns, Parlett, & Redfield, 1991). This study is concerned with a government training environment that commonly refers to software instruction as CBT, and this term will be used throughout this paper.

The use of CBT has grown more popular since the development of the microchip that made personal computers (PCs) affordable (Thyfault,1994) and the creation of authoring systems that allowed nonprogrammers to develop curricula (Tessmer, Jonassen, & Caverly, 1989). Various methods have been devised to allow learners to respond to CBT instruction; these include typing a response from a keyboard, touching the computer monitor's screen, and using a mouse or trackball with a cursor to select objects on the screen. These types of input devices are distracting and, in the case of the keyboard, place disproportionate demands on the learner's motor system (Jonassen & Hannum, 1987). This in turn affects the level of information processing, which impacts the degree of learning during CBT instruction.

The user input, or interface, has been the focus of a strong effort to improve the quality of the communication between man and machine. According to Burger and Desoi (1991), the interface quality is a dominant factor in determining user productivity, performance and satisfaction. Nine classifications of interface techniques were outlined by Baecker and Buxton (1987) and are reiterated in Table 1(Murphy, 1992). Users of one of the newer interface methodologies called direct-manipulation (Item 8, Table 1) reported positively on feelings of success and satisfaction (Burger & Desoi, 1991). However, natural language (Item 3, Table 1) is cited as probably the most promising interface "for use in the long term for simplicity of learning" (Murphy, 1992, p. 31). Although Murphy acknowledged that limited restricted language interfaces have made some progress, he stated that "it would appear that we are still a long way from being able to provide effective full natural language interfaces with computers" (p. 31).

Fortunately, natural language interface, known as speech recognition may have been developing more rapidly than was appreciated by Murphy (1992). This is due to several factors: the 1,000% increase in microprocessor power, a price drop in PCs, and new developments in voice generation devices (Thyfault, 1994). The new devices offer greater accuracy, speed, and reliability (Lee, 1989). Table 2 depicts the progress of speech recognition expressed in word error rate from the 1970s to the present (Rudnicky, Hauptmann, & Lee, 1994). Speech recognition is a "process of automatically identifying spoken words (Foster & Schalk, 1993, p. 8.). A more technical description is that it is a technology that converts sound, words or phrases spoken by humans into electrical signals, and these signals are transformed into meaningful coding patterns (Adams, 1990). Speech recognition systems are usually divided into the categories of speaker dependent and speaker independent (Foster & Schalk, 1993). Speaker dependent systems must be trained by the speaker and include choosing a vocabulary, training the recognizer to recognize the characteristics of the speaker, forming a reference by repeating each vocabulary word several times, and training in the environment in which the recognizer will be used (Foster & Schalk, 1993).

Table 1

Interface Classifications

NAME	DESCRIPTION
Command line	The user types instructions to the computer in a formally defined command language.
Program language	The command language allows its own extensions through the definition of procedures.
Natural language	The user's natural language is a defined subset of a natural language such as English.
Menu systems	The user issues commands by selecting sequential choices from displayed alternatives.
Filling	The user issues commands by filling in fields that are displayed on the screen.
Iconic	User commands and system feedback are expressed in pictograms instead of words.
Windows	The user's screen is divided into overlapping rectangular areas, each with a specific function.
Direct manipulation	The user manipulates, through buttonpushes and movements of a pointing device such as a mouse, a graphic representation of the underlying data.
Graphical	The user defines and modifies sketches, diagrams, renderings, and other two-or three-dimensional images and pictures.

Note. Adapted from Murphy (1992).

Table 2

Progress in Speech Recognition

Task	Error rate Late - 1970s	Error rate Late -1980s	Error rate Early -1990s
Speaker independent, independent word recognition	30%	10%	4%
Speaker independent, continuous speech recognition digits	10%	6%	0.4%
Speaker dependent, continuous speech recognition query: 1000 word - perplexity 4	2%	0.1%	
Speaker idependent, continuous speech recognition query: 1000 word - perplexity 60	-	60%	3%
Speaker dependent independent word recognition dictation: 5,000 word	-	10%	2%
Speaker independent continuous speech recognition dictation: 5,000 word	-	-	5%
Speaker independent continuous speech recognition dictation: 20,000 word	-	-	13%

Note. Adapted from Rudnicky, Hauptmann, & Lee, (1994).

Although the independent system does not have to be trained to adapt to the speaker's voice, the recognizer is trained from a large speech database prior to being used. Some systems are speaker adaptive in that they initially function as speaker independent but adjust to the speech of the individual. This causes a corresponding increase in accuracy (Rudnicky et al.,1994). Both dependent and independent systems may use discrete or continuous word recognition (Foster & Schalk, 1993). Discrete recognition uses a pause of 250 milliseconds to separate each word, whereas continuous recognition has fewer than 50 milliseconds of silence between words in a series (Foster & Schalk, 1993).

In both dependent and independent systems, a user speaks into a microphone, normally mounted on a headset. The headset is plugged into a host computer and interfaces with a speech recognition board, which contains a digital signal processor that converts the voice signal (analog) into digital signals.

The digital signals are analyzed and converted to text or decoded as a command. The processor and analyzer work from statistical and mathematical algorithms that recognize speech blocks, group them into words, and match them against models stored in the program. An illustration of a generic speech system is shown in Figure 1 (Creech, 1994).

Speech recognition is already in place for a variety of users; receiving operators for COMPAQ unpack items, track returns, credit customers, and order parts through speech recognition data entries: and mechanics at some Air Force bases state commands to the computer to "enter diameter" (Thyfault, 1994).

Doctors create digital medical records, lawyers dictate letters, stock traders tell a workstation to buy or sell, and postal clerks decipher illegible zip codes. Mobility-impaired persons are also finding that speech recognition provides a means of overcoming their disability. Among these enabling areas are (a) computer interaction for environment control, (b) translation of utterances from persons without sufficient control of articulatory muscles into intelligible speech and (c) muscle stimulation and robotic arm control.

Chamberlain (1993) conducted 18 case studies with users of four high-vocabulary, speaker-dependent, isolated word/discrete utterance, computerized voice-recognition systems. The studies included interviews with personnel in rehabilitation centers and hospitals, as well as industry and other professions. Of particular note were six repetitive stress injury (RSI) studies. This type of injury is one of the leading causes of acquired disability in the United States and Europe (Chamberlain, 1993). According to the Bureau of Labor Statistics, ergonomic disorders were the fastest growing work-related illnesses in 1992 and accounted for over 50% of those reported to the Occupational Safety and Health Administration (OSHA). The subjects in the six RSI cases stated that voice recognition was allowing them to return to work. Another investigation, from the United Kingdom (Yunus, 1992), was focused on the design of computer-based tools for training and rehabilitation of the handicapped and a specific case study of the Nuffield Dyspraxia Speech Training Program (NDSTP).

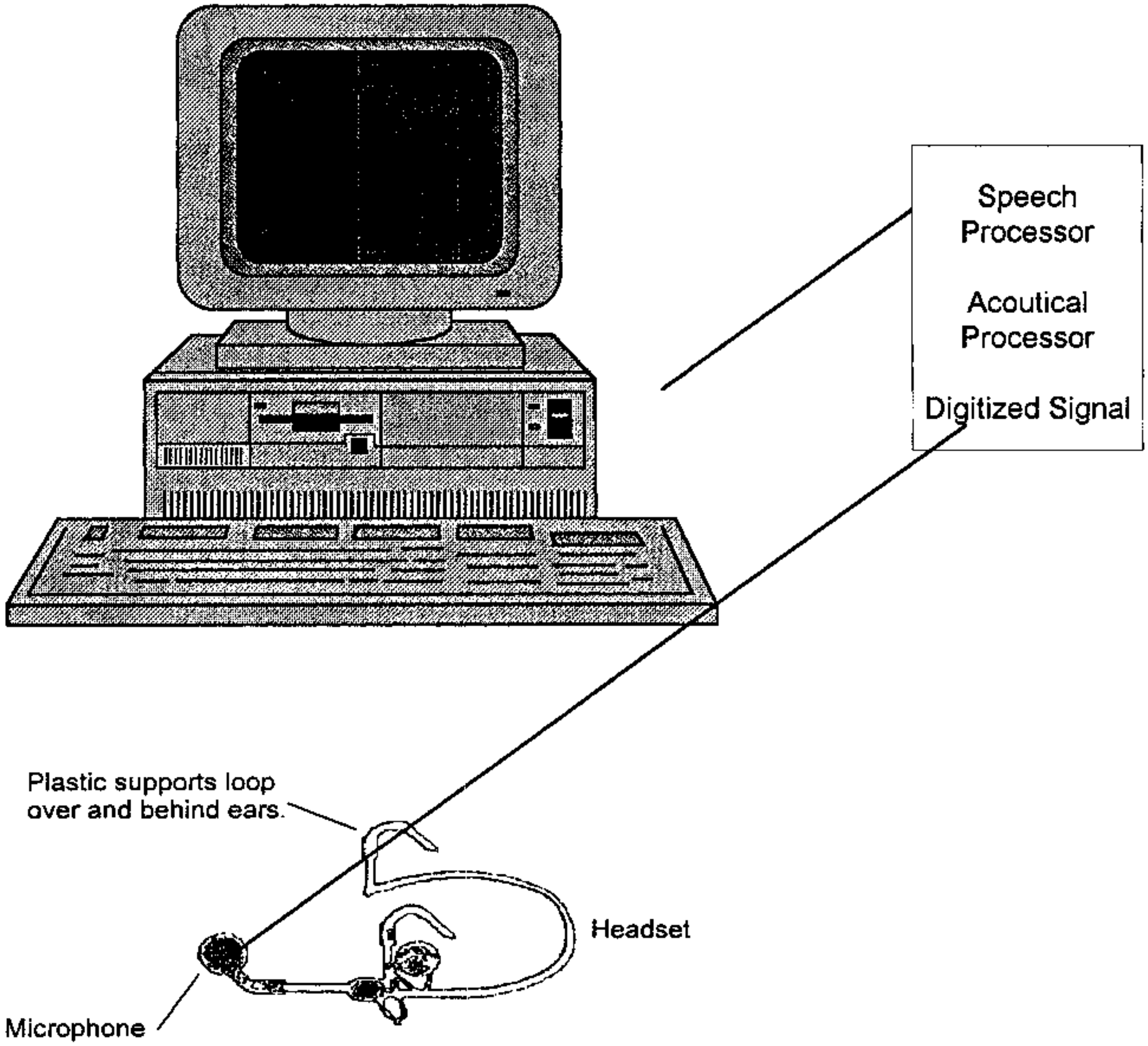

Figure 1: Generic Speech System (Creech, 1994)

The study noted that a computer-based therapy program was enhanced by a facility to use automatic performance and feedback. Developments in speech recognition were studied and tests done to select an appropriate algorithm for integration in the NDSTP environment.

One of the largest applications of voice recognition has come from telephone companies whose systems recognize miscellaneous commands such as "call home" and "call the office" (Filipczak, 1993). These applications have proven useful as time- and labor-saving tools, but are not cited in any training relationship; however, these uses have prompted several studies that relate to the effectiveness of speech recognition systems. Included in these are the effects of recognition accuracy and vocabulary size on task performance and user acceptance (Casali, Williges, & Dryden, 1990), decline in accuracy as a function of time on task (Frankish, Jones, & Hapeshi, 1992), and feedback strategies for error correction (Ainsworth & Pratt, 1992). In a direct training application, the Federal Aviation Administration (FAA) has been training air traffic controllers in a simulated control tower that uses speech recognition as the interface (Klass, 1991). A simulation for battlefield management was demonstrated by Hughes Training, Inc., which also used a speech recognition interface. In both training scenarios, the speaker was the only "live" player. The speech recognition system reacted to voice inputs, responded through speech-generated or recorded replies, and caused movement of computer-generated aircraft or tanks on computer screens or a silicon graphics interface (Creech, 1994).

Justification of the Study

For the researcher, participating as a designer of the battlefield scenario through use of speech recognition created an ardent interest in this new technology. Although the battlefield scenario demonstration revealed that speech recognition could be used in a training environment, it was totally embedded so that the effect was the same as three "live" persons responding to the battlefield commander. This was more cost effective than requiring the three persons to role-play, but any other added value was not apparent. To the researcher, the validity of this new technology in general training applications was not yet proven, and the rush to use it was parallel to some of the early efforts to use CBT. In his meta-analysis of computer-based instructional feedback, J. Harris (1994) cited several studies that criticized the development of instructional software as "educationally invalid." The studies did not use research in their design and did not use what was already known such as the application of the principles of learning. Using an established CBT lesson based on validated learning principles with a new technology to determine whether or not the combination would add value to a training application provided the impetus for this study.

Adams (1990) noted that the manual interface such as a keyboard has become an obstacle for humans in handling data entry and control functions, and Creech (1994) noted that speech recognition offers the potential for a more user-friendly interface. Apart from the convenience of data entry and control, does this new technology offer any way to positively affect learning in a training application? If so, for which kind of application or instructional strategy? There is a need for several studies during the early evolvement of the technology to help establish which instructional strategies, if any, would have a positive effect on learning if speech recognition was a part of those strategies. This study is anl attempt to provide part of this information.

Statement of the Problem

Often educators are called upon to justify the cost of equipment and hardware and should be able to provide more than intuitive evidence that technology serves a purpose in education (Poirot & Knezek, 1992). This question could be asked with each new technological advance, but may be omitted in the race to use the "state of the art." Instructional designers of corporate training programs may find themselves caught between the requirement to use the latest technology and questions about the instructional purpose it may serve. It would seem that using natural language as an interface for a computer would be the most promising interface, as noted by Murphy (1992). From the improvements cited in the previous discussion (Lee, 1989), (Thyfault, 1994), and (Rudnicky et al., 1994), it was evident that extensive progress has been made. However, after observing several demonstrations, the researcher found that the use of speech as an interface for applications such as word processing was not as easy as it appeared. For example, during a demonstration of the use of speech recognition for the interface with a word processing application using a discrete word, dependent-speech recognizer, the system locked up after several editing functions were attempted. The speaker told the system to move the cursor "three lines down" and to "move to second word." After several of these kinds of repetitions, the recognizer seemed to become confused and had to be restarted. The same was true with another system using an independent speech, continuous word recognizer; the system recognized most of the dictation given by the speaker, but did not respond well to editing commands. This system also had to be restarted several times.

According to Jonassen and Hannum (1987), an overt response requires different levels of cognitive processing, ranging from no cognitive transformation to complex processing. Given that speech is an overt response, it seems possible that more complex cognitive processing could occur with speech recognition as the interface for CBT. However, Karl, Pettey, and Shneiderman (1993), found that short-term memory may be inhibited when using voice commands for word processing applications. (This study is discussed in more detail in chapter 2.) The problem then becomes one of determining whether more complex processing occurs with speech recognition as the interface for CBT than with traditional manual interfaces such as the mouse.

Specific Purpose

The purpose of the study is to provide information that can be used as part of a continuing effort to determine whether or not speech recognition technology is beneficial when used with CBT applications. Beneficial includes instructional design decisions that impact cost and development time, learning at a faster rate, effect on memory, and learning achievement. This study is an attempt to provide a basis for further comparisons as technologies for CBT and speech recognition evolve.

Statement of Research Questions

The examination of whether speech recognition technology is beneficial in a CBT application concerns the following questions:

1. Is speech recognition a computer interface that will affect cognitive processing, thus affecting original learning?

2. Will using speech recognition as a computer interface affect the time required to complete a CBT lesson compared to using a mouse as a computer interface?

Statistical Hypothesis

The answers to the above research questions are provided through the following hypothesis:

There will be no statistically significant difference in original learning effect when speech recognition is used as a computer interface for a CBT lesson compared to using a mouse as the computer interface. Independent variables are the types of interface. Dependent variables include achievement scores, progress check scores, and time-on-task.

Significance of the Study

This research study will contribute to the body of knowledge applied to learning through CBT as it relates to the target population. Application of the results by CBT practitioners may include, but are not limited to, the following: (a) use of speech recognition as a user interface for CBT instruction, (b) short-term memory affected by using speech recognition, (c) time-on-task, (d) design considerations, and (e) cost considerations.

Americans with disabilities make up a special population that may derive exceptional benefit from this application . Computer programming, telephone dialing, environmental control, and wheelchair maneuvering are some of the tasks accomplished through speech input (Casali et al., 1990). This same group could also benefit from CBT lessons in which the learner interfaces with the "tutor" through speech. Studies by Mullally, Kinkaid, and Kishek (1993) and Kincaid, Mullally, Meyer, and Bramble (1993) indicated that there was an excellent potential for language learning using speech recognition as the "evaluator." A similar possibility is being explored for children with learning disabilities by an institution in Waco, Texas, known as Essential Learning Systems (ELS) (Creech, 1994)

Delimitations

In this study, no attempt was made to determine which speech recognizer system best represents the state of the art. After three demonstrations, a system was chosen that achieved approximately 95% correct recognition, or a 5% rejection rate, and was reasonably priced. In addition, time, money, available staff, and computer terminals all prevented the development of a block of lessons that would have constituted a mini-course of instruction with additional instructional strategies and tests, which would have provided more data. These same factors prevented the testing of a Retained Learning Variable Set, which could provide further data on the effect of retained learning. In addition, while one type of user interface may be preferred over the other, there was no intention to test user satisfaction or to compare likability scores within the scope of this study.

Limitations

The speech recognition card in the system used for testing required that a mouse button be depressed to activate the recognizer. Other speech systems have voice-activated recognizers that provide a completely hands-free application. Only one subject could be tested at a time. The process required 7 school days whereas 2 hours would have been sufficient if the computer equipment and copies of the program had been available.

This study also limits the application to two of the four categories of Murphy's framework for Initial Learning Domain Variables (See Table 3 in chapter 2): the Learning Strategy Variable Set (LS Variables) and the Original Learning Variable Set (OL Variables). The Individual Differences Variable Set (ID Variables) and Task Characteristics Descriptor (TC Descriptor) were not used. An extensive effort would be required to determine the effect, if any, of the variables of mixed attitudes, experience, age, sex, and learning styles. The TC Descriptor was not used because the task was to, "learn the lesson content," and did not fit Murphy's quantifiable parameters. The Retained Learning Variable Set (RL Variables) was not used because of time and logistic considerations.

Definition of Terms

<u>Active Vocabulary:</u> a list of the vocabulary items stored in a speech recognizer that can be instantly recognized at any point in the application.

<u>Analog Signal:</u> the sound created by the user's voice when speaking to the speech recognizer (<u>Phonetic Engine 400</u>, 1993).

<u>Bitmap:</u> a procedure that allows areas of a CBT screen that contain large, complex graphic displays to be converted into one object (<u>Paint</u>, 1991).

Computer-Based Training (CBT): an individualized instruction concept that uses a computer as the teaching medium. The term is used interchangeably with Computer-Aided Instruction (CAI), Computer-Based Instruction (CBI), and Computer-Aided Learning (CAL).

Continuous Speech: combinations of vocabulary items, sentences, or numbers spoken in a natural, conversational style. Continuous recognition has fewer than 50 milliseconds of silence between words in a series (Foster & Schalk, 1993).

Dependent Speech: a speech recognizer that must be trained to the user's voice (Foster & Schalk, 1993).

Digitized Signal: a speech signal that has been converted from analog to digits to allow analysis and processing by a speech processor in a speech recognizer (Phonetic Engine 400, 1993).

Discrete Speech: recognition that uses a pause of 250 milliseconds to separate each word (Foster & Schalk, 1993).

Elaboration: "providing additional information at the time of processing" (Murphy, 1992, p. 19).

Feedback: the informative or evaluative information provided in response to a student's action (Thorkildsen & Reid, 1989).

Generation: creation of words or rhymes to help improve memory (Murphy, 1992).

Grammar: a defined sequence of spoken words that structures the application to provide a smooth input flow and help achieve faster response and higher accuracy (Phonetic Engine 400, 1993). Grammar is also referred to as syntax.

Independent Speech Recognizer: a speech recognizer that does not have to be trained to each user's voice (Foster & Schalk, 1993).

Instructional Design: field or profession dealing with identifiable systematic stages of the teaching/learning process. Procedures of instructional design embrace lesson planning, module development, media selection, presentation technique, and system evaluation, within which the goals and outcomes are prescribed (Gagne & Briggs, 1979).

Instructional Programs: computer software in which the content is based on curriculum and is often accompanied by a manual and/or workbook as a package in the courseware. The programs range from basic exercises to complex, interactive simulations and are often developed in conjunction with a variety of multimedia devices such videodisc, videotape, and compact disc.

Modality: the mode of input to a computer program such as a mouse or keyboard.

Multimodality: multiple modes of input to a computer program such as mouse, keyboard, and voice recognition.

Perplexity: an index of speech recognition that corresponds to the number of word alternatives that the system must consider at any one point in the decoding process (Rudnicky et al., 1994).

Phoneme: a member of the set of the smallest units of speech that serve to distinguish one utterance from another in a language or dialect.

Recognition Accuracy: percentage of time in which the speech recognizer correctly classifies an utterance (Foster and Schalk, 1993).

Rejection Error: an errr made "when the recognizer does not classify a spoken word, but rejects it" (Foster and Schalk, 1993, p. 15)

Rejection Error Rate: the rate, usually expressed in a percentage, at which rejection errors occur (Foster & Schalk, 1993).

Simulation: a highly automated, dynamic learning system often used to teach operational tasks through a trial and error process. The system depicts realistic entities of environments that provide a feeling and sense of a real situation. Participants deduce, make analytical judgments, and participate in decision-making. (Schmalz & Sipl, 1972).

Speech Recognition: a "process of automatically identifying spoken words" (Foster & Schalk, 1993, p. 8) or a technology that converts sound, words or phrases spoken by humans into electrical signals that have meaningful coding patterns (Adams, 1990).

<u>Speech Recognizer:</u> a computer program, usually installed on a separate board, that allows users to input data into an instrument, controller PC, mini, or mainframe computer using speech rather than a keyboard or mouse. The recognizer digitizes and analyzes speech input, then converts it to text (<u>Phonetic Engine 400</u>, 1993).

<u>Spurious Response Error:</u> an error made "when the recognizer classifies a sound or invalid word as a valid word" (Foster and Schalk, 1993, p. 15).

<u>Substitution Error:</u> an error made "when a recognizer substitutes an incorrect word for the spoken word" (Foster and Schalk, 1993, p. 15).

<u>Tutorials:</u> instructional programs that provide electronic learning in a dialogue format that introduces new information to be learned. Segments of information are displayed on the monitor for students to read and learn by participation in a repetitive question and answer process (Coburn, 1982). Tutorials present a page of text (which may include graphics), ask a question, and provide feedback on the correctness of the answer. The program may provide remedial material for an incorrectly answered question (Merrill, 1987).

<u>Utterance:</u> "a group of sounds that make a word, phrase, or sentence that a speech recognizer is programmed or trained to detect "(Creitz, 1991, p. 79).

CHAPTER II

REVIEW OF RELATED LITERATURE

This chapter notes one of the basic needs for an improved interface and reviews the research efforts in speech recognition during the 1980s, the attempts at language applications during the 1990s, and the shift to multimodality interface research in both the late 1980s and the 1990s. Concepts of short-term and long-term memory are reviewed in relation to features designed to enhance memory retention and retrieval. The framework developed to use the memory enhancement features and the methods to quantify learning effectiveness are then explored.

Both the development of the computer and the concept of voice recognition date back to before to World War II. The advancement of computer technology has been steady since the early 1940s and has grown rapidly since the late 1970s (Thyfault, 1994). Conversely, the progress of voice recognition was not well received by the scientific and technical communities until the 1980s. A major reason has been the degree of accuracy with a voice input compared to that of a keyboard or manual input (Lee, 1989). Ironically, it is the manual interface that has become an obstacle for humans in handling the data entry and system control functions. The need for an improved interface between the information systems and their users is a prime factor in current technology research efforts (Adams, 1990). Voice or speech recognition offers a potential for a more user-friendly interface and is the object of renewed interest in both military and civilian communities (Creech, 1994).

Major Research in the 1980s

Researchers in the early 1980s focused primarily on determining differences in performance when speech replaced the usual keyboard input in restricted applications (Karl et al., 1993). At that point, speaker-independent recognition was still in the stages of research and development, and recognition performance was considered a tradeoff with vocabulary size. It was not clear that speech recognition would be useful for a wide range of tasks. Karl et al. cite several studies; some of which had mixed results. One study compared speed and accuracy of speech versus typed input of command in a simulated command and control environment (Poock, 1982). The voice inputs proved to be faster and had fewer errors. A study that compared baggage-sorting tasks and again proved voice to be more error-free than keying in the destinations (Nye, 1982). Another experiment evaluated the performance of speech versus keyboard for computer program entry and editing (Leggett & Williams, 1984). The keyboard users completed 25% more of the input and editing tasks than when speech was used. Still another comparison showed that speech input was more accurate but slower in the entering of information for circuit layouts whereas another showed that voice input was faster for parcel sorting but was less accurate (Visick, Johnson, & Long, 1984).

Visick et al. (1984) stated that any superiority of a voice input to a keyboard would depend to a great extent on the type of task being performed, and they suggested that speech recognition should be applied only if there was a direct user benefit. Examples of this benefit were the freeing of the hand or eyes or the reduction of mental encoding in such activities as warehouse tasks, missile launching, and robot control. This point was further supported by Creech (1994), who noted that speech recognition remains a tool to accomplish the task and not a focal point of training. Greene, Gould, Boies, Rasamny, and Meluson (1992) also noted that choosing the interaction methods that are most appropriate for the intended users, for the application content, is a part of designing user interfaces.

Language Applications

A natural application content for speech recognition would seem to be language training or exercises to overcome speech disorders. Mullally et al. (1993) developed computer-assisted language training software that was designed for Marines to use in Somalia. This learning tool was needed for rapid language learning for a combat environment. Students spoke into a microphone, their voices were digitized, and a computer compared the speaker's pronunciation to that of a native speaker. This method was considered by the authors to be the best technique until speech recognition was available. Mullally et al., conducted a similar study with fifth and sixth graders, using a computerized practice with a voice recorder. These two studies implied that positive results would be achieved if the voice recorder/comparer function was done with a speech recognizer.

These data indicate a strong potential for speech recognition with computer-assisted language learning. This type of application was developed for the Royal Saudi Naval Forces (RSNF) by Southwest Research Institute (Golas, Fredrickson, & Negrie, 1994). The RSNF uses the Defense Language Institute (DLI) American Language Course (ALC) to instruct its personnel in the English language. The DLI course consists of classroom instruction and audio lab exercises. The course is less effective when taught outside the U.S. because the students cannot apply it in context to the Saudi Arabian environment. According to Golas et al., the DLI materials contained few illustrations and did not provide the realism required for meaningful learning. To provide a medium for the RSNF students to use English in the same context in Saudi Arabia as in the United States, a CBT supplement was developed to support one of the segments (Book 10) of the DLI course. Speech recognition was included as one of the instructional strategies in two methods: (a) phonemic stress drills on vocabulary words and phrases; and (b) selection of correct answers on exercises. The speech recognition technology was used for the English language CBT based on the following guidelines:

1. The vocabulary was limited to 300 utterances or fewer.

2. No more than seven utterances could be compared at any one time.

3. The speaker system was speaker-independent and was trained to recognize speakers with Saudi accents.

4. Users were provided with escape pathways in the event of recognition failure.

5. Only three attempts at the same word were allowed before the system provided an escape.

6. The speech recognition could not be allowed to interfere with or degrade student learning.

Data collection was made for each student input, with a percentage score tabulated for each segment of the program and a total score for the entire program. A posttest was given on the learning objectives of the supplement. Although the CBT supplement was evaluated for instructional effectiveness at Lackland Air Force Base in San Antonio and in Saudi Arabia, there were no reports of statistical significance. At the time of publication, the researcher reported that the supplement was being formally evaluated in Saudi Arabia to determine its effectiveness in the following areas: (a) learning the American language, (b) training time, (c) motivation, and (d) instructional design strategies.

The method for determining these factors was not reported, but Golas et al. (1994) noted that previous research indicated that CBT and videotapes were effective media for English instruction due to their visual presentations.

In another speech recognition application, Scott Instruments developed a Voice-Based Learning System (VBLS) with a dependent speech recognizer that incorporated a "non-technical authoring system " to meet generic computer-aided instructional requirements (Creitz, 1991, p. 88). This system included tutoring through a display of questions and answers. The student spoke the answers, which also allowed the system to adapt to the student's voice. The developers noted that they were trying to achieve simple recognition from a single correct response. No comparative studies were found that used the system.

Multimodal Interfaces

In the late 1980s, a shift was made to evaluate the utility of speech input in multimodal interfaces. Martin (1989) compared the performance of speech input with typed full-word input, single key presses, and mouse clicks for entering commands. The results of this study indicated that speech input was a more efficient response channel than the keyboard or the mouse.

In 1990 the comparison of multimodal interface was continued with a study of speech input used to control navigation in a Windows application while allowing keyboard and mouse input for other tasks (Karl et al., 1993). Although no difference in speed was noted between the two inputs, speech was superior when the windows were partially or completely hidden. A further study in multimodality cited by Karl et al. evaluated the use of speech input for graphical editors. Speech was used by an experimental group to enter commands and a mouse for pointing and selecting graphic objects. The control group used only a mouse. The results showed that there was a reduction in time for those using the speech input.

A similar effort in multimodality was done by Karl et al. (1993) to determine the utility of voice commands in parallel with a mouse as a selector and a keyboard for text entry compared with using a mouse alone. Called a "counterbalanced" design by the authors, the study investigated speech recognition used with word processing applications. Their belief was that using speech would be easier for word processing commands when using the keyboard for text, instead of moving from the keyboard to the mouse when a word processing command was required. This supported the study by Jonassen and Hannum (1987) that this type of movement was disruptive and stressed the psychomotor system.

Four tasks were given to the subjects for evaluation, with one group of tasks using voice-first and mouse-second and the second group using mouse-first and voice-second. The first task used only commands activated by voice or mouse; no typing was required. The second task required the subjects to type a short scientific formula that contained subscript, superscript, and bold text as well as Greek symbols. Word processing commands were activated as needed. The third task asked the subjects to build a table of symbols using editing commands. This task was unique in that it required the use of short-term memory to build the table. The subjects selected and copied the symbol, memorized the symbol description, moved down to the table, pasted in the symbol, and entered the symbol description. If the subjects could not remember the description, it was dictated by the authors while recording a memory error. The table with the symbols and descriptions is shown in Table 3.

Table 3

Building a Table of Symbols

Procedure: Use voice commands to copy and paste the symbol; then enter the symbol description.	m = Earth gravitational constant Re = Earth equatorial radius Ld = luminosity of the sun Ad = satellite surface area Re = satellite position vector Vd = satellite velocity vector h = satellite surface reflectance
Symbol	Description

Note. Adapted from Karl , Pettey, and Shneiderman (1993).

The fourth task required the subjects to type a short paragraph that contained subscripted, superscripted, italicized, and bold text. Again, word processing commands through speech recognition were used as necessary.

A subjective questionnaire was also developed to determine the acceptability of speech recognition as an interface as compared to the mouse. Results of the Karl et al. (1993) study showed that there was a significant reduction in the average time for each task when speech was used as an input . The effects of order (voice-first, mouse-second) and interaction were not significant.

An unexpected result of the Karl et al. study was that issuing voice commands interfered with short-term memory. Over half of the subjects noted that recalling descriptions was more difficult when using voice commands. In addition, the reduction in time was less than expected because the subjects paused to memorize the descriptions. The authors proposed that the interference in short-term memory was caused by an interaction between two tasks of the same modality. The verbal processing of the memorization requirement interfered with the verbal response modality of the speech user, but did not interfere with the manual modality of the mouse users. It was not clear if this interference would affect more complicated mental tasks such as problem solving and composition. The authors recommended further study.

To provide a practical measurement of how an interface implementation performs with respect to both learnability and subsequent recall, Murphy (1992) developed a framework for testing human-computer interface learning. This study defined a new criterion for quantifying human-computer learning and recall while providing a tool for designers to use that can determine learnability metrics during the design process. Murphy's framework tested an original interface design that attempted to improve human learning speed and memory retention using elaborative learning techniques and what he calls a "generation effect." The results of this study indicated that there were significant differences between interfaces with respect to recall performance. The present study integrates applicable measurement techniques validated by Murphy.

Concepts of Short-term and Long-Term Memory

To attempt an adaptation of any cognitive processing model or framework, a review of short- and long-term memory is useful. Although debate continues about the characteristics of short-term or primary memory, Travers (1982) offered four ways that it may be conceptualized: (a) as a store from which information is selected for transfer to long-term memory; (b) as a consciousness of information; (c) as a perceptual process; and (d) as an organizer prior to memorization.

The first concept (store) states that information must stay in the primary memory for a period of time before it can be transferred to long-term memory.

The period of time is dependent on the rehearsal effort to ensure the information is sufficiently ingrained before it is transferred. The long-term memory then becomes the result of the primary memory through a slow process of information being rehearsed, maintained, and transferred. This concept implies that more rehearsals enhance the retention of long-term memory.

The second concept (consciousness) is the store of information of which we are aware. This may be compared to the random-access memory (RAM) of a computer, which is the active processor.

The third concept (perceptual) links short-term memory with the immediate perceptions drawn from inputs from the sensory systems and the interpretation of these inputs from information drawn from long-term memory. The fourth concept (organizer) notes that short-term memory attempts to provide a structure for organizing random items before starting the rehearsal or memorization process. Travers (1982) provided the example of a study that presented subjects with jumbled items to memorize. The items were part of different groups such as animals, professionals, and vegetables. All the subjects placed the items in categories when they were asked to write down what they had seen.

All four concepts may converge and operate as one. According to Travers (1982), the organizing activities for short-term memory are essential for preparing material for long-term retention.

Although the short-term memory organizes items so that they can be memorized, the long-term memory is also organized for rapid retrieval of information. Several structural models provide the long-term retention. Travers (1982) discussed seven: (a) associative; (b) hierarchical; (c) propositions (d) schemata; (e) episodic and semantic, (f) attributes, and (g) piecemeal.

The associative theory originated with Aristotle and still continues to be the subject of many research efforts. It purports to be a complex network that links together a system of ideas with associations.

The hierarchical model suggests that memory is organized into general ideas followed by more specific ideas such as sparrows under birds Propositions are short versions that contain essential meanings, which allows a statement to be made that contains the essential meaning.

Schemata is a concept that is related primarily to learning behavior and is said to be structured in chunks or frames that provide a context for interpreting a perceptual input. Travers (1982) provided an example of a comfortable armchair. If seen in a home, it may elicit perceptual responses of relaxing, but if seen in a store, it may be looked at from the perspective of price or construction. This implies a link between an input and a reaction, which may result in appropriate or inappropriate behavior if the link is incorrect: for example, asking the price of the chair when in a private home. Transferring this concept to memory would incorporate the correctness of the link with the input and the retrieval of retained information.

Episodic and semantic systems form a dual memory structure. Episodic events are chronologically stored as they occur, along with a record of the incident. These are occasions, such as a first airplane ride or the ride to work through blinding fog the previous week. The semantic memory structure stores facts, such as the formula for a rectangle or the temperature for freezing. The facts are not part of any episode but are components of the cognitive system that may be recalled to solve a multitude of problems.

Attributes are characteristics of experiences that are stored in memory. This is similar to the theory of perception in that the recognition is achieved by a person's analyzing his or her experience input based on an object's attributes and then determining what the object is.

One other theory of long-term memory is that information is stored piece-meal. Travers (1982) provides an example of people trying to recall rare words in which they could remember the first or last syllables or rhymes that sounded like the word. This example is often exercised in the game of charades.

Elaboration and Generation

One of Murphy's (1992) objectives in his attempt to find the best computer interface was to find a feature that would enhance the retention and retrieval of long-term memory. Two methods he cited are <u>elaboration</u> and <u>generation</u>.

Elaboration is the providing of additional redundant information at the time of learning. It is based, in part, on the hierarchical category defined previously, but the hierarchy is networked by nodes, which represent the major ideas. According to the network theorists cited by Murphy, information in the long-term memory can be retrieved by activating the nodes in the network. By increasing the number of elaborations, the number of retrieval paths is increased.

With this objective in mind, Murphy (1992) found that the number of elaborations was not as important as the appropriateness of the elaboration. He cited studies that indicated that the depth of cognitive processing was increased when the subjects generated the elaboration. Although not defined by Murphy, generation implies a method of verbalization. He cited three forms: making rhymes rather than reading them; using phonemic processing; and reading sentences upside down. Although the last form may not be practical, these studies indicate that, if cognitive processing can be increased at the time of learning, then the memory trace can be improved, thus providing longer retention of what is being learned.

The forms of generation all have a common basis: verbalizing some input that provides redundant information. From this commonality, Murphy proposed that a gain in recall may be possible with subjects who generate words or simple sentences that are meaningful to them. He further contended that this would be a useful feature of a computer interface. This contention is in consonance with

Fleming (1987) who observed that "learning is facilitated where the learner reacts to or interacts with the critical information" (p.249). He also noted that learning can be increased "from the generation of a sentence or paragraph context for embedding words to be memorized" (p. 250). According to Fleming , this is a form of mental elaboration on the given information that allows meaningful relations to be formed "between the words to be remembered and the sentence context." (p.250). Interaction with critical information also helps to hold the learner to the on-task activity and eliminate the off-task activity (Fleming, 1987). The relationship between the words to be remembered and the sentence context raises the level of cognitive processing from rote to conceptual. These forms of generation are similar to those cited previously by Murphy (1992).

Given from the previous discussion that natural language is probably one of the most promising interfaces "for use in the long term for simplicity of learning" (Murphy, 1992, p. 83), continuous speech recognition, combined with elaboration and generation, should have a positive effect on long-term memory and should affect the degree of learning.

Although Murphy's (1992) efforts are focused on finding an improved interface for performing tasks on the computer such as sending and receiving electronic mail, this study attempts to use Murphy's framework to transfer to learning facts and procedures from a CBT academic lesson. Moving from the premise that speech recognition may provide a viable interface for CBT, the next discussion describes the framework used for improvement in learning a computer task and transfers the structure to a platform for academic learning on the computer.

Framework Functions

Murphy's (1992) framework has four functions: (a) Strategy Selection, which involves choosing the best learning and test strategy; (b) Initial Learning Measurement, which includes an initial learning domain that is defined as the conditions present during the first learning of an interface task; (c) Learning Retention Measurement, which provides a measurement of the learning retrieval domain, which is a subsequent time to the initial learning in which the user retrieves and applies the previously learned skills and knowledge; and (d) Task Learning Effectiveness, a quantitative metric of the performance of the interface.

The premise for this framework is that a computer task involves initial learning, and that the knowledge is retrieved at a later date to accomplish the same task. Depending on the amount of rehearsals (practice), the task may be completed but may incur a percentage of errors. The time between the first learning occasion and the second causes a decay in the memory that stores the knowledge and skill to do the task. If the interface can cause more cognitive processing at the time of learning, then the retrieval of the knowledge and skill is easier and can be more readily applied to the next new task. This concept follows many instructional strategies applied to CBT, such as presentation, drill and practice, and the relationship of old to new (Lillie, Hannum, & Stuck, 1989), but it is in the subset of variables in the computer interface that Murphy (1992) attempts to increase the effects on initial learning and learning retention.

Whereas the individual differences variables, task characteristics, and learning style variables are important elements in Murphy's (1992) study, the present study was limited to two of the categories of the Initial Learning Domain Variables: Learning Strategy Variable Set (LS Variables) and the Original Learning Variable Set (OL Variables) as shown in Table 3. The categories of Individual Differences Variables Set (ID Variables) and Task Characteristics Descriptor (TC Descriptor) were not used. This is a practical approach because many instructional situations bring together an audience of mixed attitudes, experience, age, sex, and learning styles and would require an extensive effort to determine the effect, if any, of these variables. The Learning Strategy Variable Set (Table 4) is applicable when choosing a learning strategy that matches the task characteristics. This variable is already given, because the lesson in this study was developed using an instructional strategy consisting of a tutorial for presenting factual information that combines drill and practice. The element of speech recognition becomes part of the instructional strategy for the experimental group of the study, thus providing a match for the "task," which is to learn the lesson content. The Retained Learning Variable Set (RL Variables) in

Table 5 was not used due to time and logistic considerations. This left the selection of applicable variables of Original Learning (Table 4): OLtime, OLerrors, and OLpass. The variables "OLiterations" and "OLlevel" in this set were not used because only one iteration was allowed, and no level or threshold of proficiency was established for this study.

The variable OLtime is determined by measuring how much time has elapsed for the user to complete the task (Murphy, 1992). The task in this study included the diagnostic questions and the achievement test. The variable OLerrors was used to measure the number of errors made during the original learning session, whereas OLpass indicated whether or not the user achieved a designed benchmark. The latter included a combination of OLtime, OLerrors, and the benchmark indicator.

Farr (1987) found that the most important determinant of knowledge and skill retention is the degree of original task learning. To significantly impact original learning, and subsequent retention of that learning, the short-term memory must also be affected since it is the organizer of the long-term memory activities (Travers, 1982). It can be argued that there is no long term-memory without short-term memory. Accordingly, short-term memory must be strongly affected by the original learning situation.

Table 4

Initial Learning Domain Variables

Individual Differences Variables Set (ID Variables)
*Attitude *Experience *Age *Sex *Learning Style
Task Characteristics Descriptor (TC Descriptor)
*Task Type *Task Complexity
Learning Strategy Variable Set (LS Variables)
*Appropriate Learning Strategy (s)
Original Learning Variable Set (OL Variables)
*OLtime *OLerrors *OLlevel *OLpass *OLiterations

Note. Adapted from Murphy (1992).

Table 5

Learning Retrieval Domain Variables
Retained Learning Variable Set (RL Variables)
*RLinterval *RLtime *RLerrors *RLlevel *RLpass *RLiterations

<u>Note</u>. Adapted from Murphy (1992).

Learning Effectiveness Evaluation

To provide a means of quantifying the effectiveness of his framework, Murphy (1992,) cited summary metrics derived from usability engineering practices. His total Learning Effectiveness Evaluation includes both Original Learning (OL) and Retained Learning (RL); however, this study was limited to the examination of OL effects, so the focus was on the formula for this category. The OL variable set from a new interface treatment is compared to a reference set, based on time required and expressed in a ratio normalized by the reference standard. The formula for the OLeffectivness Score for OL effectiveness is:

$$\text{OLeffectivness} = \frac{\text{OLtime(ref)} - [\text{OLtime(test)} - \text{OLtime (ref)}]}{\text{OLtime(ref)}}$$

$$= \frac{2^{*}\text{OLtime(ref)} - \text{OLtime (test)}}{\text{Oltime(ref)}}$$

If the two interface treatments are equal, then a central value of 1 is achieved. A score of less than 1 indicates less effectiveness, and a score of greater than 1 indicates greater effectiveness with respect to time. Although the resulting score is based on a test for the level of proficiency achieved on a computer task, an attempt was made to overlay the formula with variables from this study. OLtime(ref) related to the time of the experimental group since they were using the conventional interface (mouse), and OLtime(test) equated to the time of the control group interface (speech recognition).

CHAPTER III

METHODS AND PROCEDURES

This chapter describes the study's research population, sampling procedure, data gathering instruments, and data analysis procedures.

Population and Sampling Procedures

The sample was selected from the Shackleford Junior High School population in Arlington, Texas, made up of seventh-grade students from the computer literacy classes ranging in age from 12 to 14. The students were categorized by the teacher as being average to good with computer applications such as word processing, hyper-card, and graphics. A control group of 20 subjects and an experimental group of 20 subjects were formed by a chance drawing of odd and even numbers from a container. Those persons who received odd numbers composed the experimental group. According to Kirk (1982), a minimum group number of 17 is sufficient to obtain a power of .80 with α at .05 when two groups are involved. Using Cohen's rule of thumb estimate for a power .80, a small effect size $d = .2$ with sample size $n = 155$; a medium-effect size $d = .5$ with $n = 25$; and a large effect size $d = .8$ with $n = 10$ (Kirk, 1982). Using this estimate with an effect size of .6 yields $n = 17$.

Both groups were given a CBT lesson previously developed for aircrew training that contained generic concepts that did not require a knowledge of flying as a prerequisite. The control group took the lesson using a mouse as the computer interface. The experimental group used speech recognition as the computer interface; however, the mouse was also used as a default interface. All classes were given a brief overview of the lesson followed by a review of detailed instructions with each individual. The subjects then took the lesson in a separate room that was free from background noise except for the school's public announcement loudspeakers. These speakers interfered with one subject's speech recognition interactions for approximately five minutes and intermittently with several others.

Data-gathering Instruments

The data-gathering instruments consisted of a CBT lesson and the CBT hardware. The lesson included test scores, diagnostic question scores (progress checks), and time-on-task. The lesson was called "Threat Avoidance." It consisted of a tutorial that presented declarative and procedural knowledge, provided a medium amount of interaction, and contained progress checks after "chunks" of information, in addition to the multiple choice achievement test. The lesson had been validated by Hughes Training, Inc., in a program for aircrew training on the jet transport aircraft C-141 and resulted from the formal Instructional Systems Development (ISD) process using subject matter experts, instructional designers, curriculum developers, and course evaluators. It had been through two formal tryouts and was integrated into the C-141 Aircrew Training Center curriculum in early 1994. Sample test questions are in Appendix A. Hughes' Training Test and Evaluation staff determined the test to have a reliability (r) coefficient of .78 using the Kuder - Richardson (KR-21) formula.

Lesson Interaction Procedures

Specific items of interaction included the selection of highlighted objects on the screen, response to multiple choice items on progress checks (diagnostic questions), response to choices in a threat exercise, selection of the icon for the next screen, and response to multiple choice items on the lesson test. Immediate feedback was provided for exercises, progress checks, and multiple choice tests.

Control Group (Mouse)

After drawing even numbers, the subjects were presented with the title screen of the lesson. They were given verbal instructions to use the left button on the mouse to click on highlighted areas of the screen or a right arrow at the botton of the screen to move forward in the lesson. Since the subjects were familiar with the use of the mouse, no further instructions were given, nor were they required.

Experimental Group (Speech Recognition)

After drawing odd numbers, the subjects were presented the screen with the following instructions, shown in Figure 2:

1. Place the headset and microphone correctly.
2. Speak by pressing the right mouse button and holding it down.
3. Identify the cursor which indicated it was waiting for a speech input.
4. Identify the speaker response symbol.
5. Say the word or phrase following the speaker response symbol.
6. Recognize the function of the hourglass cursor.
7. Recognize the function of the dialog box.
8. Use the left mouse button to select "OK" on the dialog box when speech recognition failed, and the right mouse button to try the speech recognition again.

The subjects were then verbally told that, if the dialog box appeared three times, indicating speech recognition failure, they would use the left button to select "OK," then select the item of interaction on the screen by pointing the cursor to the item and clicking with the left button so that the lesson could continue. In some cases this was a highlighted object such as the first item on the checklist, and on others it was the right arrow at the bottom right corner of the screen. When the next screen appeared, the speech recognition would reset and the subjects would have to use the right mouse button and attempt inputs again by voice.

Place the headset over your ears.
Ensure the microphone is in the proper position.
To speak, press the right mouse button and hold it down.
A cursor will appear that indicates it is waiting for you to speak.
When you have finsished speaking, release the button.

When you see the speaker response symbol, (((( )
Say the words or phrase following it.
The hourglass cursor will appear,
indicating it is processing your words or phrase.

Example: As you read these instructions, the speech recognition system will be loading. When complete, a Speaker Response Symbol will appear at the bottom right corner of the screen. Say "next screen" to continue.

If a dialog box appears that looks like this:

select "OK", then hold down the right mouse button and say the words or phrase again.

(((( )Next Screen

Figure 2: Speech Recognition Instructions

After they read the instructions and given verbal clarification, the subjects were asked to note the speaker response symbol at the bottom right corner of the screen and say the phrase, "Next Screen." This brought up the title screen, and the subjects proceeded with the lesson.

When two or three lines of text that contained critical phrases had to read, the speaker response symbol did not appear until after a pause of 5 seconds. The pause allowed the subjects to scan the information, then to state the word or phrase. For those screens with critical phrases, correct recognition caused the next screen of information to appear. Therefore, the speakers had to say the critical phrase and achieve good recognition before they could progress.

The use of critical phrases was a method of applying Fleming's (1987) learning strategies with a new technology: generation of sentences and mental elaboration, assuming that the learner is active, overtly or covertly, and holding the learner to the on-task activity while eliminating the off-task activity. As discussed earlier, these factors were supportive of the forms of generation and elaboration cited earlier by Murphy (1992).

Lesson Structure

The lesson was primarily a visual presentation, with extensive use of graphics and still video. It was presented in the following sequence: (a) Introduction; (b) Segment A: Combat Entry Checklist; (c) Segment B: Threat

Avoidance Support Duties; (d) Segment C: Combat Exit Checklist; (e) Review; and (f) Lesson Test.

The introduction contained an overview of the lesson and the lesson objectives. Segment A -Combat Entry Checklist provided a graphical representation of the checklist used by the C-141 crew to enter a combat zone (Figure 3). Initials on the checklist indicated which crew emembers were involved: P Pilot, CP = Co-Pilot, N = Navigator, E = Engineer, S = Scanner or Observer, and LM = Loadmaster. The checklist items were to acknowledge the initiation of the checklist, secure the survival equipment, set the aircraft's internal and external lights, secure loose items within the plane, position observers to watch for threats, and acknowledge the completion of the checklist. The subject followed the checklist through video and graphics which illustrated the procedures.

As each item was completed, the next item was highlighted. The control group used the mouse to click on the highlighted item. The experimental group had to say the checklist item, i.e., "Survival Equipment Secured." After the item was selected or recognized, the first screen of information was presented. The learner then interacted with the continuing screens by selecting a highllighted item.

If only text was presented with a graphic or video and no other interaction, the control group used the mouse to select a right arrow to get to the next screen.

The experimental group had to say "Next Screen," which was cued by the Speaker Response Symbol (((( ) (Figures 3 - 9).

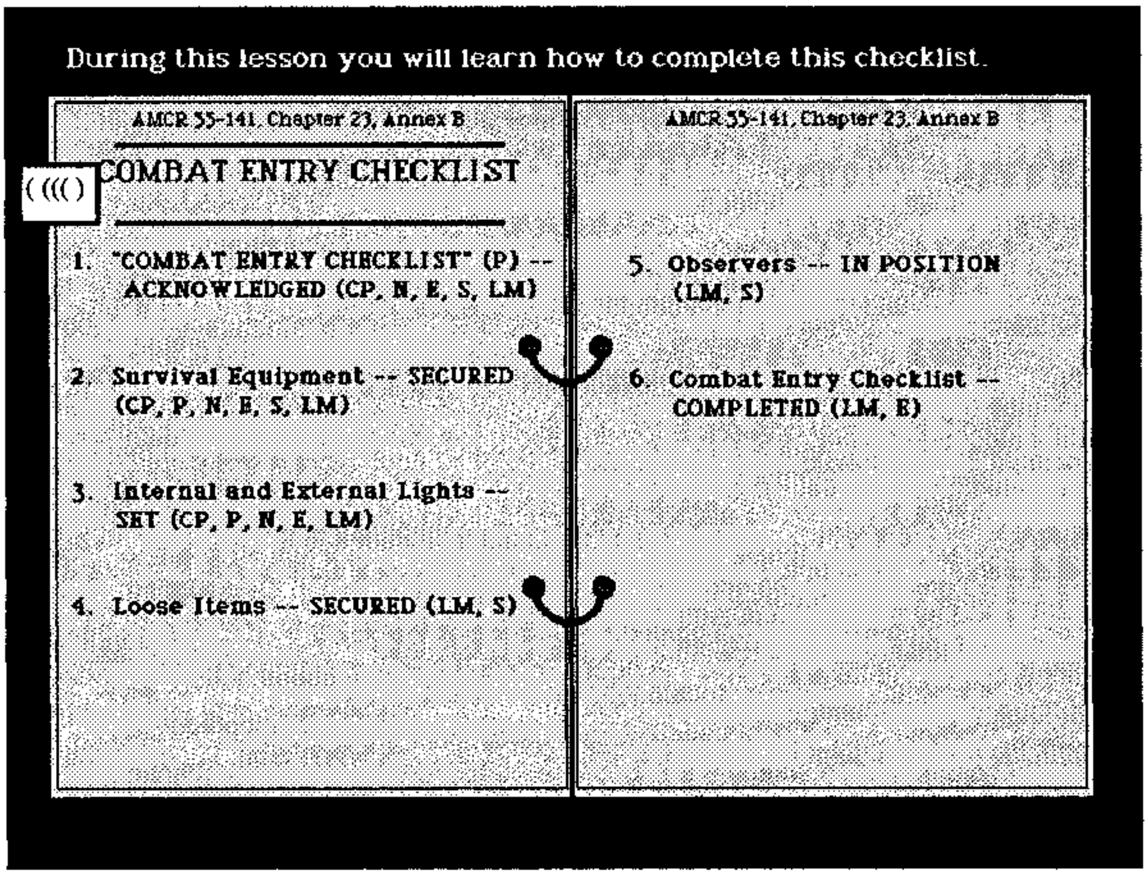

Figure 3. Combat Entry Checklist

Segment B, Threat Avoidance Support Duties, included instruction in how to scan for threats to the aircraft , how to identify the location of the threats in relation to clock positions, and what information was needed to pass to the pilot to avoid the threat. An example of scanning areas in relation to the aircraft is shown in Figures 4, 5, and 6. An example of the information to be passed to the

Segment C, Combat Exit Checklist (Figure 9), mirrored the presentation of Segment A. A graphic of a checklist was presented, with each of the procedures illustrated by graphics or video. The procedures were a reversal of the Combat Entry Checklist. These were to acknowledge the initiation of the checklist, reposition the observers or scanners to their normal station on the plane, assess and report any battle damage, check the survival equipment, check the internal and external lights, and report the checklist completed. As with the Combat Entry, the checklist items were highlighted, then chosen by the subject with the mouse or with speech recognition. The tutorial screens then presented the lesson information. An example of the Battle Damage information screen is shown in Figure 10.

Syntax Development

To enable the Authorware System to run in conjunction with the speech recognition application, a syntax had to be developed that used the utterances associated with the CBT lesson. The syntax allows the recognizer more flexibility by defining categories of related items. The two types of syntax categories were word and phrase-structure (Phonetic Engine 400, 1993). The word category consists of a list of single words and phrase-structure consists of a grouping of words, phrases, and/or category names.

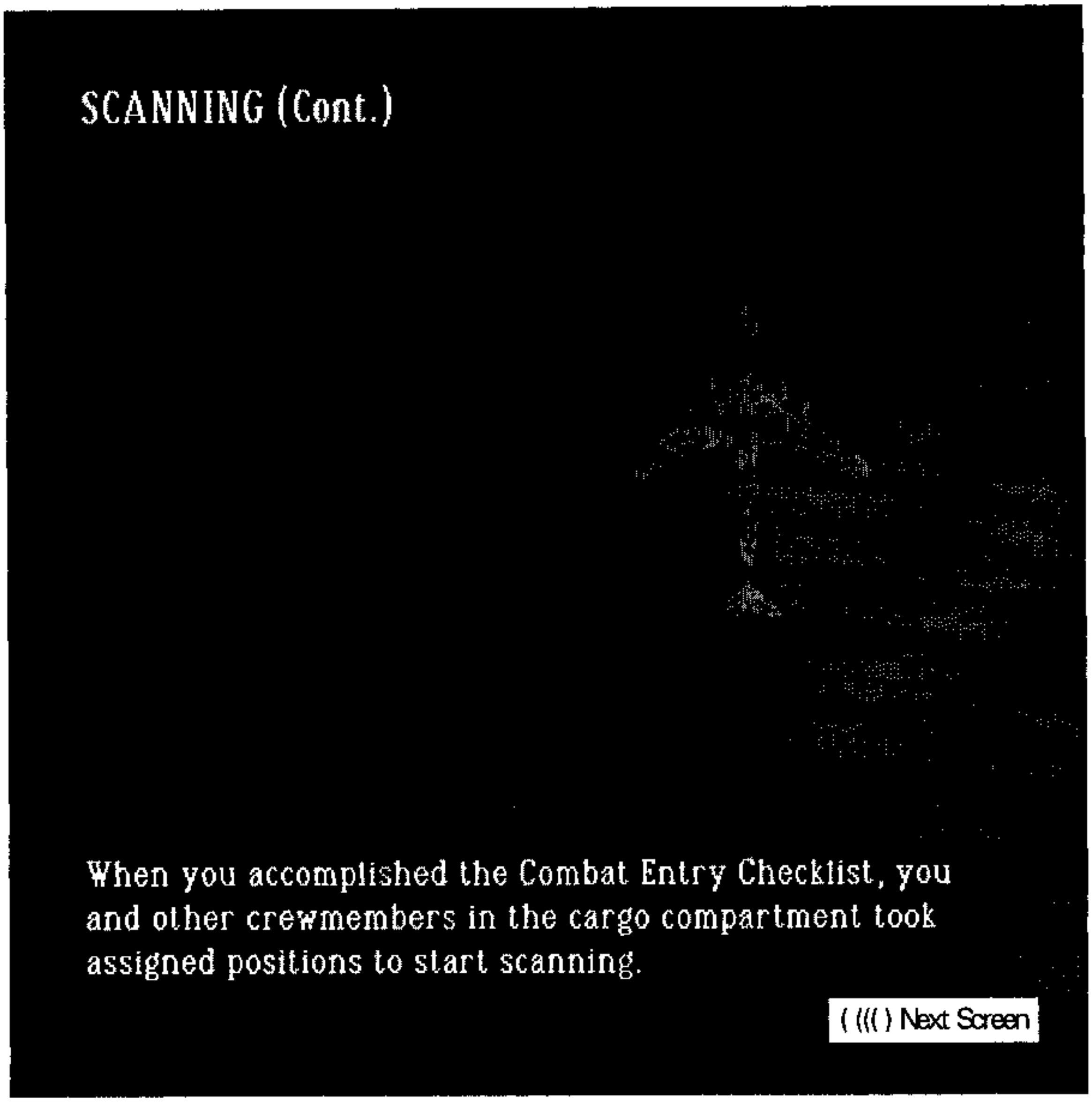

Figure 4. Scanning Areas

To define the categories, symbols called operators were used (Phonetic Engine 400, 1993):

== (two "equals" signs) identify a word category

> ("greater-than" sign) identifies a phrase-structure category

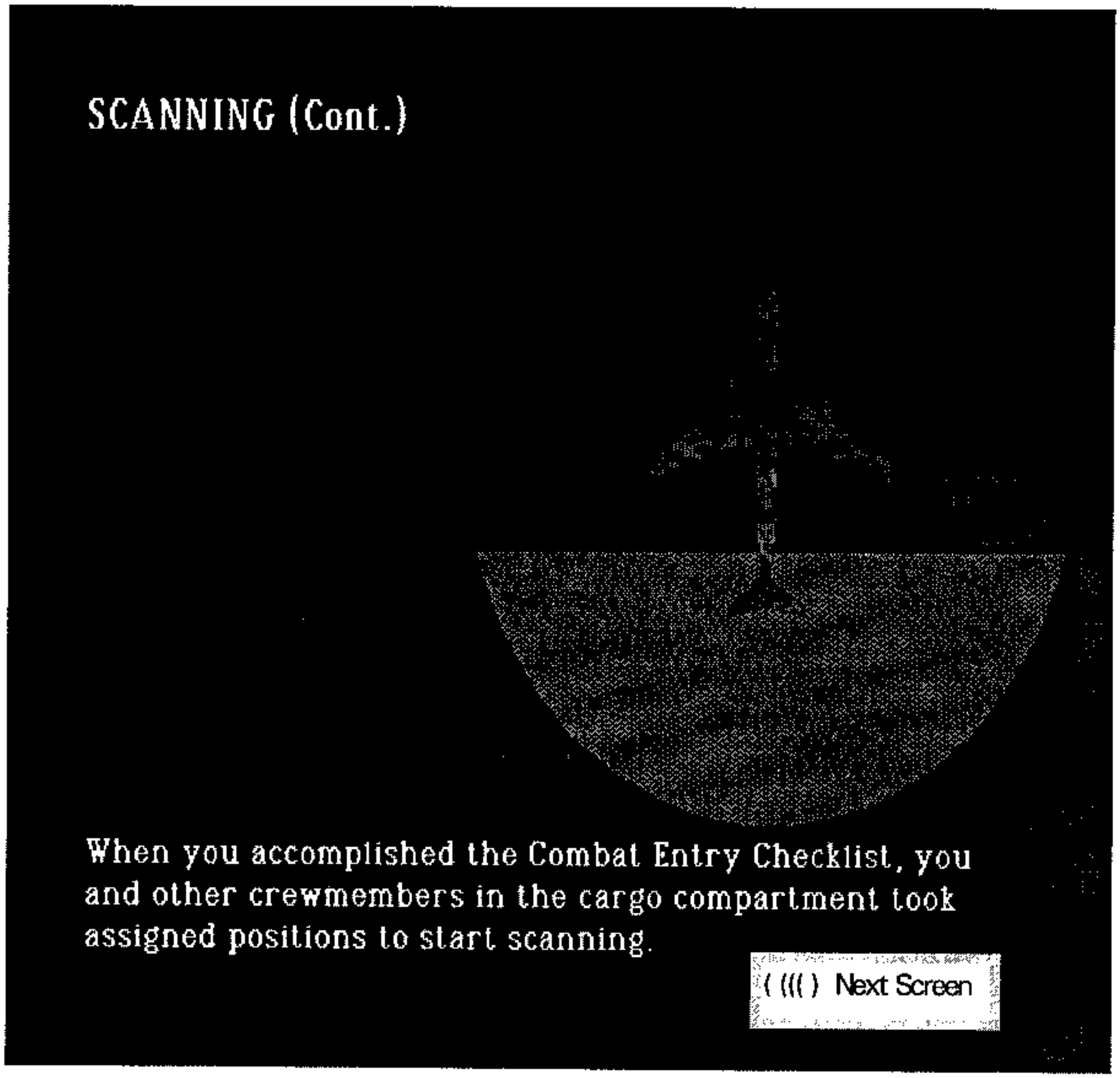

Figure 5. Scanning Areas

The category definitions use these kinds of forms:

word_category_name == word1 word2 word3....

phrase_category_name -> word1 _ word2 word3 _ category1 _ category2

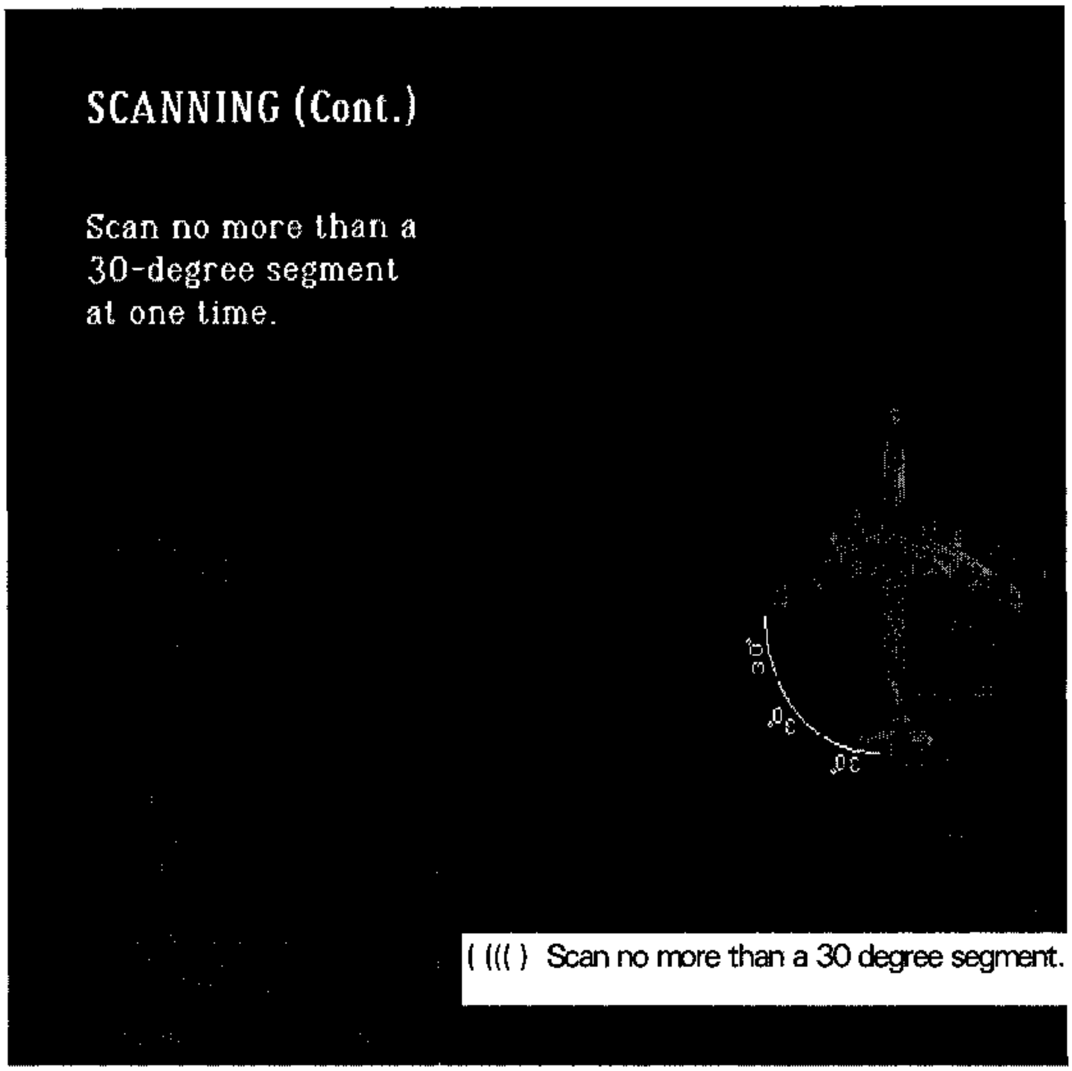

Figure 6. Scanning Segments

The underscore (_) and the notation symbol (_) are characters that allow the syntax to be compiled or put together in logic for the recognizer. Other characters are the plus sign (+), the minus sign (-), the apostrophe ('), the period (.), and brackets ([ ]).

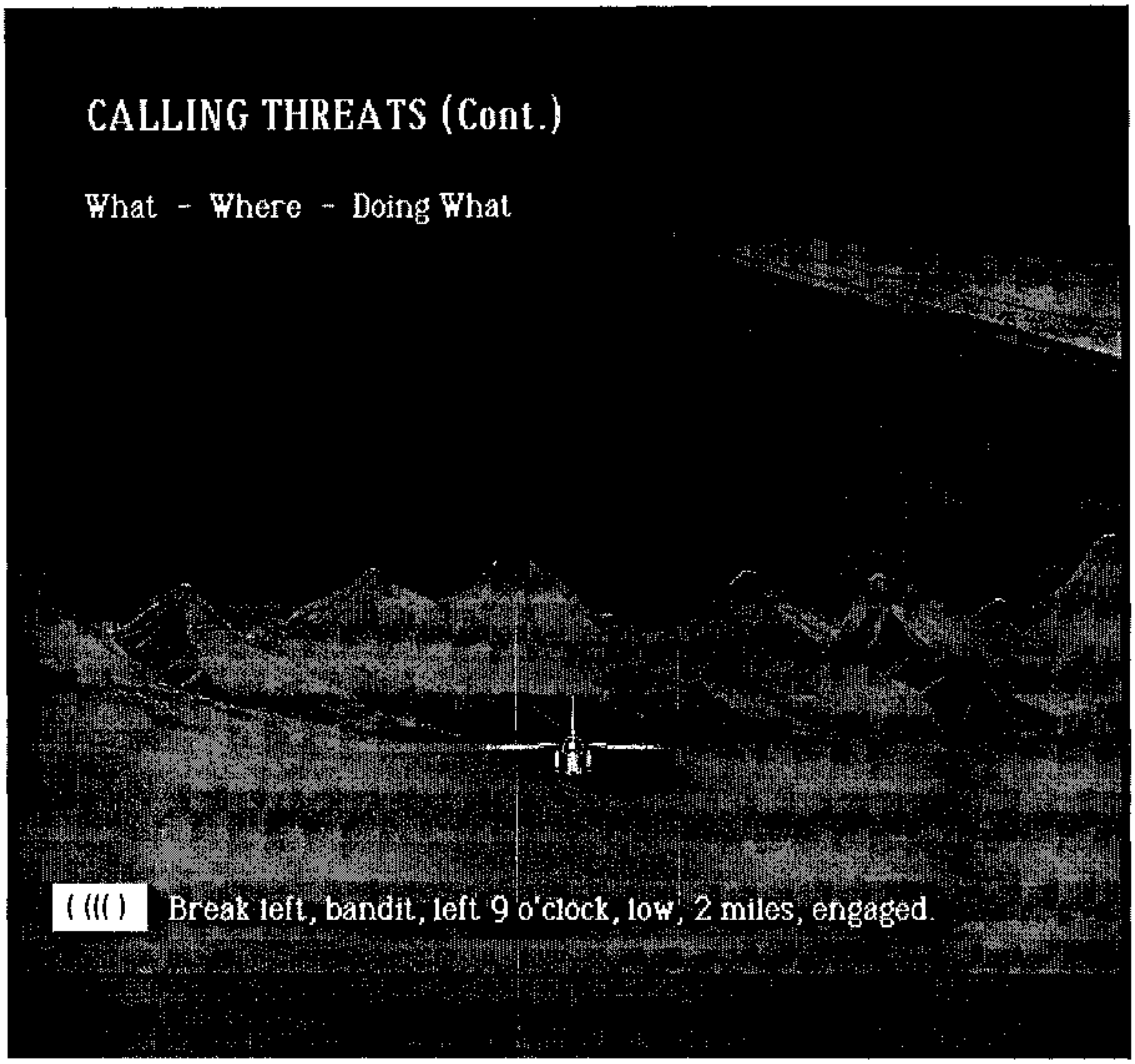

Figure 7. Calling Threats

One other key item in developing a syntax was the sentence rule, which allowed the developer to define complex phrases rather than simple sequences of words and categories (Phonetic Engine 400, 1993). A sentence can be in a traditional structure such as, "Instructional designers use different instructional strategies," but it is generally a rule that begins with a sentence rule definition

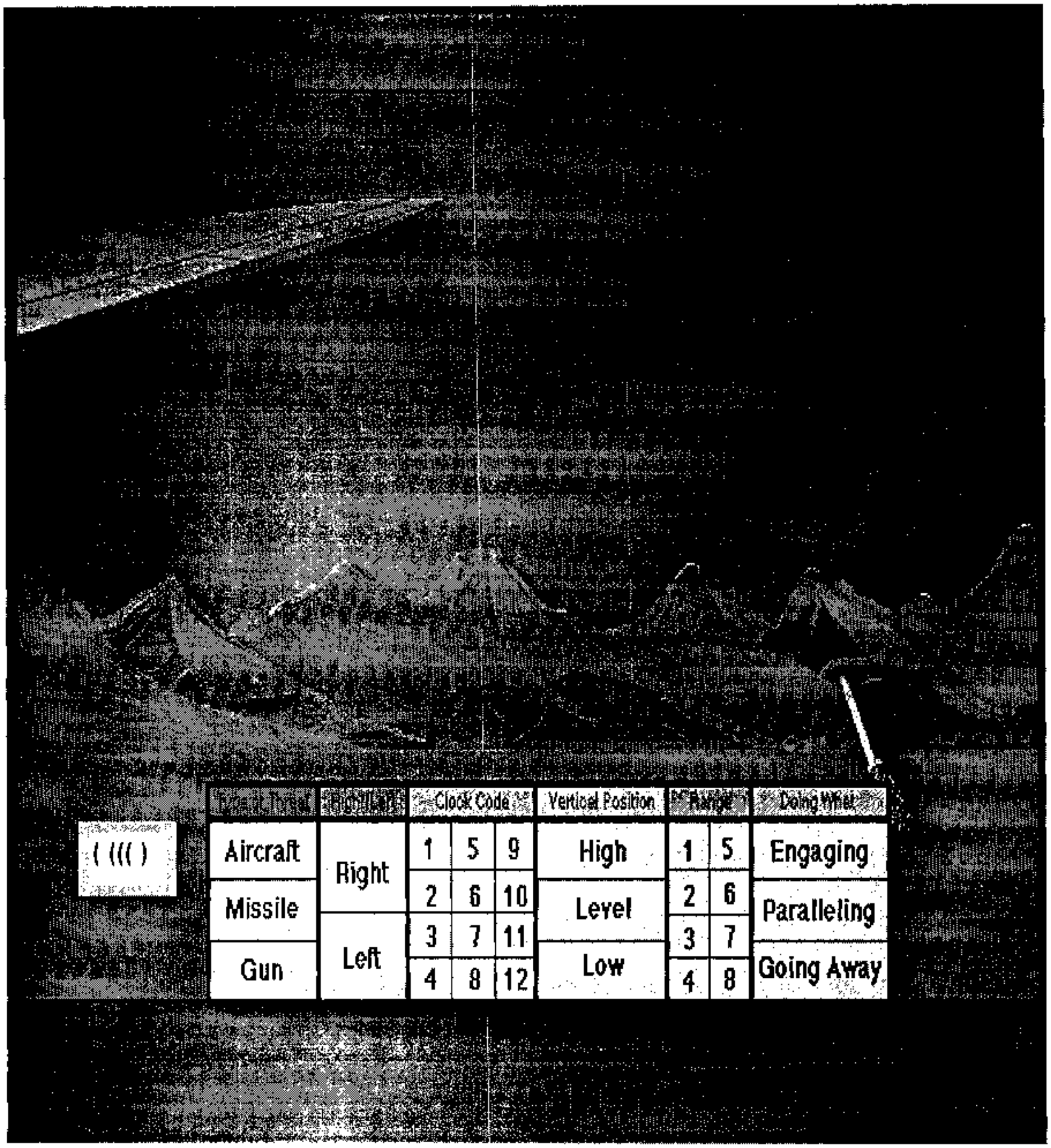

Figure 8. Threat Call Practice Chart

symbol, which is an "S". The S is used with the symbols --> to indicate that the sentence rule is defined. The following example shows how the word

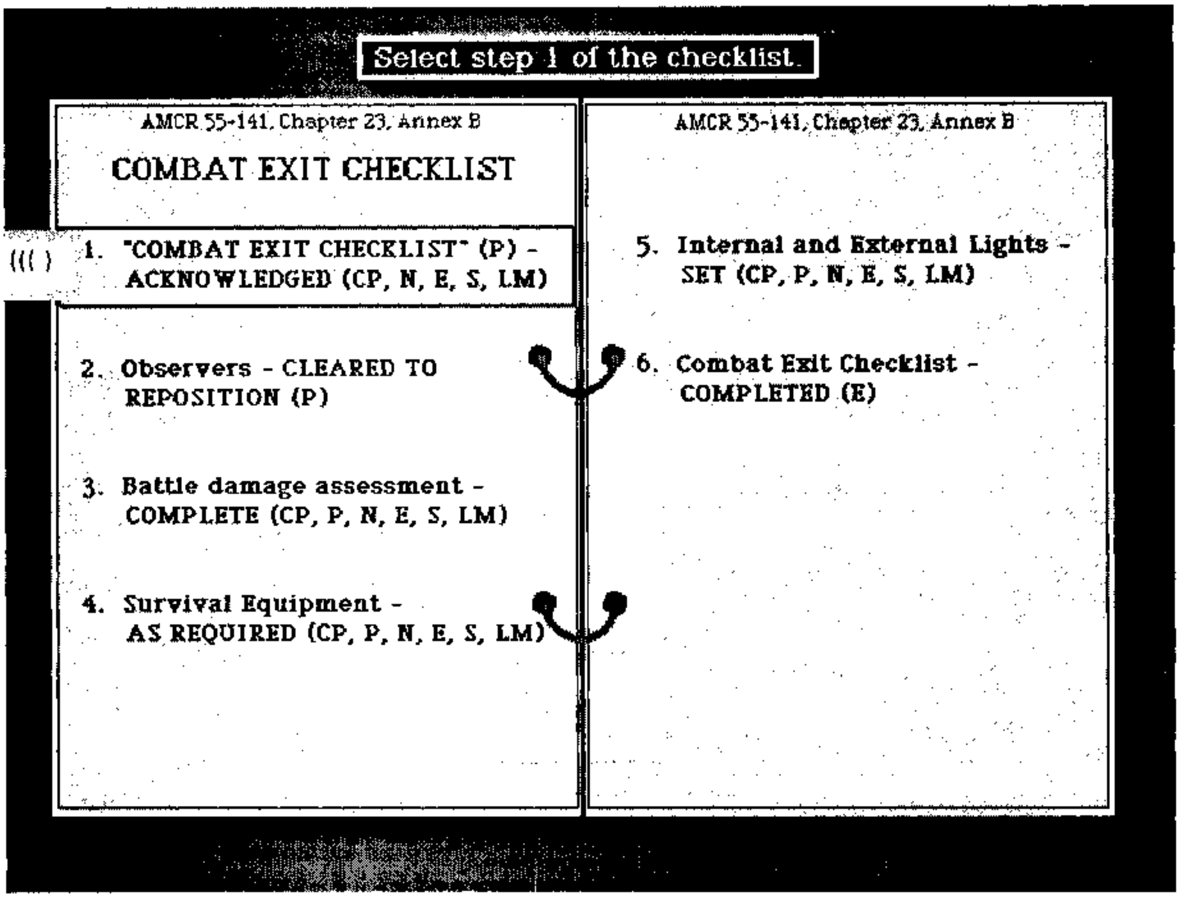

Figure 9: Combat Exit Checklist

category, the phrase-structure category, and a sentence rule were used together to allow an increased flexibility in the creation of a syntax (Phonetic Engine 400).

machine+ == computer device fax telephone

device type+ --> new equipment _ machine+_ expensive printer

S--> the device_type+ was working all day

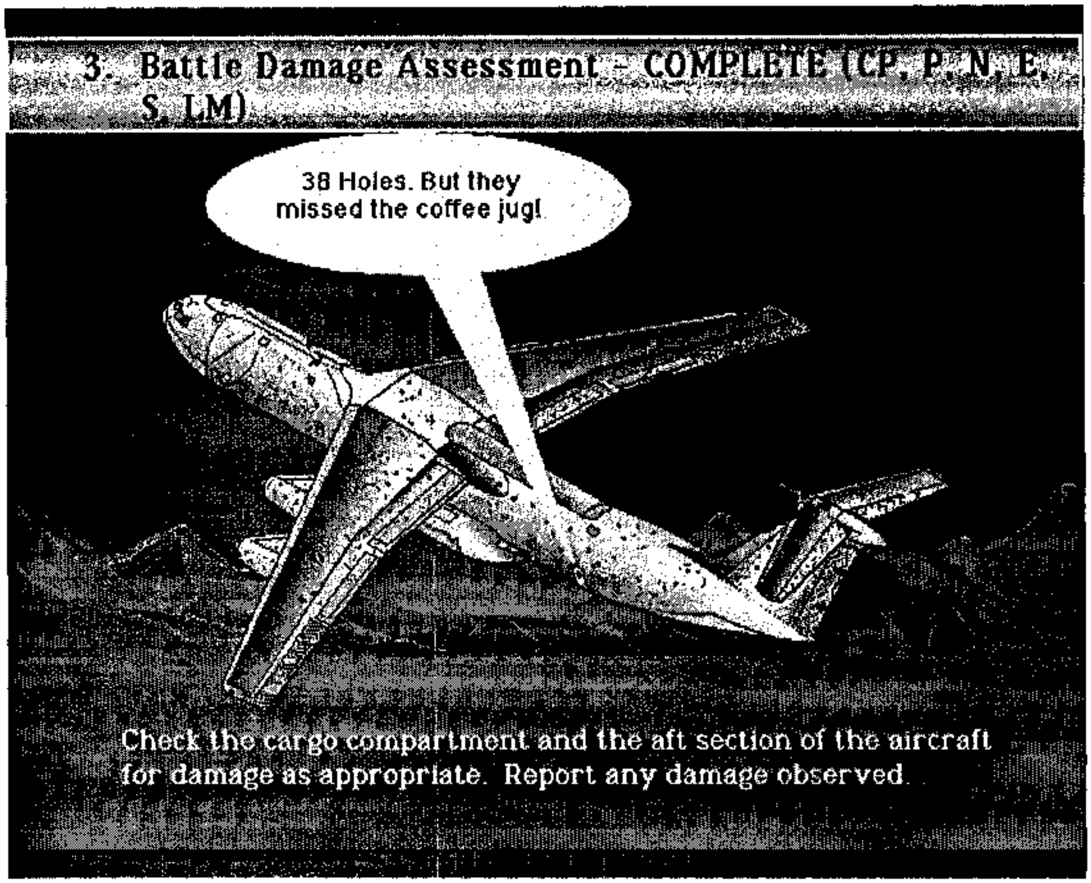

Figure 10. Example of a Battle Damage Information Screen

By using the combination of word category, phrase-structure category, and a sentence rule, the following six sentences can be presented more efficiently:

S--> the new equipment was working all day.

S--> the computer was working all day.

S--> the device was working all day.

S--> the fax was working all day.

S--> the telephone was working all day.

S--> the expensive printer was working all day

A sample of the syntax source file for the lesson is shown in Table 6. The remainder of the file is presented in Appendix C.

If the factors of generation and elaboration have been demonstrated to increase cognitive processing, they must have done so without the use of speech recognition, because the technology did not exist at the time of the previous studies. What additional factor can be added then, by using the technology? The assumption in this study is that, in addition to being overtly involved with the elaboration and generation of critical phrases, the learners are also motivated to achieve good recognition. This means that they must say the phrases loud enough and with sufficient enunciation for the recognizer to reward them with "good feedback" (the brief indication of the icon that it is processing the input). The good feedback is reinforced by "forward movement," the presentation of the next screen of information. These activities help to focus the learner's on-task behavior noted by Fleming (1987) in the previous discussion. An example of a text screen with a critical phrase and the speaker response symbol is shown in Figure 11.

Hardware for the Study

The equipment consisted of the following items: an IBM compatible 486SX PC with a 66MHz processor; a Sony Lasermax videodisc player; a Speech Systems continuous, independent speech processor; a stereo headset with attached unidirectional microphone; and a mouse.

Table 6

Sample of the Syntax Source File

```
NUMBERS+ == One Two Three Four Five Six Seven Eight Nine Ten Eleven Twelve
     Thirteen Fourteen Fifteen Sixteen Seventeen Eighteen Nineteen Twenty
     Thirty Forty Fifty Sixty Seventy Eighty Ninety Hundred Thousand Million

RESPONSE_WORDS+ == A Aircraft B Battle C Copilot D Eight Eleven Engaging
     Green Gun Helmet High Interphone Introduction Left Level Low Missile
     Navigator No Observers Parachute Paralleling Pitch Proceed Radar Red
     Results Right Scan Seven Summary Ten Turning Twelve Yes
!
! Allowed Verbiage
!
AIRSPEED -> Airspeed Indicator

ALL_PERSONNEL -> All Personnel Required To Be Mobile

APPLY_RESTRAINTS -> Apply Additional Restraints To Equipment At Risk Of Coming Loose

BANDIT -> Bandit Left Eight O Clock Level Five Miles Turning Left

BREAK -> Break Left Bandit Left Eight O Clock High Three Miles Engaging

BRIGHT -> Put the BRIGHT DIM Switch On Bright

CHANGE_MODEL -> Change Model

CHEMICAL_DEFENSE -> Chemical Defense Ensemble

CLOCK_CODE -> Clock Code

CLOCK_SEGMENT -> NUMBERS+ Clock Segment

CMBT_ENTRY_CHCKLST -> Combat Entry Checklist

CMBT_ENTRY_CHCKLST_ACK -> Combat Entry Checklist Acknowledged

CMBT_ENTRY_CHCKLST_CMPLT -> Combat Entry Checklist Completed

CMBT_EXIT_CHCKLST -> Combat Exit Checklist

CMBT_EXIT_CHCKLST_ACK -> Combat Exit Checklist Acknowledged

CMBT_EXIT_CHCKLST_CMPLT -> Combat Exit Checklist Completed

COMING -> Coming In Range Of SAM batteries

COMMANDER -> Aircraft Commander
```

Emission Reduction

After setting the lights, the flightdeck crew will take steps to reduce electromagnetic emissions that might reveal the aircraft's position.

Non-essential radios will be turned off.

Radar and the radar altimeter may be turned off as well.

(((( ) Reduce electromagnetic emissions

Figure 11. Critical Phrase with the Speaker Response Symbol

The equipment was set up on a three-by-five foot table that allowed the subjects easy access to the headset, mouse, and computer-screen.

Software for the Study

The software included the following applications: Windows application for DOS; programming language "C" which was built for the authoring language, Authorware; and Speech Systems, Inc., System Development Kit with associated manuals, include the PE400 User's Guide™, the SPOT ™ Programmer's Guide, and Syntax Development.

The application code, developed by a software engineer from Hughes Training, Inc., contains specialized language, and it was not the intent of this study to explain each function; however, it is included to demonstrate its validity in Appendix B.

The CBT lesson was converted from SAGE, a Windows application of WICAT, to Authorware. The conversion process saved development time from original text and graphics, but was still time-consuming because each screen had to be captured as a bit-map to a disk, then stored as a separate file in the Authorware system. Capturing of screens was done with a software package called TEMPRA. The original video laser disc was used with the Authorware system and had only to be inserted on the correct frame. The conversion enabled the stand-alone data-collection functions inherent in Authorware. The functions used were (a) to record the time to complete each lesson, (b) to capture diagnostic question scores which include the "call-out" drill in Segment B, and (c) to record lesson test scores. An example of the printout used to collect the data is shown in Table 7.

Table 7

Data Collection Functions

Date - Time
Speech Recognition Active...
In Release Mode...
Total Time: minutes
Number of callouts correct:
Number of callouts incorrect:
Number of progress check questions correct:
Number of progress check questions incorrect:
Number of final test questions correct:
Number of final test questions incorrect:

Design

The experiment was designed according to a Posttest-Only Control Group Design, which is diagrammed below (Kerlinger, 1986):

R		O_1 (Control)
R	X	O_2 (Experimental)

R = Random selection
X = Treatment (type of interface)

O_1 and O_2 = Posttest

Data Analysis Procedures

This section includes a restatement of the research hypothesis and the analysis procedures.

There will be no statistically significant difference in original learning effect when speech recognition is used as a computer interface for a CBT lesson compared to using a mouse as the computer interface. Independent variables are the types of interface. Dependent variables are achievement scores, progress check scores, and time-on-task.

The data were processed on a PC, using the statistical package, SPSS/PC+. The variables were tested simultaneously at the .05 level of significance, using Hotelling's $\mathbf{T}^2$ for multivariate analysis. Hotelling's $\mathbf{T}^2$ analysis involves generalizing from the univariate $\boldsymbol{t}$ to a multivariate statistic. The null hypothesis for a univariate $\boldsymbol{t}$ is: $Ho: \mu_1 = \mu_2$ *(The population means are equal)*. For multivariate data, the null hypothesis is:

$$Ho: \begin{bmatrix} \mu_{11} \\ \mu_{21} \\ . \\ \mu_{p1} \end{bmatrix} = \begin{bmatrix} \mu_{12} \\ \mu_{22} \\ . \\ \mu_{p2} \end{bmatrix}$$ *(the population mean vectors are equal)*

According to Stevens (1986), Hotelling's **T^2** can be used to test the hypothesis that there is no difference between Group 1 and Group 2 when compared simultaneously on y_1, and y_2. The following numbers are used for demonstrating the integration of data into the T^2 formula:

Group 1* (Control/Mouse) (independent variable)		Group 2* (Experimental/Speech Recognition) (independent variable)	
Test	Time	Test	Time
(dependent variables)		(dependent variables)	
y_1	y_2	y_1	y_2
1	3	4	6
2	7	6	8
2	2	6	8
--	--	5	10
$y_{11} = 2$	$y_{21} = 4$	5	10
		4	6
		--	----
		$y_{12} = 5$	$y_{22} = 8$

* In the actual experiment, n = 20.

The formula for Hotelling's **T^2** is:

$$T^2 = \frac{n_1 n_2}{n_1 + n_2} (y_1 - y_2)' S^{-1} (y_1 - y_2)$$

The means of the variables from the univariate **t** are replaced by the vectors of means in each group. The estimate of the assumed common within-population variance for **t** is noted by s^2 ,which is calculated from the within sums of squares for Groups 1 and 2. The multivariate generalization, ***S***, is used to replace the univariate measure of the pooled estimate of the common within-population variance and becomes the estimate of the assumed common population covariance matrix (R. Harris, 1985). To obtain ***S*** for multivariate calculations, the within-groups variability is replaced by matrix generalizations W_1 and W_2, which represent the within variability in the dependent variables in each group. There are two variables in Group 1, therefore, W_1 contains two variables with variabilities on each. These are noted as ss_1, and ss_2 with covariables as ss_{12} *and* ss_{21}. Matrix W_1 appears in the following structure (Stevens, 1986):

$$W_1 = \begin{matrix} ss_1 & ss_{12} \\ ss_{21} & ss_{22} \end{matrix}$$

Matrix W_2 is constructed in a similar pattern and is the estimate of the

within variability on the variables in Group 2. The two matrices are then added and divided by the degrees of freedom to determine **S**, the multivariate error term (Stevens, 1986). The calculation to find ***S*** is shown in Appendix D (Stevens, 1986).

The SPSS/PC+ Multiple Analysis of Variance (MANOVA) program was used to calculate Hotelling's T^2. If the null hypothesis is rejected, then univariate test results can be examined to determine which variables are statistically significant. These are the same as the ***F*** values from a one-way analysis of variance. For two groups, the ***F*** values are the squares of the two-sample **t** values; therefore a one-way analysis of variance followed by Sheffe's contrast method would determine the significant variable or variables (R. Harris, 1985).

The example just presented was included to demonstrate the mathematical matrix inversion procedures and included only two dependent variables for each group, y_1, and y_2. The actual study included three dependent variables for each group:

y_1 = achievement scores

y_2 = diagnostic question (progress check) scores

y_3 = time-on-task

For more than two variables, a computer is more efficient for calculating the matrix inversions (Manly, 1986). The results of the analyses using these variables are presented in chapter 4. Chapter 5 discusses recommendations for further study.

CHAPTER IV

RESULTS OF THE ANALYSES

Summary

The purpose of this experiment was to evaluate the effectiveness of speech recognition as an interface for CBT. The speech recognition interface was compared with the mouse interface, using an academic CBT lesson previously taught to Air Force students. The method of interface served as the independent variable, with each group having three dependent variables: achievement test score (Y_1), diagnostic test score (Y_2), and time-on-task (Y_3). The hypothesis for the study was as follows:

> There will be no statistically significant difference in original learning effect when speech recognition is used as a computer interface for a CBT lesson compared to using a mouse as the computer interface.

A multiple analysis of variance (MANOVA) program was used to calculate Hotelling's $\mathbf{T}^2$ to determine the significance of the treatment effect. The MANOVA data input file is shown in Appendix F.

A preliminary review of the data indicated that MANOVA assumptions were met. MANOVA treatment effects were significant between groups[($F(3,36) = 0.00$, $p<.05$)] therefore, the hypothesis is rejected. Posthoc results (univariate

F-tests) indicated a significant difference in variable Y_1 (achievement test) [(F (1,38) = .032, p<.05)] and variable Y_3 (time-on-task)[(F (1,38) = .000, p<.05)]. Variable Y_2 did not indicate a significant difference[(F (1,38) = .671, p<.05)]. An illustration of the mean scores per variable is shown in Table 8. A printout of the analysis results is shown in Appendix G.

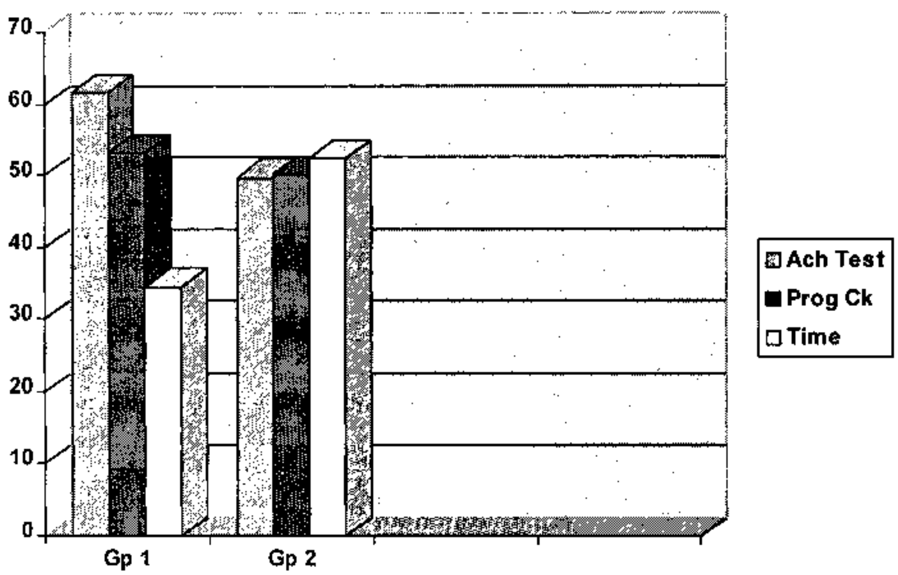

Figure 12. Mean Scores Per Variable

Cohen's estimate for effect size to attain a power of .80 for the test was confirmed by applying actual test values: (d = 61.6 - 49.8)/17.5 = .67

d = Effect size

61.6 = GP1 mean

49.8 = GP2 mean

17.5 = standard deviation

Substituting values to determine the sample size *n* with α at .05:

$$n = \frac{[\,(z_{.05})\ 1.645\ -(-z_{20})\ (-0.84)]^2}{(0.67)^2} = 14$$

According to Cohen's estimate and the formula cited by Kirk (1982), only a sample size of 14 was required to detect the standard deviation of 17.5, but 20 subjects participated.

The significant difference in variable y_1 indicates that, for this sample, the mouse was a more effective interface than was speech recognition. Although Murphy's (1992, p.103) formula for Original Learning (OLeffectiveness) is based on time, it can be applied to the mean scores of the achievement test to provide a measure of effectiveness.

$$\text{OLeffectivness} = \frac{\text{OLtime(ref)} - [\text{OLtime(test)} - \text{OLtime (ref)}]}{\text{OLtime(ref)}}$$

$$= \frac{2^{*}\text{OLtime(ref)} - \text{OLtime (test)}}{\text{OLtime(ref)}}$$

Substituting the mean score in the formula for the control group (Group 1) as the reference (ref) and the mean score of the experimental group (Group 2) as the test provides the following result:

$$OLeffectivness = \frac{OL(ref) - [OLtest) - OL\ (ref)]}{OL(ref)}$$

$$= \frac{2{*}OL(61.6) - OL(49.8)}{OL(61.6)}$$

$$= \frac{23.6}{61.6}$$

$$= .38$$

$OL(ref) = GP1, y_1$ (Mouse)
$OL\ (test) = GP2,\ y_1$ (Speech Recognition)

According to Murphy (1992), a central value of 1 is achieved if the two interfaces are equal; less than 1 indicates less effectiveness than the reference, and a score of greater than 1 indicates greater effectiveness. An OLeffectiveness score of .38 indicates less efffectiveness with GP2, y_1 than GP1, $y_{1.}$ Applying the same formula for variable $y_{3,}$ the following results are obtained with respect to time-on-task:

$$OLeffectivness = \frac{OLtime(ref) - [OLtime(test) - OLtime\ (ref)]}{OLtime(ref)}$$

$$= \frac{2{*}OLtime(ref) - OLtime\ (test)}{OLtime(ref)}$$

$$= \frac{34.4 - 52.3 - 34.4}{34.4}$$

$$= \frac{2(34.4) - 52.3}{34.4}$$

$$= .47$$

OLtime (ref) = GP1, y_3 (mouse)

OLtime (time) = GP2, y_3 (speech recognition)

An OLeffectiveness score of .47 indicates less efffectiveness with GP2, y_3 than GP1, y_3.

The results of applying these metrics support the results indicated by the analyses. For this sample, the mouse was a more effective interface than was speech recognition. A contributing factor could have been the difficulty in achieving good recognition by several of the experimental group subjects. Many had to pronounce each command or critical phrase two to three times to achieve recognition; however, a common block of words and phrases for both male and female subjects failed to achieve any recognition. This was further compounded by the substitution of incorrect answers on the progress checks and lesson tests. The progress checks did not allow answers to be changed, but this feature was available with the lesson test. For example, a subject could state the correct word or phrase two or three times to achieve recognition only to have the system misunderstand and substitute an incorrect response. The subject might have had to continue to say the correct answer two or three more times before the system recognized the correct response. In addition to being time consuming, the effort to achieve recognition was frustrating and tiring. Although the speaker model was changed to accommodate the voices of certain subjects, the same common block of words and phrases could not achieve recognition by both male and female, verifying that it was the speech system and not the speaker. An estimation of the speech system accuracy based on the total number of utterances designed (587) was approximately 80%. This meant that 20%, or an average of 117 of the utterances, failed to achieve recognition and could have been said three times before the system defaulted to the mouse. This figure is higher than the 13% error rate cited in Table 2 for the same type of system (Speaker Independent Continuous Speech Recognition) during the early 90s, although the speech system vocabulary available was larger (50,000 versus 20,000), and the application was different (critical phrases versus dictation). Reviewing the utterances that failed constantly revealed the following list:

"A missile"

"Battle damage assessment complete"

"Brevity code"

"Break left, break right"

"Bright"

"Communicates the priority"

"Crewmembers are cleared"

"Flak vest "

"Hostile aircraft"

"In-flight threat suport duties"

"In time for the crew to take action"

"Observers cleared"

"Or from the ground"

"Reconfigure an aircraft"

"Report any damages observed"

"Stow any survival equipment"

"Survival equipment - as required"

"Survival equipment secured"

"The aircraft commander assigns scanning positions"

"The pilot initiates the checklist"

"To enter a threatening environment"

"Turn using 60 degrees"

"Vertical or horizontal"

"Where you see it"

"Will don helmets"

These common failure utterances equal 84, or 14%, of the lesson total of 587, which is more in line with the error rate of 13% cited in Table 2. While the failure of these utterances can probably be attributed to an algorithm failure of the speech system, it is likely that the remaining 6% utterance failures were due to variations in the voices of the subjects and the possibility of some subjects electing to "short-cut" the design.

Several subjects had to be coached to speak louder but some seemed to have an inherently soft voice that did not allow them to extend their range of volume or pitch. To short-cut the design, a subject having difficulty with recognition might try in earnest on the first attempt to achieve recognition, then say the next two attempts rapidly and cause the default to the mouse. The multimodality used when transferring from speech recognition to the right button of the mouse could have been distracting, because all participants habitually used the left button.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

Summary

This study sought to determine the effectiveness of a new technology called speech recognition as a user interface for Computer-based Training (CBT). A review of literature on computer interfaces and speech recognition indicated that there have been many technological improvements in both areas. Nine classifications of interfaces for computers were reviewed, with natural language noted by Murphy (1992) as probably the most promising interface. The progress in speech recognition from the late 1970s to the early 1990s was also noted, which showed a dramatic increase in vocabulary capacity and the development of continuous speech. Both dependent and independent speech systems were discussed.

Moving from a description of speech systems, several applications of speech recognition were described, including a training application that used the technology in a battefield scenario. It was noted by Creech (1994) that the application was cost effective, but that any other value was not apparent.

Research that did provide a basis for using speech recognition as an interface included Jonassen and Hannum (1987), who noted that an overt response requires different levels of cognitive processing. These researchers also found that input devices such as a mouse and a keyboard were distracting and, in the case of the keyboard, placed disproportionate demands on the learner's motor system. Further studies were reviewed that attempted the use of speech to replace the keyboard . Some of the studies that met with mixed results were findings by Karl et al. (1993), Nye (1982), Legget and Williams (1984). and Visick et al. (1984). A consensus that speech recognition should be used only if there was direct user benefit was supported by Visick et al., Greene, Gould and Boies (1992), and Creech (1994).

Several studies in the 1980s were reviewed that stressed multimodal interfaces: Speech input was compared with typed words, single key presses, and mouse clicks. Martin (1989) found speech to be the most efficient response in this comparison. Karl et al. (1993) attempted a similar comparison for a Windows application and found a reduction of time for using speech input. Karl et al. also tried a counterbalanced design that used speech or mouse for word processing commands and a keyboard for text entry.

To determine whether or not a user interface could be measured for its effectiveness, Murphy (1992) designed a framework for quantifying certain variables that would provide this kind of measurement. Although noting that natural language was the most promising interface, his framework was based on traditional inputs such as the mouse and keyboard and was designed to measure the performance of computer entry tasks such as sending E-mail or similar processes. Murphy's framework included theories of generation and elaboration of words or phrases that were designed to affect cognitive processing of short- and long-term memory, which he related to Original Learning (OL) and Retained Learning (RL).

Fleming (1987) also supported the use or words or simple sentences to affect cognitive processing. This study then attempted to use part of Murphy's (1992) framework related to OL and the theories of generation and elaboration to determine whether or not speech recognition could be used as an effective interface for CBT. An effective interface was one that would affect OL and was demonstrated by differences in variables of achievement scores, progress check scores, and time-on-task.

An experiment was performed using a sample of the seventh-grade population at Shackleford Junior High School in Arlington, Texas. Random selection of a control group and an experimental group was made, using a chance drawing of odd and even numbers. The control group used the mouse as an interface, and the experimental group used speech recognition as the interface on a CBT academic lesson derived from an Air Force curriculum. The lesson did not include any aerodynamics and required no knowledge of flying. The instructional strategy for the lesson was tutorial, which presented declarative and procedural knowlege. The lesson included progress check questions and a multiple-choice achievement test.

The research questions asked by this study were as follows:

1. Is speech recognition a computer interface that will affect cognitive processing, thus affecting original learning?

2. Will using speech recognition as a computer interface affect the time required to complete a CBT lesson compared to using a mouse as a computer interface?

Conclusions

In response to Question 1, the results of the study did not support the studies cited by Jonassen and Hannum (1987) that a mouse as an input device was distracting and placed disproportionate demands on the learner's motor system. From observation, more demand was placed on the subjects' attempting to achieve speech recognition due to the use of critical phrases in addition to verbal commands and difficulties with the speech recognition system.

The results were supportive of the study by Karl et al. (1993), which proposed that interference in short-term memory was caused by use of a verbal response modality but did not occur when using the manual modality (mouse) was used. Issuing voice commands or critical statements may have interfered with short-term memory and Original Learning (OL). Many subjects were so engaged in achieving speech recognition that they failed to absorb the lesson content. A prime example was the critical phrase, "The aircraft commander assigns scanning positions," which usually had to be repeated two or three times to achieve recognition. A Progress Check question followed asking, "Who assigns scanning positions?" and was often answered incorrectly.

Factors contributing to the results may have been (a) the tendency of some subjects to take a "short-cut" when they recognized that the system would default to the mouse after three attempts at speech recognition and (b) the voice range of some subjects, which may have been peculiar to the age group of the sample.

The use of critical phrases noted by Fleming (1987) as a means of applying the techniques of generation and elaboration proposed by Murphy (1992) did not appear to facilitate short-term memory and in-turn OL. It seems that the repetition of critical phrases would have helped the subject's OL processes, regardless of whether or not the speech recognizer worked to a high percentage, but this was not supported.

In response to Question 2, an application designed to use critical phrases will probably take more time. As demonstrated in this study, it takes longer to say a phrase of two to six words than to click a mouse button. Unless a recognizer achieves 100% recognition, some of the words or phrases have to be repeated. Adding the possibility of a percentage of substitutions by the speech recognizer also increases the potential for additional time.

A review of the findings by Martin (1989), Karl et al. (1993), and Visick et al. (1984) shows a reduction in time when using speech as an input compared to a keyboard. This would seem logical if actual typing is required. In addition, the suppression of single keys such as F1 requires extending the fingers from the base keys. Using a mouse provides quicker access than a keyboard and will probably be similar in time for commands as speech.

The findings of this study tend to support earlier observations by Visick et al. (1984) that the superiority of a voice input to a keyboard would depend on the task to be performed and that speech recognition should be applied only if there was a direct user benefit. A similar point was supported by Creech (1994), who noted that speech recognition remains a tool to accomplish the task. Greene et al. (1992) summed up this direction by stating that the user interface design includes choosing the interaction methods that are most appropriate for the application content and the intended users.

Given this line of reasoning, it appears that, unless there is a specific requirement for a speech interface for a CBT academic lesson, such as for the mobility impaired or language learning, there would be no added value to the application. If used, this interface requires the following cost considerations: (a) increase in development time, (b) additional hardware, and (c) variation in classroom construction.

The increase in development time stems from the requirement to overlay the speech recognition application and to develop a syntax after a lesson has been constructed. The design of the lesson must also give special consideration to speech recognition, with the vocabulary and structure of the language to be used in addition to special instructions and default options. A speech recognition card and headset must be installed in each computer, and the classroom must be set up similar to a language learning laboratory, with sound-absorbing partitions for each student.

Although this study's speech recognition error rate was higher than the average cited for the early 1990s in Table 2, this experiment differed from the application in the citation, and the technology is still in the development state. Two of three other systems observed in demonstrations in 1994 failed to achieve 80% recognition accuracy when independent continuous speech was applied. While the results of this study have to be stated with caution, they can be categorized as indications from the state of the art, with strong inferences for interference with short-term memory, thus affecting cognitive processes for Original Learning.

Recommendations for Further Study

Further comparative studies are recommended, using similar hypotheses with a sample from an adult population from the general public or an adult population of the mobility impaired. A sample from either of these populations may have different results and show an effect on Original Learning (OL). Further studies should also include the factors for measuring the effect on Retained Learning (RL).

A key element in a follow-up study would be a recognizer that can achieve at least 95% recognition so that the interaction is not frustrating or stressful. If critical phrases are used, they should be short and simple. The words themselves should not have more than three syllables, if possible. Content areas for the studies should include general academic or technical subjects as well as language training but should focus on the learning effect. An instructional strategy that includes previous learning on a related subject may also show different results. Data collection should include the capability to count the number of substitutions by the recognizer, invalid speech responses, time to complete the lesson, time to complete the test, and correct and incorrect selections of test questions.

A specific study recommended is English as a second language, using speech recognition as the interface and speech synthesis as an evaluator response. This combination of speech recognition and speech synthesis would also provide potential interface comparisons for artifical intelligence applications. A continuous speech speaker-dependent system may provide the best results because these systems adapt to the speaker's voice and could provide an increase in accuracy.

Should a trend develop in any academic, technical, or language area that demonstrates that learning is increased through the facilitation of the short-term memory by speech recognition, then the additional cost for hardware, design, and development could be worth the investment. Prices for speech recognition hardware have decreased by 50% or more (Thyfault, 1994), making it more of an option for use in the educational environment. The fact that the additonal costs are normally a one-time expense may provide further justification for the acquisition of speech recognition technology to assist in a learning environment.

Although there is no full agreement with Murphy's (1992, p. 31) statement that " we are still a long way from being able to provide effective full natural language interfaces with computers," there is still room for much more investigation in this area.

APPENDIX A

SAMPLE OF LESSON TEST QUESTIONS

Which of these terms indicates the highest priority of a threat on a tactical mission?

a. paralleling
b. engaging
c. turning
d. rolling in

Answer: b

When you take your assigned position to scan for threats on a tactical VFR mission, which threat calling term is immediately determined?

a. type of threat
b. vertical position
c. right/left
d. clock code

Answer: c

Which of the following is an extra effort required to prepare the aircraft to enter the threat environment on a tactical mission?

a. Open first-aid kits for quick access.
b. Cover windows of doors or hatches to reduce visible light.
c. Prepare cargo for jettisoning.
d. Apply additional restraints to equipment at risk of coming loose.

Answer: d

Who is required to wear a helmet in the cargo compartment while in a threat environment on a tactical mission?

a. loadmaster and scanner only
b. all personnel required to be mobile
c. any crewmember
d. all personnel
Answer: d

How wide a segment around the aircraft does a single scan for threats cover?

a. one clock segment
b. two clock segments
c. 20 degrees
d. 45 degrees

Answer: b

Which of the following are accomplished with the Combat Exit Checklist?

a. stow survival equipment, set internal lights to normal
b. secure loose equipment, close troop or cargo doors
c. prepare survival equipment, report battle damage
d. secure loose equipment, set internal lights to dim or red

Answer: a

Before what phase of a tactical mission should the Combat Entry Checklist be completed?

a. entry into the threat environment
b. landing at an airfield in a hostile environment
c. coming in range of SAM batteries
d. rendezvous with an AWACS

Answer: a

To reduce unnecessary electronic emissions prior to entry into a combat area, which of the following items may be turned off?

a. airspeed indicator
b. fuel indicator
c. radar
d. interphone

Answer: c

Which term conveys the most urgent action for threat avoidance?

a. turn
b. break
c. pitch
d. scan

Answer: b

If the troop doors are open in a combat area during a night mission, the inside lights should be:

a. red
b. green
c. bright
d. dim

Answer: a

APPENDIX B

APPLICATION CODE

```
#include "stdafx.h"
#include "c141.h"
BOOL gbDecodeExists;
COLORREF dwOldSysBtnFaceColor;
COLORREF dwOldSysBtnHighlightColor;
COLORREF dwOldSysBtnShadowColor;
^COLORREF dwOldSysBtnTextColor;
HGLOBAL ghSPOTBuffer;
HHOOK ghHookFunc;
HWND ghParentWnd;
WORD gwInvalidResponses;
WORD gwSpeechID;
WORD gwSpeechTry;

void CenterAndPlaceDialog(HWND hWnd, BOOL bVCenter)
{
    RECT rParentRect;
    RECT rRect;
    if (!hWnd)
        return;

    GetParent(hWnd) ? GetClientRect(GetParent(hWnd), &rParentRect) :
        GetClientRect(GetDesktopWindow(), &rParentRect);
    GetWindowRect(hWnd, &rRect);
    rRect.right += rRect.left = (rParentRect.right - (rRect.right -
        rRect.left)) / 2;
    // If bVCenter is TRUE, center it vertically. Otherwise, place it on
    // the bottom of the window.
    if (bVCenter)
        rRect.bottom += rRect.top = (rParentRect.bottom - (rRect.bottom -
                rRect.top)) / 2;
    else
        rRect.bottom += rRect.top = rParentRect.bottom - (rRect.bottom -
                rRect.top);
    MoveWindow(hWnd, rRect.left, rRect.top, rRect.right - rRect.left,

                            rRect.bottom - rRect.top, TRUE);
}
int ChangeModel(void)
{
    return SpotChooseModel(gwSpeechID, ghParentWnd, NULL, NULL,
        "Change Model");
}
HWND CreateLoadingDialog(HWND hParentWnd)
{
    HWND       hLoadingDialog;
    if (!hParentWnd)
        return 0;
    hLoadingDialog = CreateDialog(GetModuleHandle(DLL_NAME),
```

```
        MAKEINTRESOURCE(IDD_LOADING), hParentWnd ? hParentWnd :
        GetDesktopWindow(), (DLGPROC)LoadingDialogProc);
    FlushMessageQueue(hLoadingDialog);
    return hLoadingDialog;
}
HBRUSH DlgCtlColor(HDC hDC, HWND hCtrl, int nCtrlType)
{
    if (!hDC || !hCtrl)
        return NULL;
    SetBkColor(hDC, GetSysColor(COLOR_BTNFACE));
    SetTextColor(hDC, GetSysColor(COLOR_BTNTEXT));
    return GetStockObject(LTGRAY_BRUSH);
}
void DlgPaint(HWND hDlg)
{
    HANDLE hDC;
    HPEN hHighlightPen, hShadowPen, hOldPen;
    RECT rRect;
    (!hDlg)
        return;
    // Put a shadowed edge around the dialog box.
    GetClientRect(hDlg, &rRect);
    hDC = GetDC(hDlg);
    // Draw the shadow.
    hShadowPen = CreatePen(PS_SOLID, 1,
        GetSysColor(COLOR_BTNSHADOW));
    hOldPen = SelectObject(hDC, hShadowPen);
    MoveTo(hDC, rRect.left, rRect.bottom);
    LineTo(hDC, rRect.left, rRect.top);
    LineTo(hDC, rRect.right, rRect.top);
    MoveTo(hDC, rRect.left + 1, rRect.bottom - 1);
    LineTo(hDC, rRect.left + 1, rRect.top + 1);
    LineTo(hDC, rRect.right - 1, rRect.top + 1);
    MoveTo(hDC, rRect.left + 2, rRect.bottom - 2);
    LineTo(hDC, rRect.left + 2, rRect.top + 2);
    LineTo(hDC, rRect.right - 2, rRect.top + 2);
    // Draw the highlight.
    hHighlightPen = CreatePen(PS_SOLID, 1,
        GetSysColor(COLOR_BTNHIGHLIGHT));
    SelectObject(hDC, hHighlightPen);
    MoveTo(hDC, rRect.left + 1, rRect.bottom - 1);
    LineTo(hDC, rRect.right - 1, rRect.bottom - 1);
    LineTo(hDC, rRect.right - 1, rRect.top + 1);
    MoveTo(hDC, rRect.left + 2, rRect.bottom - 2);
    LineTo(hDC, rRect.right - 2, rRect.bottom - 2);
    LineTo(hDC, rRect.right - 2, rRect.top + 2);
    MoveTo(hDC, rRect.left + 3, rRect.bottom - 3);
    LineTo(hDC, rRect.right - 3, rRect.bottom - 3);
    LineTo(hDC, rRect.right - 3, rRect.top + 3);
```

```
    // Do cleanup.
    SelectObject(hDC, hOldPen);
    DeleteObject(hHighlightPen);
    DeleteObject(hShadowPen);
    ReleaseDC(hDlg, hDC);

    // Validate the entire rectangle since we drew it.
    // However, we must invalidate the text control.
    ValidateRect(hDlg, NULL);
    InvalidateRect(GetDlgItem(hDlg, IDC_LOADING), NULL, TRUE);
}
WORD FAR PASCAL _export FindWindowByTitle(LPCSTR lpszWindowName)
{
    return FindWindow(NULL, lpszWindowName);
}
void FlushMessageQueue(HWND hWnd)
{
    MSG Msg;

    if (!hWnd)
        return;
    while(PeekMessage(&Msg, hWnd, NULL, NULL, PM_REMOVE))
    {
        TranslateMessage(&Msg);
        DispatchMessage(&Msg);
    }
}
WORD FAR PASCAL _export GetInvalidResponses(void)
{
    return gwInvalidResponses;
}
WORD FAR PASCAL _export GetSpeechTry(void)
{
    return gwSpeechTry;
}
LRESULT CALLBACK _export HookProc(int nCode, WPARAM wParam,
LPARAM lParam)
{
    if ((nCode >= 0) && ghParentWnd)
    {
        switch (((MSG *)lParam)->message)
        {
                case WM_CLOSE:
                    {
                                                COLORREF
    adwRGBValues[4];
                int                             anDisplayElements[4];
```

```
// Reinstate the old colors.
anDisplayElements[0] = COLOR_BTNFACE;
anDisplayElements[1] =
COLOR_BTNHIGHLIGHT;
anDisplayElements[2] =
COLOR_BTNSHADOW;
anDisplayElements[3] =
COLOR_BTNTEXT;
adwRGBValues[0] =
dwOldSysBtnFaceColor;
adwRGBValues[1] =
dwOldSysBtnHighlightColor;
adwRGBValues[2] =
dwOldSysBtnShadowColor;
adwRGBValues[3] =
dwOldSysBtnTextColor;
SetSysColors(4,
anDisplayElements, adwRGBValues);

// Free up the SPOT buffer.
GlobalFree(ghSPOTBuffer);

// Uninitialize the speech
recognition.
UninitializePE400(0);
}
break;

case WM_HAVEDECODE:
nCode = -1;
if (ghSPOTBuffer)
{
static char
lpszOldBuffer[MAX_STRING];
LPSTR lpszSPOTBuffer =
GlobalLock(ghSPOTBuffer);
lstrcpy(lpszOldBuffer,
lpszSPOTBuffer);
lpszSPOTBuffer[0] = '\0';
SpotGetText(gwSpeechID,
SpotFindTag(gwSpeechID, "S", 0, 0),
lpszSPOTBuffer, MAX_STRING);
// Because of an unforeseen
problem, we are getting two messages
// for each decode. This doesn't
affect the program except in
// the case where a dialog is
displayed. At that point, though
```

```
                                        // we may not always want this, if
we get a successful decode,
                                        // fail if the next one is the same
text and that text is for
                                        // model or gain.
                                        if (!lstrcmp(lpszOldBuffer,
lpszSPOTBuffer) &&
    (strstr(lpszSPOTBuffer, "Model") || strstr(lpszSPOTBuffer,
    "Gain")))
                                                            break;
                                        gbDecodeExists = TRUE;
                                        // Change model or set gain?
                                        if (strstr(lpszSPOTBuffer,
"Model"))
                                        {
    ChangeModel();
    gbDecodeExists = FALSE;
                                        }
                                        if (strstr(lpszSPOTBuffer, "Gain"))
                                        {
    SetGain();
    gbDecodeExists = FALSE;
                                        }
                                        GlobalUnlock(ghSPOTBuffer);
                        }
                        break;

                case WM_NODECODE:
                        nCode = -1;
                        {
                                        LPSTR lpszSPOTBuffer;
                                        lpszSPOTBuffer =
GlobalLock(ghSPOTBuffer);
                                        lpszSPOTBuffer[0] = '\0';
                                        GlobalUnlock(ghSPOTBuffer);
                        }
                        NoDecodeDialog(ghParentWnd);
                        gbDecodeExists = FALSE;
                        break;
```

```
        }
    }

    return CallNextHookEx(ghHookFunc, nCode, wParam, lParam);
}

WORD FAR PASCAL _export InitializePE400(HWND hParentWnd)
{
    COLORREF        adwRGBValues[4];
    HGLOBAL  hPHDFileName;
    HGLOBAL  hSTDFileName;
    HWND            hLoadingDialog;
    int             anDisplayElements[4];
    LPSTR           lpszPHDFileName;
    LPSTR           lpszSTDFileName;
    WORD            wSpeechID;

    if (!hParentWnd || gwSpeechID)
        return 0;

    // Set up our new colors for this application.
    dwOldSysBtnFaceColor = GetSysColor(COLOR_BTNFACE);
    dwOldSysBtnHighlightColor = GetSysColor(COLOR_BTNHIGHLIGHT);
    dwOldSysBtnShadowColor = GetSysColor(COLOR_BTNSHADOW);
    dwOldSysBtnTextColor = GetSysColor(COLOR_BTNTEXT);
    anDisplayElements[0] = COLOR_BTNFACE;
    anDisplayElements[1] = COLOR_BTNHIGHLIGHT;
    anDisplayElements[2] = COLOR_BTNSHADOW;
    anDisplayElements[3] = COLOR_BTNTEXT;
    adwRGBValues[0] = RGB( 192, 192, 192);
    adwRGBValues[1] = RGB( 255, 255, 255);
    adwRGBValues[2] = RGB( 128, 128, 128);
    adwRGBValues[3] = RGB( 0, 0, 0);
    SetSysColors(4, anDisplayElements, adwRGBValues);

    // Put up a dialog telling us we are loading the driver.
    hLoadingDialog = CreateLoadingDialog(hParentWnd);

    // Check to make sure the PE400 is available.
    if (!SpotAvailable())
    {
        MessageBeep(MB_ICONSTOP);
        MessageBox(hParentWnd, "PE400 not available for speech "
                "recognition.\n",                "Error!", MB_ICONSTOP |
MB_OK);

        // Destroy the loading dialog.
        DestroyWindow(hLoadingDialog);
```

```
    return 0;
}

// Initialize the voice recognizer.
hPHDFileName = GlobalAlloc(MAX_STRING + 1,
    GMEM_ZEROINIT | GMEM_MOVEABLE);
lpszPHDFileName = GlobalLock(hPHDFileName);
LoadString(GetModuleHandle(DLL_NAME), IDS_PHD_FILENAME,
    lpszPHDFileName,                    MAX_STRING);
hSTDFileName = GlobalAlloc(MAX_STRING + 1,
    GMEM_ZEROINIT | GMEM_MOVEABLE);
lpszSTDFileName = GlobalLock(hSTDFileName);
LoadString(GetModuleHandle(DLL_NAME), IDS_STD_FILENAME,
    lpszSTDFileName,                    MAX_STRING);
wSpeechID = (WORD)SpotMakeContext("", lpszPHDFileName,
    lpszSTDFileName);
if (!wSpeechID)
{
    MessageBeep(MB_ICONSTOP);
    MessageBox(hParentWnd, "SpotMakeContext failed.\n",
            "Error!", MB_ICONSTOP | MB_OK);

    // Destroy the loading dialog.
    DestroyWindow(hLoadingDialog);

    return 0;
}

if (!SpotSetRecogParam(wSpeechID, 4) ||
    !SpotSetRejectParams(wSpeechID, 15, 5, REJECT_PARAM1, 0,
            REJECT_PARAM2) ||
    !SpotSetMessage(wSpeechID, hParentWnd, hParentWnd,
    WM_HAVEDECODE, wSpeechID, 0L, hParentWnd, WM_NODECODE,
    wSpeechID, 0L))
{
    MessageBeep(MB_ICONSTOP);
    MessageBox(hParentWnd, "Could not initialize voice "
            "recognition system.\n", "Error!", MB_ICONSTOP | MB_OK);

    // Destroy the loading dialog.
    DestroyWindow(hLoadingDialog);

    return 0;
}

// Allocate memory for the SPOT buffer.
ghSPOTBuffer = GlobalAlloc(MAX_STRING + 1,
```

```
        GMEM_ZEROINIT | GMEM_MOVEABLE);

    // Set up our much needed globals.
    ghParentWnd = hParentWnd;
    gwSpeechID = wSpeechID;

    // Set up the hook procedure to catch WM_HAVEDECODE
    // and WM_NODECODE.
    SetHookProc();

    // Destroy the loading dialog.
    DestroyWindow(hLoadingDialog);

    return wSpeechID;
}

LRESULT CALLBACK LoadingDialogProc(HWND hDlg, UINT uMsg, WPARAM
wParam, LPARAM
    lParam)
{
    switch (uMsg)
    {
        case WM_CLOSE:
                SetCursor(LoadCursor(NULL, IDC_ARROW));
                break;

        case WM_CTLCOLOR:
                return DlgCtlColor((HDC)wParam, (HWND)LOWORD(lParam),
                        (int)HIWORD(lParam));

        case WM_INITDIALOG:
                CenterAndPlaceDialog(hDlg, FALSE);
                break;

        case WM_MOUSEMOVE:
                SetCursor(LoadCursor(NULL, IDC_WAIT));
                break;

        case WM_PAINT:
                DlgPaint(hDlg);
                break;

        default:
                return FALSE;
    }

    return TRUE;
}
```

```
void NoDecodeDialog(HWND hParentWnd)
{
    if (!hParentWnd || FindWindow(NULL, "Recognition Failed"))
        return;

    gwSpeechTry++;
    gwInvalidResponses++;
    DialogBox(GetModuleHandle(DLL_NAME),
MAKEINTRESOURCE(IDD_NODECODE),
        hParentWnd ? hParentWnd : GetDesktopWindow(),
        (DLGPROC)NoDecodeDialogProc);
}

LRESULT CALLBACK NoDecodeDialogProc(HWND hDlg, UINT uMsg,
WPARAM wParam, LPARAM
    lParam)
{
    switch (uMsg)
    {
        case WM_COMMAND:
                if (wParam == IDOK)
                        EndDialog(hDlg, IDOK);
                break;

        case WM_CTLCOLOR:
                if (HIWORD(lParam) != CTLCOLOR_BTN)
                {
                        HBRUSH hResult;

                        hResult = DlgCtlColor((HDC)wParam,
(HWND)LOWORD(lParam),
                                                (int)HIWORD(lParam));
                        InvalidateRect(GetDlgItem(hDlg, IDOK), NULL, TRUE);

                        return hResult;
                }

                return GetStockObject(LTGRAY_BRUSH);
                break;

        case WM_INITDIALOG:
                CenterAndPlaceDialog(hDlg, TRUE);
                break;

        case WM_PAINT:
                DlgPaint(hDlg);
                break;

        default:
```

```
                return FALSE;
    }

    return TRUE;
}

int SetGain(void)
{
    return SpotSetGain(gwSpeechID, ghParentWnd, NULL, NULL,
        "Set Gain", NULL, NULL);
}

WORD SetHookProc(void)
{
    // Don't do anything if we have already initialized it.
    if (ghHookFunc)
        return 1;

    ghHookFunc = SetWindowsHookEx(WH_GETMESSAGE,
(HOOKPROC)HookProc,
        GetModuleHandle(DLL_NAME), GetWindowTask(ghParentWnd));
    if (!ghHookFunc)
        return 0;

    return 1;
}

WORD FAR PASCAL _export SetInvalidResponses(WORD wInvalidResponses)
{
    return gwInvalidResponses = wInvalidResponses;
}

WORD FAR PASCAL _export SetSpeechTry(WORD wSpeechTry)
{
    return gwSpeechTry = wSpeechTry;
}

WORD FAR PASCAL _export UninitializePE400(WORD wDummy)
{
    int nResult;

    if (!gwSpeechID)
        return 0;

    nResult = SpotClearMessage(gwSpeechID);
    nResult |= SpotDeleteContext(gwSpeechID);
    gwSpeechID = 0;
    return (WORD)nResult;
```

```
}

WORD UnSetHookProc(void)
{
   // Don't do anything if we have already uninitialized it.
   if (!ghHookFunc)
        return 1;

   return UnhookWindowsHookEx(ghHookFunc);
}

WORD FAR PASCAL _export ValidResponse(WORD wResponseID, WORD
wMoreToCome)
{
   WORD wResult = FALSE;
   LPSTR lpszSPOTBuffer;
   static int nCorrectResponse = 0;

   if (!ghSPOTBuffer || !wResponseID || !gbDecodeExists)
        return 0;

   lpszSPOTBuffer = GlobalLock(ghSPOTBuffer);

   switch (wResponseID)
   {
        case RESPONSE_A:
                if (strstr(lpszSPOTBuffer, "A"))
                        wResult = TRUE;
                break;
        case RESPONSE_ACKNOWLEDGE:
                if (strstr(lpszSPOTBuffer, "Acknowledged"))
                        wResult = TRUE;
                break;
        case RESPONSE_AIRCRAFT:
                if (strstr(lpszSPOTBuffer, "Aircraft"))
                        wResult = TRUE;
                break;
        case RESPONSE_AIRSPEED:
                if (strstr(lpszSPOTBuffer, "Airspeed"))
                        wResult = TRUE;
                break;
        case RESPONSE_ALLPERSONNEL:
                if (strstr(lpszSPOTBuffer, "All Personnel"))
                        wResult = TRUE;
                break;
        case RESPONSE_APPLY:
                if (strstr(lpszSPOTBuffer, "Apply"))
                        wResult = TRUE;
                break;
```

```
case RESPONSE_B:
        if (strstr(lpszSPOTBuffer, "B"))
                wResult = TRUE;
        break;
case RESPONSE_BANDIT:
        if (strstr(lpszSPOTBuffer, "Bandit"))
                wResult = TRUE;
        break;
case RESPONSE_BATTLE:
        if (strstr(lpszSPOTBuffer, "Battle"))
                wResult = TRUE;
        break;
case RESPONSE_BREAK:
        if (strstr(lpszSPOTBuffer, "Break"))
                wResult = TRUE;
        break;
case RESPONSE_BRIGHT:
        if (strstr(lpszSPOTBuffer, "Bright"))
                wResult = TRUE;
        break;
case RESPONSE_C:
        if (strstr(lpszSPOTBuffer, "C"))
                wResult = TRUE;
        break;
case RESPONSE_CHEMICAL:
        if (strstr(lpszSPOTBuffer, "Chemical"))
                wResult = TRUE;
        break;
case RESPONSE_CLOCKCODE:
        if (strstr(lpszSPOTBuffer, "Clock"))
                wResult = TRUE;
        break;
case RESPONSE_COMBATENTRY:
        if (strstr(lpszSPOTBuffer, "Entry"))
                wResult = TRUE;
        break;
case RESPONSE_COMBATEXIT:
        if (strstr(lpszSPOTBuffer, "Exit"))
                wResult = TRUE;
        break;
case RESPONSE_COMING:
        if (strstr(lpszSPOTBuffer, "Coming"))
                wResult = TRUE;
        break;
case RESPONSE_COMMANDER:
        if (strstr(lpszSPOTBuffer, "Commander"))
                wResult = TRUE;
        break;
case RESPONSE_COMPLETED:
```

```
        if (strstr(lpszSPOTBuffer, "Completed"))
            wResult = TRUE;
        break;
    case RESPONSE_COPILOT:
        if (strstr(lpszSPOTBuffer, "Copilot"))
            wResult = TRUE;
        break;
    case RESPONSE_COVER:
        if (strstr(lpszSPOTBuffer, "Cover"))
            wResult = TRUE;
        break;
    case RESPONSE_CREWMEMBER:
        if (strstr(lpszSPOTBuffer, "Crew"))
            wResult = TRUE;
        break;
    case RESPONSE_D:
        if (strstr(lpszSPOTBuffer, "D"))
            wResult = TRUE;
        break;
    case RESPONSE_DIM:
        if (strstr(lpszSPOTBuffer, "Dim"))
            wResult = TRUE;
        break;
    case RESPONSE_EIGHT:
        if (strstr(lpszSPOTBuffer, "Eight"))
            wResult = TRUE;
        break;
    case RESPONSE_EIGHTEEN:
        if (strstr(lpszSPOTBuffer, "Eighteen"))
            wResult = TRUE;
        break;
    case RESPONSE_EIGHTHIGH:
        if ((strstr(lpszSPOTBuffer, "Break"))
            && (strstr(lpszSPOTBuffer, "Left")))
            wResult = TRUE;
        break;
    case RESPONSE_ELEVEN:
        if (strstr(lpszSPOTBuffer, "Eleven"))
            wResult = TRUE;
        break;
    case RESPONSE_EMERGENCY:
        if (strstr(lpszSPOTBuffer, "Emergency"))
            wResult = TRUE;
        break;
    case RESPONSE_ENGAGING:
        if (strstr(lpszSPOTBuffer, "Engaging"))
            wResult = TRUE;
        break;
    case RESPONSE_ENTRY:
```

```
		if (strstr(lpszSPOTBuffer, "Entry"))
			wResult = TRUE;
		break;
	case RESPONSE_EQUIPMENT:
		if (strstr(lpszSPOTBuffer, "Equipment"))
			wResult = TRUE;
		break;
	case RESPONSE_FIRSTAID:
		if (strstr(lpszSPOTBuffer, "First"))
			wResult = TRUE;
		break;
	case RESPONSE_FIVE:
		if (strstr(lpszSPOTBuffer, "Five"))
			wResult = TRUE;
		break;
	case RESPONSE_FIVEHIGHTWO:
		if ((strstr(lpszSPOTBuffer, "Five"))
			&& (strstr(lpszSPOTBuffer, "High")))
			wResult = TRUE;
		break;
	case RESPONSE_FIVEHIGHTHREE:
		if ((strstr(lpszSPOTBuffer, "Right"))
			&& (strstr(lpszSPOTBuffer, "Bandit")))
			wResult = TRUE;
		break;
	case RESPONSE_FIVELOW:
		if ((strstr(lpszSPOTBuffer, "Five"))
			&& (strstr(lpszSPOTBuffer, "Low")))
			wResult = TRUE;
		break;
	case RESPONSE_FOUR:
		if (strstr(lpszSPOTBuffer, "Four"))
			wResult = TRUE;
		break;
	case RESPONSE_FOURTYFIVE:
		if (strstr(lpszSPOTBuffer, "Fourty"))
			wResult = TRUE;
		break;
	case RESPONSE_FUEL:
		if (strstr(lpszSPOTBuffer, "Fuel"))
			wResult = TRUE;
		break;
	case RESPONSE_GOINGAWAY:
		if (strstr(lpszSPOTBuffer, "Going"))
			wResult = TRUE;
		break;
	case RESPONSE_GREEN:
		if (strstr(lpszSPOTBuffer, "Green"))
			wResult = TRUE;
```

```
        break;
case RESPONSE_GUN:
        if (strstr(lpszSPOTBuffer, "Gun"))
                wResult = TRUE;
        break;
case RESPONSE_HELMET:
        if (strstr(lpszSPOTBuffer, "Helmet"))
                wResult = TRUE;
        break;
case RESPONSE_HIGH:
        if (strstr(lpszSPOTBuffer, "High"))
                wResult = TRUE;
        break;
case RESPONSE_INTERPHONE:
        if (strstr(lpszSPOTBuffer, "Interphone"))
                wResult = TRUE;
        break;
case RESPONSE_LANDING:
        if (strstr(lpszSPOTBuffer, "Landing"))
                wResult = TRUE;
        break;
case RESPONSE_INFLIGHT:
        if (strstr(lpszSPOTBuffer, "Threat"))
                wResult = TRUE;
        break;
case RESPONSE_INTERNAL:
        if (strstr(lpszSPOTBuffer, "Internal"))
                wResult = TRUE;
        break;
case RESPONSE_INTRODUCTION:
        if (strstr(lpszSPOTBuffer, "Introduction"))
                wResult = TRUE;
        break;
case RESPONSE_LEAVING:
        if (strstr(lpszSPOTBuffer, "Leaving"))
                wResult = TRUE;
        break;
case RESPONSE_LEFT:
        if (strstr(lpszSPOTBuffer, "Left"))
                wResult = TRUE;
        break;
case RESPONSE_LESSONTEST:
        if (strstr(lpszSPOTBuffer, "Lesson"))
                wResult = TRUE;
        break;
case RESPONSE_LEVEL:
        if (strstr(lpszSPOTBuffer, "Level"))
                wResult = TRUE;
        break;
```

```
case RESPONSE_LOADMASTER:
        if (strstr(lpszSPOTBuffer, "Load"))
                wResult = TRUE;
        break;
case RESPONSE_LOOSEITEMS:
        if (strstr(lpszSPOTBuffer, "Loose"))
                wResult = TRUE;
        break;
case RESPONSE_LOW:
        if (strstr(lpszSPOTBuffer, "Low"))
                wResult = TRUE;
        break;
case RESPONSE_MISSILE:
        if (strstr(lpszSPOTBuffer, "Missile"))
                wResult = TRUE;
        break;
case RESPONSE_MISSILELEFT:
        if ((strstr(lpszSPOTBuffer, "Missile"))
                && (strstr(lpszSPOTBuffer, "Left")))
                wResult = TRUE;
        break;
case RESPONSE_MOBILEPERSONNEL:
        if (strstr(lpszSPOTBuffer, "Mobile"))
                wResult = TRUE;
        break;
case RESPONSE_NAVIGATOR:
        if (strstr(lpszSPOTBuffer, "Navigator"))
                wResult = TRUE;
        break;
case RESPONSE_NEXTSCREEN:
        if (strstr(lpszSPOTBuffer, "Next"))
                wResult = TRUE;
        break;
case RESPONSE_NINE:
        if (strstr(lpszSPOTBuffer, "Nine"))
                wResult = TRUE;
        break;
case RESPONSE_NINEHIGH:
        if ((strstr(lpszSPOTBuffer, "Nine"))
                && (strstr(lpszSPOTBuffer, "High")))
                wResult = TRUE;
        break;
case RESPONSE_NINELOWFIVE:
        if ((strstr(lpszSPOTBuffer, "Nine"))
                && (strstr(lpszSPOTBuffer, "Low")))
                wResult = TRUE;
        break;
case RESPONSE_NINELOWTWO:
        if ((strstr(lpszSPOTBuffer, "Break"))
```

```
                && (strstr(lpszSPOTBuffer, "Left")))
                wResult = TRUE;
            break;
    case RESPONSE_NO:
            if (strstr(lpszSPOTBuffer, "No"))
                wResult = TRUE;
            break;
    case RESPONSE_OBSERVERS:
            if (strstr(lpszSPOTBuffer, "Observers"))
                wResult = TRUE;
            break;
    case RESPONSE_OFFLOADING:
            if (strstr(lpszSPOTBuffer, "Offloading"))
                wResult = TRUE;
            break;
    case RESPONSE_ONE:
            if (strstr(lpszSPOTBuffer, "One"))
                wResult = TRUE;
            break;
    case RESPONSE_PARACHUTE:
            if (strstr(lpszSPOTBuffer, "Parachute"))
                wResult = TRUE;
            break;
    case RESPONSE_PARALLELING:
            if (strstr(lpszSPOTBuffer, "Paralleling"))
                wResult = TRUE;
            break;
    case RESPONSE_PILOT:
            if (strstr(lpszSPOTBuffer, "Pilot"))
                wResult = TRUE;
            break;
    case RESPONSE_PITCH:
            if (strstr(lpszSPOTBuffer, "Pitch"))
                wResult = TRUE;
            break;
    case RESPONSE_PREPARE:
            if (strstr(lpszSPOTBuffer, "Prepare"))
                wResult = TRUE;
            break;
    case RESPONSE_PROCEED:
            if (strstr(lpszSPOTBuffer, "Proceed"))
                wResult = TRUE;
            break;
    case RESPONSE_RADAR:
            if (strstr(lpszSPOTBuffer, "Radar"))
                wResult = TRUE;
            break;
    case RESPONSE_RED:
            if (strstr(lpszSPOTBuffer, "Red"))
```

```
                wResult = TRUE;
            break;
        case RESPONSE_RENDEZVOUS:
            if (strstr(lpszSPOTBuffer, "Rendezvous"))
                wResult = TRUE;
            break;
        case RESPONSE_RESULTS:
            if (strstr(lpszSPOTBuffer, "Results"))
                wResult = TRUE;
            break;
        case RESPONSE_RETURNING:
            if (strstr(lpszSPOTBuffer, "Returning"))
                wResult = TRUE;
            break;
        case RESPONSE_RIGHT:
            if (strstr(lpszSPOTBuffer, "Right"))
                wResult = TRUE;
            break;
        case RESPONSE_RIGHTLEFT:
            if (strstr(lpszSPOTBuffer, "Right"))
                wResult = TRUE;
            break;
        case RESPONSE_ROLLINGIN:
            if (strstr(lpszSPOTBuffer, "Rolling"))
                wResult = TRUE;
            break;
        case RESPONSE_SCAN:
            if (strstr(lpszSPOTBuffer, "Scan"))
                wResult = TRUE;
            break;
        case RESPONSE_SECURECLOSE:
            if ((strstr(lpszSPOTBuffer, "Secure"))
                && (strstr(lpszSPOTBuffer, "Close")))
                wResult = TRUE;
            break;
        case RESPONSE_SECURESET:
            if ((strstr(lpszSPOTBuffer, "Secure"))
                && (strstr(lpszSPOTBuffer, "Set")))
                wResult = TRUE;
            break;
        case RESPONSE_SELECT:
            if (strstr(lpszSPOTBuffer, "Select"))
                wResult = TRUE;
            break;
        case RESPONSE_SEVEN:
            if (strstr(lpszSPOTBuffer, "Seven"))
                wResult = TRUE;
            break;
        case RESPONSE_SEVENLEVEL:
```

```
        if ((strstr(lpszSPOTBuffer, "Seven"))
                && (strstr(lpszSPOTBuffer, "Level")))
                wResult = TRUE;
        break;
case RESPONSE_SEVENLOW:
        if ((strstr(lpszSPOTBuffer, "Seven"))
                && (strstr(lpszSPOTBuffer, "Low")))
                wResult = TRUE;
        break;
case RESPONSE_SIX:
        if (strstr(lpszSPOTBuffer, "Six"))
                wResult = TRUE;
        break;
case RESPONSE_STOW:
        if (strstr(lpszSPOTBuffer, "Stow"))
                wResult = TRUE;
        break;
case RESPONSE_SUMMARY:
        if (strstr(lpszSPOTBuffer, "Summary"))
                wResult = TRUE;
        break;
case RESPONSE_SURVIVAL:
        if (strstr(lpszSPOTBuffer, "Survival"))
                wResult = TRUE;
        break;
case RESPONSE_TAKEOFF:
        if (strstr(lpszSPOTBuffer, "Takeoff"))
                wResult = TRUE;
        break;
case RESPONSE_TEN:
        if (strstr(lpszSPOTBuffer, "Ten"))
                wResult = TRUE;
        break;
case RESPONSE_TENLEVEL:
        if ((strstr(lpszSPOTBuffer, "Bandit"))
                && (strstr(lpszSPOTBuffer, "Left")))
                wResult = TRUE;
        break;
case RESPONSE_TENLOW:
        if ((strstr(lpszSPOTBuffer, "Break"))
                && (strstr(lpszSPOTBuffer, "Left")))
                wResult = TRUE;
        break;
case RESPONSE_THREE:
        if (strstr(lpszSPOTBuffer, "Three"))
                wResult = TRUE;
        break;
case RESPONSE_THREEHIGH:
        if ((strstr(lpszSPOTBuffer, "Three"))
```

```
                && (strstr(lpszSPOTBuffer, "High")))
                wResult = TRUE;
        break;
case RESPONSE_THREELOW:
        if ((strstr(lpszSPOTBuffer, "Three"))
                && (strstr(lpszSPOTBuffer, "Low")))
                wResult = TRUE;
        break;
case RESPONSE_TITLESCREEN:
        if (strstr(lpszSPOTBuffer, "Title"))
                wResult = TRUE;
        break;
case RESPONSE_TURN:
        if (strstr(lpszSPOTBuffer, "Turn"))
                wResult = TRUE;
        break;
case RESPONSE_TURNING:
        if (strstr(lpszSPOTBuffer, "Turning"))
                wResult = TRUE;
        break;
case RESPONSE_TWELVE:
        if (strstr(lpszSPOTBuffer, "Twelve"))
                wResult = TRUE;
        break;
case RESPONSE_TWENTY:
        if (strstr(lpszSPOTBuffer, "Twenty"))
                wResult = TRUE;
        break;
case RESPONSE_TWO:
        if (strstr(lpszSPOTBuffer, "Two"))
                wResult = TRUE;
        break;
case RESPONSE_TYPEOFTHREAT:
        if (strstr(lpszSPOTBuffer, "Type"))
                wResult = TRUE;
        break;
case RESPONSE_VERTICALPOS:
        if (strstr(lpszSPOTBuffer, "Vertical"))
                wResult = TRUE;
        break;
case RESPONSE_WHATDOING:
        if ((strstr(lpszSPOTBuffer, "What"))
                && (strstr(lpszSPOTBuffer, "Doing")))
                wResult = TRUE;
        break;
case RESPONSE_WHATSEE:
        if ((strstr(lpszSPOTBuffer, "What"))
                && (strstr(lpszSPOTBuffer, "See")))
                wResult = TRUE;
```

```
                break;
            case RESPONSE_WHERESEE:
                if (strstr(lpszSPOTBuffer, "Where"))
                    wResult = TRUE;
                break;
            case RESPONSE_YES:
                if (strstr(lpszSPOTBuffer, "Yes"))
                    wResult = TRUE;
                break;
    }

    GlobalUnlock(ghSPOTBuffer);

    if (wResult && wMoreToCome)
        nCorrectResponse = 1;

    if (!wResult && !wMoreToCome && !nCorrectResponse)
        PostMessage(ghParentWnd, WM_NODECODE, gwSpeechID, NULL);

    if (!wMoreToCome)
    {
        gbDecodeExists = FALSE;
        nCorrectResponse = 0;
    }

    return wResult;
}

int FAR PASCAL _WEP(int nExitType)
{
    UnSetHookProc();
    return 1;
```

APPENDIX C

SYNTAX SOURCE FILE

Syntax Source File

```
COVER_WINDOWS -> Cover Windows Of Doors And Hatches To Reduce Visible Light

CREWMEMBER -> Any Crewmember

DEGREES -> NUMBERS+ Degrees

DIM -> Put the BRIGHT DIM Switch On Dim

EIGHT_HIGH -> Break Left Bandit Left Eight O Clock High Three Miles Engaging

EMERGENCY_LIGHTS -> Turn On The Emergency Exit Lights

ENTRY -> Entry Into The Threat Environment

FIRST_AID -> Open First Aid Kits For Quick Access

FIVE_HIGH_TWO -> Break Right Missile Left Five O Clock High Two Miles
    Engaged

FIVE_HIGH_THREE -> Break Right Bandit Right Five O Clock High Three Miles
    Rolling In

FIVE_LOW -> Break Right Missile Right Five O Clock Low Two Miles Engaged

FUEL -> Fuel Indicator

GOING_AWAY -> Going Away

IN_FLT_THRT_AVDNCE -> In Flight Threat Avoidance Support Duties

INTERNAL_LIGHTS -> Internal And External Lights

LANDING -> Landing At An Airfield In A Hostile Environment

LEAVING -> After Leaving A Threat Environment

LESSON_TEST -> Lesson Test

LOADMASTER -> Loadmaster And Scanner Only

LOOSE_ITEMS -> Loose Items

MISSILE_LEFT -> Missile Left Eight O Clock Low Three Miles Paralleling
```

Syntax Source File (Continued)

```
NEXT_SCREEN -> Next Screen

NINE_HIGH -> Hard Left Bandit Left Nine O Clock High Five Miles Rolling In

NINE_LOW_FIVE -> Hard Left Bandit Left Nine O Clock Low Five Miles Engaging

NINE_LOW_TWO -> Break Left Bandit Left Nine O Clock Low Two Miles Engaged

OFFLOADING -> Before Offloading Cargo in A Threat Environment

PILOT -> Pilot Flight Deck And Loadmaster Cargo Compartment

PREPARE_CARGO -> Prepare Cargo For Jettisoning

PREPARE_EQUIPMENT -> Prepare Survival Equipment Report Battle Damage

RED_LIGHTS -> Turn On The Red Lights

RENDEZVOUS -> Rendezvous With An AWACS

RETURNING -> After Returning From A Tactical V F R Mission

RIGHT_LEFT -> Right Left

ROLLINGIN -> Rolling In

SECURE_CLOSE -> Secure Loose Equipment Close Trap Or Cargo Doors

SECURE_SET -> Secure Loose Equipment Set Internal Lights To Dim Or Red

SET_GAIN -> Set Gain

SEVEN_LEVEL -> Break Right Missile Left Seven O Clock Level Two Miles
     Engaged

SEVEN_LOW -> Break Right Missile Right Seven O Clock Low Two Miles Engaged

STEP -> Step NUMBERS+

STOW -> Stow Survival Equipment Set Internal Lights To Normal

SURVIVAL_EQUIPMENT -> Survival Equipment

SURVIVAL_VEST -> Survival Vest
```

Syntax Source File (Continued)

```
TAKEOFF -> Before Takeoff From An Airfield In A Threat Environment

TEN_LEVEL -> Bandit Left Ten O Clock Level Six Miles Rolling In

TEN_LOW -> Break Left Missile Left Ten O Clock Low One Mile Engaged

THREE_HIGH -> Hard Left Bandit Left Three O Clock High Five Miles Rolling In

THREE_LOW -> Hard Left Bandit Left Three O Clock Low Five Miles Engaging

TITLE_SCREEN -> Title Screen

TURN -> Turn Left Bandit Left Eight O Clock High Five Miles Engaging

TYPE_OF_THREAT -> Type Of Threat

VERTICAL_POS -> Vertical Position

WHAT_DOING -> What It Is Doing

WHAT_SEE -> What You See

WHERE_SEE -> Where You See It

!
! Sentence Definitions
!
S --> AIRSPEED

S --> ALL_PERSONNEL

S --> APPLY_RESTRAINTS

S --> BANDIT

S --> BREAK

S --> BRIGHT

S --> CHANGE_MODEL

S --> CHEMICAL_DEFENSE

S --> CLOCK_CODE
```

Syntax Source File (Continued)

```
S --> CLOCK_SEGMENT
S --> CMBT_ENTRY_CHCKLST
S --> CMBT_ENTRY_CHCKLST_ACK
S --> CMBT_ENTRY_CHCKLST_CMPLT
S --> CMBT_EXIT_CHCKLST
S --> CMBT_EXIT_CHCKLST_ACK
S --> CMBT_EXIT_CHCKLST_CMPLT
S --> COMING
S --> COMMANDER
S --> COVER_WINDOWS
S --> CREWMEMBER
S --> DEGREES
S --> DIM
S --> EIGHT_HIGH
S --> EMERGENCY_LIGHTS
S --> ENTRY
S --> FIRST_AID
S --> FIVE_HIGH_TWO
S --> FIVE_HIGH_THREE
S --> FIVE_LOW
S --> FUEL
S --> GOING_AWAY
S --> IN_FLT_THRT_AVDNCE
S --> INTERNAL_LIGHTS
```

Syntax Source File (Continued)

```
S --> LEAVING
S --> LESSON_TEST
S --> LOADMASTER
S --> LOOSE_ITEMS
S --> MISSILE_LEFT
S --> NEXT_SCREEN
S --> NINE_HIGH
S --> NINE_LOW_FIVE
S --> NINE_LOW_TWO
S --> OFFLOADING
S --> PILOT
S --> PREPARE_CARGO
S --> PREPARE_EQUIPMENT
S --> RED_LIGHTS
S --> RENDEZVOUS
S --> RESPONSE_WORDS+
S --> RETURNING
S --> RIGHT_LEFT
S --> ROLLINGIN
S --> SECURE_CLOSE
S --> SECURE_SET
S --> SET_GAIN
S --> SEVEN_LEVEL
S --> SEVEN_LOW
```

Syntax Source File (Continued)

```
S --> STEP

S --> STOW

S --> SURVIVAL_VEST

S --> SURVIVAL_EQUIPMENT

S --> TAKEOFF

S --> TEN_LEVEL

S --> TEN_LOW

S --> THREE_HIGH

S --> THREE_LOW

S --> TITLE_SCREEN

S --> TURN

S --> TYPE_OF_THREAT

S --> VERTICAL_POS

S --> WHAT_DOING

S --> WHAT_SEE

S --> WHERE_SEE
```

APPENDIX D

CALCULATION OF *S* AND HOTELLING'S T^2

Calculation of ***S***

$ss_1 = \Sigma (y_1(i) - y_{11})^2 = 2$

$ss_2 = \Sigma (y_2(i) - y_{21})^2 = 14$

$ss_{12} = \Sigma (y_1(i) - 2)(y_2(i) - 4) = 4$

$ss_{21} = \Sigma (y_2(i) - 2)(y_1(i) - 4) = 4$

The results of these calculations form the matrix for Group 1 which becomes W_1:

$$W_1 = \begin{bmatrix} 2 & 4 \\ 4 & 14 \end{bmatrix}$$

Similar procedures are completed for the Group 2 matrix, W_2.

$$W_2 = \begin{bmatrix} 4 & 4 \\ 4 & 16 \end{bmatrix}$$

The two matrices are pooled to calculate ***S:***

$$S = \frac{W_1 + W_2}{n_1 + n_2 - 2}$$

Substituting values,

$$S = \frac{\begin{bmatrix} (2 & 4) & + & (4 & 4) \\ (4 & 14) & & (4 & 16) \end{bmatrix}}{7} = \frac{\begin{matrix} 6/7 & 8/7 \\ 8/7 & 30/7 \end{matrix}}{}$$

The inverse of ***S*** is then calculated using the difference **D** between $ss_1 ss_2$ - $ss_{12} ss_{21}$ (Manly, 1986). Table 8 extracted from an Excel Software spreadsheet demonstrates the inversion process:

S^{-1} Inversion Process

2	4	+	4	4	=	6	8
4	14		4	16		8	30
0.857143	1.142857						
1.142857	4.285714						
			6/7	*8/7*		*30/7*	*-(8/7)*
-0.95386			0.857143	1.142857		4.285714	-1.14286
						2.367347	2.367347
			8/7	*30/7*	=	*-(8/7)*	*6/7*
			1.142857	4.285714		-1.14286	0.857143
1.811	-0.483					2.367347	2.367347
-0.483	0.362						

Calculation of Hoteling's $\mathbf{T}^2$

Hotelling's $\mathbf{T}^2$ is calculated using S^{-1} (Stevens, 1986):

$$\mathbf{T}^2\_ = \frac{n_1 n_2}{n_1 + n_2} (\mathbf{y}_1 - \mathbf{y}_2 -)' \, S^{-1} \, (y_1 - y_2)$$

$$= \frac{3(6)}{3+6} \, (2-5,\ 4-8) \begin{pmatrix} 1.811 & -.483 \\ -.483 & .362 \end{pmatrix} \begin{pmatrix} 2-5 \\ 4-8 \end{pmatrix}$$

$$= (-6,\ 8) \begin{pmatrix} -3.501 \\ .001 \end{pmatrix} = 21$$

The ***F*** statistic is then derived from the following formula (Stevens, 1986):

$$F = \frac{n_1 + n_2 - p - 1}{(n_1 + n_2 - 2)\,p} \; T^2\_ \qquad (p = \textit{number of dependent variables})$$

$$\frac{9 - 2 - 1}{7(2)} (21) = 9$$

For 2 and 6 degrees of freedom, Fcv = 5.14 with a at .05, indicating a statistical significant difference between the two groups.

APPENDIX E

TEST RESULTS FILE

February 22,1995 - 9:1:24
Speech Recognition Not Active...
In Release Mode...
Total Time: 31.01 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 9
Number of final test questions incorrect: 16

February 23,1995 - 14:2:59
Speech Recognition Active...
In Release Mode...
Total Time: 50.78 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 9
Number of final test questions incorrect: 16
Number of invalid speech responses: 164

February 23,1995 - 15:2:16
Speech Recognition Active...
In Release Mode...
Total Time: 42.08 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 9
Number of final test questions incorrect: 16
Number of invalid speech responses: 89

February 24,1995 - 8:59:50
Speech Recognition Active...
In Release Mode...
Total Time: 48.34 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 4
Number of progress check questions incorrect: 4
Number of final test questions correct: 18
Number of final test questions incorrect: 7
Number of invalid speech responses: 88

February 24,1995 - 10:0:49
Speech Recognition Active...
In Release Mode...
Total Time: 46.89 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 8
Number of progress check questions incorrect: 0
Number of final test questions correct: 10
Number of final test questions incorrect: 15
Number of invalid speech responses: 84

February 24,1995 - 10:59:28
Speech Recognition Active...
In Release Mode...
Total Time: 54.91 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 20
Number of final test questions incorrect: 5
Number of invalid speech responses: 187

February 24,1995 - 11:57:28
Speech Recognition Active...
In Release Mode...
Total Time: 59.27 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 10
Number of final test questions incorrect: 15
Number of invalid speech responses: 89

February 24,1995 - 13:25:15
Speech Recognition Not Active...
In Release Mode...
Total Time: 55.11 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 13
Number of final test questions incorrect: 12

February 24,1995 - 14:3:23
Speech Recognition Active...
In Release Mode...
Total Time: 56.62 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 8
Number of final test questions incorrect: 17
Number of invalid speech responses: 90

February 24,1995 - 15:4:46
Speech Recognition Active...
In Release Mode...
Total Time: 51.87 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 14
Number of final test questions incorrect: 11
Number of invalid speech responses: 137

February 28,1995 - 8:3:29
Speech Recognition Active...
In Release Mode...
Total Time: 46.89 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 17
Number of final test questions incorrect: 8
Number of invalid speech responses: 99

February 28,1995 - 8:56:39
Speech Recognition Active...
In Release Mode...
Total Time: 56.46 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 10
Number of final test questions incorrect: 15
Number of invalid speech responses: 155

February 28,1995 - 10:0:36
Speech Recognition Active...
In Release Mode...
Total Time: 49.06 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 9
Number of final test questions incorrect: 16
Number of invalid speech responses: 138

February 28,1995 - 14:6:56
Speech Recognition Active...
In Release Mode...
Total Time: 54.1 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 17
Number of final test questions incorrect: 8
Number of invalid speech responses: 132

February 28,1995 - 15:7:51
Speech Recognition Active...
In Release Mode...
Total Time: 55.58 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 4
Number of progress check questions incorrect: 4
Number of final test questions correct: 8
Number of final test questions incorrect: 17
Number of invalid speech responses: 129

February 28,1995 - 16:11:5
Speech Recognition Active...
In Release Mode...
Total Time: 54.3 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 12
Number of final test questions incorrect: 25
Number of invalid speech responses: 128

March 1,1995 - 8:57:18
Speech Recognition Active...
In Release Mode...
Total Time: 57.3 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 11
Number of final test questions incorrect: 14
Number of invalid speech responses: 152

March 1,1995 - 10:5:42
Speech Recognition Active...
In Release Mode...
Total Time: 51.74 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 14
Number of final test questions incorrect: 11
Number of invalid speech responses: 157

March 1,1995 - 11:5:58
Speech Recognition Active...
In Release Mode...
Total Time: 46.99 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 7
Number of progress check questions incorrect: 1
Number of final test questions correct: 15
Number of final test questions incorrect: 10
Number of invalid speech responses: 90

March 1,1995 - 13:0:27
Speech Recognition Not Active...
In Release Mode...
Total Time: 34.76 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 15
Number of final test questions incorrect: 10

March 1,1995 - 14:0:40
Speech Recognition Not Active...
In Release Mode...
Total Time: 35.23 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 14
Number of final test questions incorrect: 11

March 1,1995 - 14:34:23
Speech Recognition Not Active...
In Release Mode...
Total Time: 27.23 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 14
Number of final test questions incorrect: 11

March 1,1995 - 15:4:31
Speech Recognition Not Active...
In Release Mode...
Total Time: 41.18 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 7
Number of progress check questions incorrect: 1
Number of final test questions correct: 23
Number of final test questions incorrect: 2

March 2,1995 - 8:58:59
Speech Recognition Not Active...
In Release Mode...
Total Time: 36.19 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 13
Number of final test questions incorrect: 12

March 2,1995 - 9:59:47
Speech Recognition Not Active...
In Release Mode...
Total Time: 31.99 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 12
Number of final test questions incorrect: 13

March 2,1995 - 10:59:39
Speech Recognition Not Active...
In Release Mode...
Total Time: 34.72 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 13
Number of final test questions incorrect: 12

March 2, 1995 - 13:2:13
Speech Recognition Not Active...
In Release Mode...
Total Time: 38.25 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 7
Number of progress check questions incorrect: 1
Number of final test questions correct: 22
Number of final test questions incorrect: 3

March 2,1995 - 13:4:54
Speech Recognition Not Active...
In Release Mode...
Total Time: 45.88 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 8
Number of final test questions incorrect: 17

March 2,1995 - 14:4:22
Speech Recognition Not Active...
In Release Mode...
Total Time: 26.77 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 13
Number of final test questions incorrect: 12

March 2,1995 - 15:4:38
Speech Recognition Not Active...
In Release Mode...
Total Time: 44.44 minutes
Number of callouts correct: 1
Number of callouts incorrect: 5
Number of progress check questions correct: 8
Number of progress check questions incorrect: 0
Number of final test questions correct: 21
Number of final test questions incorrect: 4

March 3,1995 - 8:55:5
Speech Recognition Not Active...
In Release Mode...
Total Time: 35.05 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 16
Number of final test questions incorrect: 9

March 3,1995 - 9:32:53
Speech Recognition Not Active...
In Release Mode...
Total Time: 23.77 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 3
Number of progress check questions incorrect: 5
Number of final test questions correct: 17
Number of final test questions incorrect: 8

March 3,1995 - 9:59:51
Speech Recognition Not Active...
In Release Mode...
Total Time: 33.49 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 4
Number of progress check questions incorrect: 4
Number of final test questions correct: 19
Number of final test questions incorrect: 6

March 3,1995 - 10:59:16
Speech Recognition Not Active...
In Release Mode...
Total Time: 26.7 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 17
Number of final test questions incorrect: 8

March 3,1995 - 13:5:8
Speech Recognition Not Active...
In Release Mode...
Total Time: 33.23 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 4
Number of progress check questions incorrect: 4
Number of final test questions correct: 16
Number of final test questions incorrect: 9

March 3,1995 - 14:4:29
Speech Recognition Not Active...
In Release Mode...
Total Time: 36.24 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 5
Number of progress check questions incorrect: 3
Number of final test questions correct: 8
Number of final test questions incorrect: 17

March 3,1995 - 15:4:16
Speech Recognition Not Active...
In Release Mode...
Total Time: 36.02 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 10
Number of final test questions incorrect: 15

March 6,1995 - 10:5:42
Speech Recognition Active...
In Release Mode...
Total Time: 47.74 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 14
Number of final test questions incorrect: 11
Number of invalid speech responses: 82

March 6,1995 - 10:5:42
Speech Recognition Active...
In Release Mode...
Total Time: 51.34 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 14
Number of final test questions incorrect: 11
Number of invalid speech responses: 85

March 6,1995 - 10:5:42
Speech Recognition Active...
In Release Mode...
Total Time: 49.24 minutes
Number of callouts correct: 0
Number of callouts incorrect: 6
Number of progress check questions correct: 2
Number of progress check questions incorrect: 6
Number of final test questions correct: 14
Number of final test questions incorrect: 11
Number of invalid speech responses: 81

APPENDIX F

MANOVA DATA INPUT

MANOVA Data Input

```
TITLE 'MANOVA'.
DATA LIST FREE / GP Y1 Y2 Y3.
BEGIN DATA
1.0  52.0  25.0   26.77
1.0  32.0  37.5   45.88
1.0  52.0  37.5   34.77
1.0  48.0  62.5   31.99
1.0  52.0  37.5   36.19
1.0  36.0  37.5   31.01
1.0  92.0  87.5   41.18
1.0  56.0  25.0   27.23
1.0  60.0  25.0   34.76
1.0  84.0  87.5   36.06
1.0  84.0  100.0  44.44
1.0  64.0  62.5   35.05
1.0  68.0  37.5   23.77
1.0  76.0  50.0   33.49
1.0  68.0  62.5   26.7
1.0  64.0  50.0   33.23
1.0  32.0  62.5   36.24
1.0  40.0  25.0   36.02
1.0  88.0  87.5   38.25
1.0  84.0  62.5   35.23
2.0  60.0  87.5   46.99
2.0  56.0  25.0   51.74
2.0  44.0  25.0   57.3
2.0  32.0  50.0   55.58
2.0  68.0  37.5   54.1
2.0  36.0  25.0   50.78
2.0  52.0  62.5   55.11
2.0  40.0  62.5   59.27
2.0  48.0  62.5   54.3
2.0  44.0  75.0   57.2
2.0  80.0  62.5   54.91
2.0  40.0  100.0  46.89
2.0  72.0  50.0   48.34
2.0  36.0  37.5   42.08
2.0  56.0  62.5   51.1
2.0  32.0  62.5   56.62
2.0  56.0  37.5   51.87
2.0  68.0  25.0   46.89
2.0  40.0  25.0   56.46
2.0  36.0  25.0   49.06
END DATA.
MANOVA Y1 Y2 Y3 BY GP(1,2)
/OMEANS
/ANALYSIS
/PRINT=CELLINFO (COV)
/PRINT=HOMOGENEITY (COCHRAN BARTLETT)
/PLOT=BOXPLOT
/DESIGN.
```

APPENDIX G

MANOVA ANALYSIS

MANOVA Analysis

40 cases accepted
0 cases rejected because of out-of-range factor values.
1 design will be processed.
-------------------CELL NUMBER

	1	2
Variable		
GP	1	2

Cell Means and Standard Deviations

Variable..Y1

FACTOR	CODE	Mean	Std. Dev.	N
GP	1	61.600	18.914	20
GP	2	49.800	14.244	20
For entire sample		55.700	17.574	40

Variable..Y2

FACTOR	CODE	Mean	Std. Dev.	N
GP	1	53.125	23.604	20
GP	2	50.000	22.580	20
For entire sample		51.562	22.855	40

Variable..Y3

FACTOR	CODE	Mean	Std. Dev.	N
GP	1	34.413	5.657	20
GP	2	52.330	4.518	20
For entire sample		43.371	10.385	40

MANOVA Analysis (Continued)

```
40 cases accepted.
0 cases rejected because of out-of-range factor values.
1 design will be processed.
 - - - - - - - - - -
          CELL NUMBER
           1   2
 Variable
  GP         1   2
 Univariate Homogeneity of Variance Tests
 Variable .. Y1
     Cochrans C(19,2) =                 .63808, P =  .226 (approx.)
     Bartlett-Box F(1,4332) =          1.46873, P =  .226
 Variable .. Y2
     Cochrans C(19,2) =                 .52216, P =  .849 (approx.)
     Bartlett-Box F(1,4332) =           .03639, P =  .849
 Variable .. Y3
     Cochrans C(19,2) =                 .61058, P =  .335 (approx.)
     Bartlett-Box F(1,4332) =           .92844, P =  .335

* * ANALYSIS  OF  VARIANCE -- DESIGN   1 * *
 Cell Number .. 1
 Variance-Covariance matrix
              Y1        Y2        Y3
 Y1        357.726
 Y2        302.632   557.155
 Y3         12.587    63.286    32.007

 Determinant of Variance-Covariance matrix =    2409027.52305
 LOG(Determinant) =                              14.69473
```

MANOVA Analysis (Continued)

Cell Number .. 2
Variance-Covariance matrix

	Y1	Y2	Y3
Y1	202.905		
Y2	10.526	509.868	
Y3	-9.578	1.138	20.414

Determinant of Variance-Covariance matrix = 2062351.62387
LOG(Determinant) = 14.53936

\- - - - - - - - - -

Pooled within-cells Variance-Covariance matrix

	Y1	Y2	Y3
Y1	280.316		
Y2	156.579	533.512	
Y3	1.504	32.212	26.210

Determinant of pooled Variance-Covariance matrix 3000310.52335
LOG(Determinant) = 14.91423

\- - - - - - - - - -

Multivariate test for Homogeneity of Dispersion matrices

Boxs M = 11.29287
F WITH (6,10462) DF = 1.72009, P = .112 (Approx.)
Chi-Square with 6 DF = 10.32703, P = .112 (Approx.)

\- - - - - - - - - -

MANOVA Analysis (Continued)

* * ANALYSIS OF VARIANCE -- DESIGN 1 * *

EFFECT .. GP
Multivariate Tests of Significance (S = 1, M = 1/2, N = 17)

Test Name	Value	Approx. F	Hypoth. DF	Error DF	Sig. of F
Pillais	.78202	43.05125	3.00	36.00	.000
Hotellings	3.58760	43.05125	3.00	36.00	.000
Wilks	.21798	43.05125	3.00	36.00	.000
Roys	.78202				

- - - - - - - - - -
Univariate F-tests with (1,38) D. F.

Variable	Hypoth. SS	Error SS	Hypoth. MS	Error MS	F	Sig. of F
Y1	1392.40000	10652.0000	1392.40000	280.31579	4.96725	.032
Y2	97.65625	20273.4375	97.65625	533.51151	.18304	.671
Y3	3210.00969	995.99385	3210.00969	26.21036	122.47101	.000

* * ANALYSIS OF VARIANCE -- DESIGN 1 * *

EFFECT .. CONSTANT
Multivariate Tests of Significance (S = 1, M = 1/2, N = 17)

Test Name	Value	Approx. F	Hypoth. DF	Error DF	Sig. of F
Pillais	.98885	1064.11945	3.00	36.00	.000
Hotellings	88.67662	1064.11945	3.00	36.00	.000
Wilks	.01115	1064.11945	3.00	36.00	.000
Roys	.98885				

- - - - - - - - - -
Univariate F-tests with (1,38) D. F.

Variable	Hypoth. SS	Error SS	Hypoth. MS	Error MS	F	Sig. of F
Y1	124099.600	10652.0000	124099.600	280.31579	442.71356	.000
Y2	106347.656	20273.4375	106347.656	533.51151	199.33526	.000
Y3	75242.6133	995.99385	75242.6133	26.21036	2870.71985	.000

REFERENCES

Adams, R. (1990). Sources of automatic identification and data collection. New York: Van Nostrand Reinhold.

Ainsworth, W. A., & Pratt, S. R. (1992). Feedback strategies for error correction in speech recognition systems. International Journal Man-Machine Studies, 36, 833-842.

Baecker, R. M., & Buxton, W. A. (1987). Human-computer interaction, a multidisciplinary approach. Los Altos, CA: Morgan Kaufman.

Burger, M. L., & DeSoi, John F. (1991). The cognitive apprenticeship analogue: A strategy for using ITS technology for the delivery of instruction and as a research tool for the study of teaching and learning. International Journal of Man-Machine Studies, 36, 775-795.

Burns, H., Parlett, J. W., & Redfield, C. L. (Eds.). (1991). Intelligent tutoring systems: Evolutions in design. Hillsdale, NJ: Erlbaum.

Casali, S., Williges, B., and Dryden, R. (1990, April) Effect of recognition accuracy and vocabulary size of a speech recognition system on task performance and user acceptance. Human Factors, 34(l) 184-196.

Chamberlain, M. A. (1993). Computerized voice recognition systems and their application to the mobility impaired (Doctoral dissertation, The Florida State University, 1993). Dissertation Abstracts International, A54/10.

Coburn, P. (1982). Computers in education. Reading, MA: Addison-Wesley.

Creech, W. E. (1994, November). Voice recognition: A reborn technology for education and training. Proceedings, 16th Interservice/Industry Training Systems and Education Conference (I/ITSEC), 3-13.

Creitz, W. (1991). Applications for voice processing technology. Rockville, MD: Stoneridge Technical Services.

Doulianne, G., Garudadri, H., Trudelle, G., Hollan, R., Lennig, M., & O'Shaunessy, D. (1994, February) Experiments in continuous speech recognition using books on tape. Speech Communication Journal, 14, (1), 40-60.

Doulianne, G., Kenny, P., Lennig, M., O'Shaughnessy, D., & Mermelstein, P. (1994, February). Books on tape as training data for continuous speech. Speech Communication Journal, 14, (1), 61-70.

Farr, M. J. (1987). The long term retention of knowledge and skills: A cognitive and instructional perspective. New York: Springer-Verlag.

Filipczak, B. (1993, March). Adaptive technology for the disabled. Training, pp. 23-29.

Fleming, M. L. (1987). Displays and communication. In R.M. Gagne (Ed.), Instructional technology: Foundations (pp. 233-256). Hillsdale, NJ: Erlbaum.

Foster, P. & Schalk, T. B. (1993). Speech recognition: The complete practical reference guide. New York: Telecom Library.

Frankish, C., Jones, D., & Hapeshi, K. (1992). Decline in accuracy of automatic speech recognition as a function of time on task: Fatigue or voice drift? International Journal Man-Machine Studies, 36, 797-816.

Gagne, R. M. & Briggs, L. J. (1979). Principles of instructional design (2nd ed.) New York: Holt, Rhinehart and Winston.

Gardiner, M., & Christie, B. (Eds.). (1987). Applying cognitive psychology to user-interface design. New York:: Wiley.

Golas, K. C., Fredrickson, R. C., & Negri, M. A. (1994, November). Computer-based English language training for the Royal Saudi Naval Forces. Proceedings, 16th Interservice/Industry Training Systems and Education Conference (I/ITSEC), 3-14.

Greene, S., Gould, J., Boies, S., Rasamny, M., & Meluson, A. (1992). Entry and selection-based methods of human-computer interaction. Human Factors, 34(l), 97-113.

Harris, J. (1994). A meta-analysis of the effectiveness of feedback in computer-based instruction for adults. Unpublished doctoral dissertation, East Texas State University, Commerce.

Harris, R. (1985). A primer of multivariate statistics. Orlando, FL: Academic Press.

Jonassen, D. H., & Hannum, W. H. (1987). Research-based principles for designing computer software. Educational Technology, 27 (12), 7-14.

Karl, L. R., Pettey, M. & Shneiderman, B. (1993). Speech versus mouse commands for word processing: An empirical evaluation. International Journal Man-Machine Studies, 39, 667-687.

Karrer, U. (1988). How to develop quality courseware: Results of a survey. Journal of Educational Technology Systems, 16(4), 323-335.

Kerlinger, F. N. (1986). Foundations of behavioral research (3rd ed.). Orlando, FL: Harcourt Brace Jovanovich.

Kincaid, J. P., Mullally, D., Meyer, C., & Bramble, W., Jr. (1993, November). Development of a technology for language instruction used for multimedia PCs. Proceedings of the 15th Interservice/Industry Training Systems and Education Conference I/ITEC, 880-885.

Kirk, R. E. (1982) Experimental design: Procedures for the behavioral sciences. (2nd ed.). Belmont, CA: Brooks/Cole.

Klass, P. J. (1991, July). FAA evaluating inexpensive, PC-based system for practicing controller skills. Aviation Week and Space Technology, pp. 60-61.

Lee, K. F. (1989). Automatic speech recognition: The development of the SPHINX system. Norwell, MA: Kluwer Academic.

Leggett, J. & Williams, G. (1984). An empirical investigation of voice as an input modality for computer programming. International Journal of Man-Machine Studies, 21, 493-520.

Lewis, K., Pettey, M., & Shneiderman, B. (1993). Speech versus mouse commands for word processing: An empirical evaluation. International Journal of Man-machine Studies, 39, 667-687.

Lillie, D., Hannum, W., & Stuck, G. (1989). Computers and effective instruction. White Plains, NY: Longman.

Manly, B.F. (1986). Multivariate statistical methods: A primer. New York: Chapman and Hall.

Martin, G. L. (1989). The utility of speech input in user-computer interfaces. International Journal of Man-Machine Studies, 21, 49-63.

Merriam-Webster's ninth new collegiate dictionary. (1983). Springfield, MA: Merriam-Webster.

Merrill, J. (1987, Winter). Levels of questioning and forms of feedback: Instructional factors in courseware design. Journal of Computer-Based Instruction, 14 (1), 18-22.

Morgan, N. (1984). Talking chips. New York: McGraw Hill.

Mullally, D., Kincaid, J. P., & Kishek, G. (1993, November). Human-computer interface and training issues in the design of an expedient computer-based language training program. Proceedings of the 15th Interservice/Industry Training Systems and Education Conference, 505-512.

Murphy, M. J. (1992). A framework for testing the learning of cognition-based human-computer interfaces. McMaster University, Canada. Dissertation Abstracts International, B54/09, p. 4893. (University Microfilm No. ABD92-14857).

Nye, J. M. (1982). Human factors analysis of speech recognition systems. Speech Technology, 1, 50-57.

Paint: User's reference manual. (1991). USA: Wicat Systems.

Phonetic Engine 400 System Development Kit. (1993). Boulder, CO: Speech Systems.

Poirot, J. L. & Knezek, G. A. (1992, November). Experimental designs for determining the effectiveness of technology in education. The Computing Teacher, pp. 8-9.

Poock, G.K. (1982). Voice recognition boosts command terminal throughput. Speech Technology, 1, 36-39.

Rudnicky, A., Hauptmann, A., & Lee, K. (1994, March). Survey of current speech technology. Communications of the ACM, 37, (3), 52-57.

Schmalz, L. & Sipl, C. (1972). Computer glossary for students and teachers. New York: Funk & Wagnalls.

Stevens, J. (1986). Applied multivariate statistics for the social sciences. Hillsdale, NJ: Erlbaum.

Tessmer, M., Jonassen, D., & Caverly D.C. (1989). A nonprogrammer's guide to designing instruction for microcomputers. Englewood, CO: Libraries Unlimited.

Thorkildsen, R. & Reid, R. (1989, Spring). An investigation of the reinforcing effects of feedback on computer-assisted instruction. Journal of Special Education Technology, (3), 125-135.

Thyfault, M. (1994, May). The power of voice. Information Week,. pp. 39-46.

Travers, R. M. W. (1982). The essentials of learning. New York: Macmillian.

Visick, D., Johnson, P. & Long, J. (1984). The use of simple speech recognisers in industrial applications. Proceedings of INTERACT '84, First IFIP Conference on Human-Computer Interaction, London.

Yunus, J. (1992). An investigation of an approach to the design of computer-based tools for training and rehabilitation of the handicapped (Doctoral dissertatiion, University of Kent at Canterbury, 1992). Dissertation Abstracts International, A53/08.