Speech Recognition and Cursor Feedback in Graphics Software Applications

Edward W. Sarfeld, B.A. (Honours)

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements for the degree of
Master of Architecture

Carleton University

Ottawa, Ontario

April, 2004

© 2004, Edward W. Sarfeld

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-93996-0

Our file Notre référence

ISBN: 0-612-93996-0

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this dissertation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de ce manuscrit.

While these forms may be included in the document page count, their removal does not represent any loss of content from the dissertation.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

PAGINATION ERROR.

TEXT COMPLETE.

ERREUR DE PAGINATION.

LE TEXTE EST COMPLET.

ABSTRACT

Automatic speech recognition (ASR) is an evolving technology that is used to enhance human interaction with computers and electronic devices. Historically, speech recognition on desktop computers was used primarily to translate the spoken word into text for input within word processing software. Recently, there has been an increase in the use of speech recognition for command and control of software applications and hardware. This study examines the challenges and potential benefits of augmenting direct manipulation (keyboard and pointing device input) within computer aided design (CAD) and graphics applications through the use of automatic speech recognition. This thesis also explores the potential of cursor feedback with respect to error state identification when using ASR.

TABLE OF CONTENTS

ABSTRACT	iii
TABLE OF CONTENTS	iv
LIST OF TABLES	vii
LIST OF APPENDICES	ix
1.0 INTRODUCTION	1
1.1 Problems and Motivations	2
1.2 Goals	3
1.3 Scope	4
1.4 Approach	5
1.5 Organization	6
2.0 AUTOMATIC SPEECH RECOGNITION	7
2.1 Speech Recognition History	8
2.2 ASR Technical Overview	12
2.3 ASR in Application	18
2.4 Commercial Vendors	23
2.5 Summary	26
3.0 PRIOR STUDIES AND COMMERCIAL APPLICATION	28
3.1 ASR as an Adjunctive Input Mode	29
3.2 ASR as an Adjunctive Input Mode in CAD and Graphics Applications	33
3.3 Commercial Speech-Enabled Graphics or CAD/CAM Applications	42

3.4 Summary	50
4.0 SPEECH RECOGNITION AND DESIGN CONSIDERATIONS	52
4.1 Speech Interface Expectations	53
4.2 ASR Integration	54
4.3 Unique Elements of ASR Interface Design	57
4.4 Prototype Design Considerations	62
4.4.1 Prototype Constraints	63
4.4.2 Current Patterns of Use / Software Evolution	64
4.4.3 User Profile	65
4.4.4 Unique Functionality – One: ASR as an Adjunctive Input Modality	66
4.4.5 Unique Functionality – Two: Enhanced Cursor Feedback	67
4.4.6 Observations from Commercial Implementations	71
4.5 Summary	72
5.0 CASE STUDY	74
5.1 Study Participants	75
5.2 Experiment Design	75
5.3 Phase One – Prototype Development	76
5.4 Phase Two – Error State Scenarios	78
5.5 Survey Procedure and Evaluation Criteria	79
5.6 Enhanced Cursor Feedback Scenarios	81
5.7 Survey Results	90
5.7.1 Participant Self-Evaluation on Software Experience	90
5.7.2 Enhanced Cursor Feedback Evaluation	94

5.7.3 Question Responses and General Comments	106
5.7.4 Results Synopsis & Summary	110
5.8 Summary	112
6. Future Research and Conclusion	114
6.1 Future Research	116
6.1.1 Design Refinement & Recommendations	116
6.1.2 Creating the Next Test Model	119
6.1.3 Obstacles to Success	121
6.2 Conclusion	123
BIBLIOGRAPHY	127
APPENDICES	131

LIST OF TABLES

Figure 1: Star Trek Voice Operated Dimmer Control	22
Figure 2: IN CUBE Command Panel	43
Figure 3: thinkdesign Mini-Dialogue Input Fields	48
Figure 4: Rectangular Mask Tool Cursor States	68
Figure 5: Dragon NaturallySpeaking Results Box - Example States	69
Figure 6: thinkdesign Speech Toolbar	70
Figure 7: Scenario One Cursor State Versions	82
Figure 8: Scenario Two Cursor State Versions	83
Figure 9: Scenario Three Cursor State Versions	84
Figure 10: Scenario Four Cursor State Versions	86
Figure 11: Scenario Five Cursor State Versions	87
Figure 12: Scenario Six Cursor State Versions	89
Figure 13: AutoCAD Familiarity	91
Figure 14: PHOTO-PAINT Familiarity	92
Figure 15: Other CAD or Graphics Familiarity	92
Figure 16: Speech Recognition Familiarity	93
Figure 17: Scenario One Cursor Feedback Preference	95
Figure 18: Scenario Two Cursor Feedback Preference	96
Figure 19: Scenario Three Cursor Feedback Preference	98
Figure 20: Scenario Four Cursor Feedback Preference	100
Figure 21: Scenario Five Cursor Feedback Preference	102

Figure 22: Scenario Six Cursor Feedback Preference	104
Figure 23: Survey Question One	106
Figure 24: Survey Question Two	106
Figure 25: Survey Question Three	108

LIST OF APPENDICES

Appendix A – Chapter 3.2 Extended Reviews	132
Appendix B – Cursor States	144
Appendix B – Cursor List	145
Appendix D – Survey Form	150
Appendix E – Survey Data	156
Appendix F – Thesis Presentation Slides	158

1.0 INTRODUCTION

As computing technology evolves and processing power increases, the variety of computing devices and software that enter the user domain afford many new means for people to interact with technology. These computing devices vary from corporate mainframe computers to personal laptop and desktop computers, personal digital assistants and mobile cellular telephones. As a consequence of this hardware development, software user interfaces have evolved. New interfaces must enable users to interact with a range of devices in a seamless manner in a multitude of environments. Currently, the most common interface on personal computers employ a desktop metaphor and use input from a keyboard and pointing device. Many new computing devices are small, portable and don't easily accommodate input via a mouse or keyboard. Small devices such as the personal digital assistant incorporate a touch screen and/or automatic speech recognition. To facilitate greater ease of use, multimodal or mixed initiative interaction is becoming more commonplace. As the technology matures, automatic speech recognition (ASR) is gaining wider acceptance and recognition.

1.1 Problems and Motivations

Automatic speech recognition has been in development for the past 50 years. The first speech recognition engine was capable of recognising the numbers from 1 to 10 when spoken singly by a specific person. Today, commercially available ASR desktop software requires short enrolment/training times and provides recognition accuracy that approaches 99%.¹ In the area of telephony, interactive voice response (IVR) systems have used the advances in speech recognition and speech synthesis output to facilitate easy access to online database transactions and support services such as financial transactions and transportation schedules.

Where the consumer market is concerned, ASR has been directed primarily to the task of dictation. Products such as Dragon *NaturallySpeaking* and IBM *Viavoice* are targeted to users wishing to dictate text into a word processing software application. Accordingly, many of the features developed for these commercial ASR packages are specifically geared to allow dictation and control text formatting within word processors. Recently however, greater emphasis has been placed on the use of speech commands to control other types of software applications and operating system functions. In part, these advances in ASR technology form the basis of this thesis and have led me to explore the value and

¹ Scansoft Dragon *NaturallySpeaking* Version 7 reports 99% recognition accuracy.

viability of ASR in computer aided design (CAD) and graphics creation software.

Through the review of previous studies and literature on the use of ASR to control software, in particular graphics and CAD applications, I have focused on what I perceive as the two critical design issues that would increase the usability and user acceptance of ASR in graphics creation software. These issues are: 1) the integration of ASR into the user interface in such a method that enhances workflow and task completion 2) the identification and handling of recognition errors. This thesis explores methods to remediate these issues.

1.2 Goals

The goals of this thesis are twofold: 1) assess the potential of ASR to augment the user interface of CAD and graphics creation programmes² by decreasing the dependence on the keyboard 2) assess the potential of the cursor to provide feedback on ASR-related input.

² The phrase “graphics creation programmes” refers to a range of programmes used to create graphical images that may, for example, be used in print media or on the World Wide Web. The programmes can be grouped into three general categories - illustration, digital imaging and motion graphics. Although each has specific features intrinsic to create output, they function in a similar fashion using a pointing device and keyboard as the mode of interaction. Other examples of graphics creation programmes not reviewed in this study include, Adobe Illustrator, Adobe Photoshop, CorelDRAW, Macromedia Flash, and Macromedia Freehand.

I have chosen to study the integration of ASR into CAD and graphics creations programmes because of the potential enhancements to the workflow. This includes a reduction in hand movement between the keyboard and pointing device, as well as a reduction in pointer movement within the interface to select tools or navigate menus. Unlike ASR-enabled word processing programmes where the primary utility of ASR is for text dictation, it is believed that graphics creation programmes would be particularly well suited to utilize ASR for command and control of the software functions. ASR would provide an excellent means to invoke commands, access features buried in the interface, and input alpha-numeric data.

1.3 Scope

To realize the above goals, I have chosen to limit the scope of this thesis to the following criteria:

- Assessing current user acceptance of ASR as an adjunctive input mode for graphics and CAD software applications;
- Assessing enhanced cursor feedback as means to identify ASR error states.

1.4 Approach

The topic this thesis explores is the integration of speech user interface (SUI) elements into a graphics creation programme within a graphical user interface (GUI). For the purpose of this thesis, I have restricted the study to IBM applications running on a Microsoft Windows operating system. Dragon NaturallySpeaking 6 Professional, a leading commercial ASR software package, was used to create ASR enabled demonstration prototypes. The features available within the CAD and graphics applications I selected are representative of those available within their respective industries. Autodesk AutoCAD LT 2002 is a well established CAD application within the PC software industry. Corel PHOTO-PAINT 11 is a bitmap editing programme representative of the graphic manipulation software commonly used in graphics design and development. The interfaces of both applications have evolved through many version releases and are optimised to function quickly and efficiently using a pointing device (mouse or pen tablet) and keyboard. A survey was prepared using the speech-enabled software to assess user acceptance of the integration of ASR and the effectiveness of cursor based feedback for error identification. The creation of the survey is described in greater detail in Chapter 5.

1.5 Organization

Following this introduction, this thesis is organized in the following structure:

- Chapter 2 provides a review of related work. This includes: a brief history of speech recognition technology, a technical overview, and review of current commercial vendors;
- Chapter 3 provides an analysis of prior studies of the integration of speech recognition with graphics manipulation packages and of previous commercial applications;
- Chapter 4 describes design issues related to the integration of ASR within a GUI application and discusses those issues in context of the case study;
- Chapter 5 describes the design and creation of the case study, testing methodology and results;
- Chapter 6 examines the results of the case study, identifies other areas for future research and provides concluding remarks.

2.0 AUTOMATIC SPEECH RECOGNITION

Automatic computer speech recognition is the interpretation and transcription of acoustic information from a spoken utterance. The data is interpreted by mapping the acoustic information to the intended linguistic equivalents. Statistical processes based on Hidden Markov Models³ (HMM) are used to evaluate multiple hypotheses of the utterance's meaning. Generally, this is completed within a speech recognition system by using lexicons, language models, and acoustic models.

This chapter reviews the history and development of automatic speech recognition (ASR), from its roots in academic institutions to its implementation by commercial vendors. This will encompass a brief technical overview of the components of speech recognition and its application in software.

The review is limited to the development of speech recognition systems for desktop computer software applications and does not outline the development of speech recognition for telephony or call centre use.

³ HMM is a numeric analysis which determines the probability of the next item in a string of items. It is a two-stage probabilistic process represented graphically by means of states, transitions between states and outputs that can be used to model speech. It is called "hidden" because the sequence of state that leads to a recognition decision is not necessarily known to the model's probabilistic nature. (Rodman, 1999)

2.1 Speech Recognition History

Computer speech recognition systems process human speech to identify words and phrases.

The first demonstration of a speech recognizer was developed at the AT&T Bell Laboratories in 1952. The system could recognize the digits 0-9, when spoken singly, with an accuracy of 98%. To obtain this level of recognition accuracy, the equipment utilized a speaker-dependent template-matching algorithm and required speaker training, with each of the words being spoken in a discrete dictation fashion. (Davis et al 1952)

In 1956, a system developed at the RCA Laboratory allowed ten distinct syllables to be recognized from a single speaker. In 1959, researchers at University College in England and Lincoln Laboratories at MIT, demonstrated systems that could recognize four vowels and nine consonants when spoken singly, and ten vowels in a speaker-independent manner, respectively.

In the early 1960s, Japanese researchers developed hardware vowel, phoneme,⁴ and digit recognizers. During this decade, digital computers were developed which provided the platform for much of the speech recognition development

4 Phonemes are described in Chapter 2.2.

used to the present day. By the late 1960s, research in the dynamic tracking of phonemes began.

In 1971, the Defence Advanced Research Projects Agency (DARPA) established the Speech Understanding Research (SUR) programme to develop a computer programme that could understand continuous speech. DARPA invested \$3 million a year for five years in numerous research labs and academic institutions to advance this project. This infusion of funding increased the breadth and depth of parties contributing to research in speech recognition.

The 1970s witnessed numerous milestones in the advancement of discrete word recognition. These included the use of linear predictive coding⁵ (LPC) in speech recognition and pattern recognition in speech understanding. Research and experimentation in speaker-independent⁶ systems were also advanced in particular by AT&T BELL Labs.

Recognition speed and accuracy increased throughout the 1980s as hardware computation speeds increased. Increases in speed resulted in the reduction of the time required to train the system to a user's voice – known colloquially as “enrolment time.” This prompted a new vision of speech recognition as applied to

5 Linear predictive coding is a tool used in speech processing for representing the spectral envelope of a digital signal of speech in compressed form, using the information of a linear predictive model.

6 A speaker-independent recognition system allows speech input from multiple users speaking a specific language and is explained in Chapter 2.2.

mass consumer use. Ideally, such a system would be speaker-independent, and allow for continuous natural speech,⁷ using the entire vocabulary of any desired language. Smaller, commercially-oriented companies, such as Dragon Systems in 1982, emerged to create speech recognition software with personal computers (PC). In 1984, IBM systems demonstrated the first discrete dictation, large vocabulary (5,000-word) recognition system achieving 95% accuracy after user training. The IBM system ran on a mainframe computer. In addition, in 1984, Dragon Systems released the first speech recognition package to be incorporated into a personal computer, the Apricot Portable⁸. Dragon Systems also demonstrated the first discrete recognition system with a large vocabulary (8,000-word) for the PC in 1988. The NeXT personal computer included a speech recognition capability within its operating system when introduced to market in 1988.

The 1990s began with a demonstration of the first large vocabulary, discrete recognition system for the PC by Dragon Systems. The Macintosh Centris and Quadra models, released in July 1993, incorporated speech recognition into the operating system such that system-level commands such as “File, open” could be understood by the computer. The metaphor was that an individual, to whom the user spoke, was controlling system level options. Commands were prompted

⁷ Continuous natural speech is defined in Chapter 2.2.

⁸ Apricot computers were manufactured by Apricot Computers PLC, a British company which until 1985, was known as Applied Computer Techniques, ACT. The Apricot Portable allowed a vocabulary file of up to 4096 words, however, only 64 words could be held in RAM at one time.

by the use of this individual's name, e.g., "Connie, close file." Text-to-speech⁹ (TTS) functionality was also included to allow highlighted text to be read to the user by the system.

The 1990s witnessed an acceleration in the adoption of personal computers as processing speeds increased and prices dropped. This increase in computational power and memory allowed Dragon Systems to introduce the world's first general purpose large vocabulary continuous speech recognition utility, Dragon *NaturallySpeaking* in 1997. The product was used primarily to dictate text into word processing applications. Some elements of command and control functionality were also available to apply formatting to the dictated text.

The 2000s continued to witness speech recognition refinements in commercial software applications, telephony and the internet. Speech recognition is also being adopted by portable devices such as personal digital assistants (PDA) and cellular phones. These devices require an input modality suited to their small size and limited number of buttons on the keypads. Speech recognition is especially helpful to provide transcription or content creation as well as command and control functionality over the device. Examples of the integration of speech recognition into products are given in Chapter 2.3 - ASR in Application.

⁹ Text-to-Speech, also known as speech synthesis, refers to the process by which plain text is converted into digital audio enabling an application to have meaningful audible responses (See Chapter 2.2).

2.2 ASR Technical Overview

Recognition Methods

The template matching method is used in speech recognition to assess word templates stored in the system against one or more parts of a speech pattern to establish which template best matches the pattern. The enrolment or training completed by a speech recognition user builds templates of their spoken utterances. These templates are then used to compare and match known against future utterances by the user. Dynamic time warping can be used to expand or compress the duration of the speech pattern so that it can be compared to the template. The storage and processing requirements of template matching however can become prohibitive for large vocabularies.

An alternate speech recognition system, *allophone matching*, begins with the basic unit of language, the phoneme. Phonemes describe the sound fragments that the human vocal system produces. Allophones are the sounds of those fragments as actually pronounced. (Rodman, 1999) Allophones are grouped together to form comprehensible words and sentences. When this smallest part of a word is changed, it will alter that word's meaning. For example, the letter "t" is a phoneme and is used as the allophones [t], [t^h] and [D] as well as others, depending on the context.

Spoken recognition systems record spoken utterances and convert them from analogue time-based signals to a set of digital frequencies. The positions of vocal tract formants¹⁰ are extracted from these frequencies and converted to numerical representations. Phonemes are discerned when the numerical representation of formant positions are compared with a set of known values. Real-time extraction uses a process called linear prediction. Once identified the phoneme is processed further by the speech recognition software.

Statistical processes based on Hidden Markov models are used to evaluate multiple hypotheses of the utterances meaning. Based on the sequence in which the phoneme occurs, the model assigns a confidence value to each phoneme, indicating the likelihood that it belongs to a stored word model. A speech recognition system may use language models, and/or acoustic models, and lexicons¹¹ to evaluate the phonemes.

The use of language models allows speech recognition software to identify the relative likelihood of a sequence of words. This is accomplished by comparing a list of phonemes and extracted frequencies with a table of relevant formant positions. This table is constructed from a general model of speech from a variety

¹⁰ A formant defines a vowel sound by its higher amplitudes than those in the surrounding frequencies. Vowel sounds are characterized by the lowest three formants called the first, second and third formant.

¹¹ Lexicons are also commonly referred to as vocabularies.

of native speakers of the language, and therefore may not match the user's particular speech patterns.

An *acoustic model* is a statistical representation of the sound patterns that constitute individual words. It allows the software to predict all the ways a particular word might sound given various pronunciations, cadences, or background interference. Creating a specific acoustic model by training the speech recognition software to a user's own speech patterns increases the accuracy of the system.

Restricting the vocabulary size can also enhance recognition performance. The speech recognizer can use this restriction to limit the range of possibilities it must examine at each point and thereby provide results that are more accurate. An *active vocabulary* is also used to increase recognition accuracy by creating a subset of the entire vocabulary of a speech recognition system to which the recognizer is restricted in certain contexts. In this respect, limiting the use of ASR to command and control functions could greatly enhance performance within a graphical user interface.

Dependent versus Independent Speaker Recognition

Speaker-dependent solutions recognize speech for each speaker's voice and require that users train the software to identify their specific voice profile. This training process is called "enrolment." Speaker-dependant solutions require a voice template to be recorded for each user so that the computer understands the exact nature of the person's dialect, tone, and voice. Speaker-dependent systems have modest processing requirements and are generally considered to be more accurate than speaker-independent technologies.

A disadvantage of speaker-dependent systems is the requirement to store templates for each word or phrase to be recognized. The storage requirements increase in relation to the size of the vocabulary or growth in the number of users. Generally, the accuracy of recognition is proportional to the size of stored templates and number of users. Thus, while processing power can be modest, when a computer is operated by more than one user the processing power can be significant, especially when using ASR for continuous speech dictation in a word processing application.

Speaker-independent solutions recognize speech for multiple users speaking a specific language. Speaker-independent recognition is required for applications in which the identity of the user is not known as in, for example, a bank with an interactive voice response system. Users do not need to undergo any speaker

training; their speech will be recognized immediately. These solutions tend to be less effective where there is a diverse speaker population with regional accents and/or multilingual environments. In the speaker-independent environment, a standard pre-defined lexicon of words is established and users must be aware of which words to say for specific task confirmations and situations.

Discrete versus Continuous Word Recognition

Discrete word recognition refers to an utterance without a pause and may be a single word or multiple words. The pause allows for easy identification of word boundaries. Typically, a pause of at least 250 milliseconds (ms) is required between words or phrases. Continuous speech allows users to speak at their natural pace and rhythm and can recognize words separated by less than 50 ms of silence. Pauses are not required between words or digits. As noted, the strength of continuous speech recognition is best demonstrated when integrated with word processing systems. Users wish to speak normally and naturally and have all utterances understood. Discrete word recognition can be applied when shorter responses such as command and control statements are required.

Speech Recognition versus Voice Recognition

For the purposes of this study, it is important to distinguish between automatic speech recognition (ASR) and automatic voice recognition (AVR). These terms are commonly used interchangeably but have distinctly different meanings.

Automatic speech recognition is the determination of content from a spoken utterance whereas automatic voice recognition is the identification of the speaker of a spoken utterance. Speaker verification is extensively used for security applications. It is more accurate than fingerprinting and second in accuracy only to retinal scans.

Spoken Output

As mentioned previously, it is also important to distinguish between ASR and speech synthesis or text-to-speech. The two types of spoken output available for speech applications are: 1) recorded human speech and, 2) computer-synthesized speech also referred to as text-to-speech. Although there have been recent advances, TTS tends to sound unnatural and does not possess the qualities of natural speech in clarity and prosody. The primary strength of TTS is its ability to dynamically convert any written text to spoken output. In contrast, record human voice requires pre-recorded content that may be concatenated in a limited number of sequences. Most speech applications delivering predictable or fixed output use pre-recorded human speech.

2.3 ASR in Application

Automatic speech recognition is currently integrated in many hardware and software applications. This is directly related to the increase in hardware computational power as well as to advances in ASR technology. The purpose of this section is to highlight how ASR technology has been adapted into various software and hardware solutions across multiple usage domains.

On the PC, dictation is the primary use of ASR in mass-market recognition software applications. Commercial speech recognition software can function as a stand-alone text processor, or can be used within word-processing packages. As noted, ASR software also allows users to apply command and control functions for the dictation software itself and many other applications running on a user's computer. Thus, instead of using the software only as a transcription tool for dictation, ASR can be integrated into the global user interface as an additional input modality.

ASR has a large presence in the medical and legal fields, where dictation of documents and forms is used to enhance productivity. Physicians and other medical professionals are required to maintain rigorously detailed medical forms and records and, as a profession, have been early and enthusiastic adopters of ASR dictation software. Legal professionals also embrace ASR for dictation or

transcription of their voluminous written records.

The use of speech recognition and computer generated speech in telephony applications has also advanced rapidly in the recent past. It is used to interactively disseminate information or provide help support to users in enterprise auto-attendant solutions and automated information dispersal systems. Transactional systems for banking systems using interactive voice response are also in common use.

In the industrial and manufacturing sector, speech recognition is now used in such areas as warehouse distribution centre processing, factory assembly and quality assurance inspection tasks. These implementations tend to be in “hands-busy, eyes-busy” user environments where users may be mobile and the use of an input device such as a keyboard is not feasible. A central speech processing system communicates with the remote users via wireless microphone headsets. Spoken utterances are processed on a central system rather than at the point of origin. One of the first commercial applications of ASR was deployed in 1980 in a New Jersey central post office. Workers routed parcels onto conveyor belts by speaking their zip codes into a microphone. (Rodman, 1999)

The automotive industry has also incorporated limited ASR command and control within automobiles. Speech recognition allows for control over environmental systems, audio entertainment devices, and remote guidance systems. Many use

distinct system sounds or voice output to provide feedback to the user. BMW AG, for example, has incorporated ASR into some of their vehicles, ASR provides the driver with control over communication, navigation, entertainment and climate settings. A text-to-speech feature allows e-mail to be read out as well as short messages and telephone dialling lists. The driver can also instruct the computer navigation system as to the desired destination address by speech command and the system then plots the most suitable course.

Healthcare institutions have also adopted speech recognition to create what are known as “smart operating rooms.” Computer Motion Inc., provides numerous applications utilizing speech control in the medical field. Their Zeus product is an advanced microsurgery robotics suite, consisting of an ergonomic control console with speech recognition input and three table-mounted robotic arms which act as the surgeon's hands and eyes during endoscopic surgery. The Aseop product, an endoscope positioner, is a speech-activated robotic arm that automates endoscope positioning, providing the surgeon with direct control over the view of the internal surgical field. The Hermes control center product is a centralized system that enables the surgeon to use simple voice commands to control a network of smart medical devices in the operating room. The system also employs the use of AVR to restrict the control of specific functions to the person recognized as the surgeon.

Increasingly, personal consumer electronics products and appliances are incorporating speech recognition as an input modality. Some uses include speech remote control for home entertainments systems, automated dialling in cellular phones, and command and control of personal digital assistants. Some examples include:

- Sony Computer Entertainment America has incorporated ASR into their PlayStation 2 Title, SOCOM: U.S. Navy SEALs game. It provides multiple players with the ability to speak with one another, as well as interact through voice recognition with the game itself.
- The Home Automated Living (HAL)¹² company has developed hardware and software to allow a user to control all the systems in their home via speech. It is a scalable control system that may be activated by voice via microphones within the home, telephones within the home after pressing the # key, or remotely via telephone. The system can control individual appliances, lighting, climate, security, and entertainment systems. Control over the entertainment system includes requesting television programming information (via the internet) and scheduling video recording. HAL prompts the user with text-to-speech output.

¹² The product name pays homage to the HAL 9000 voice operated computer made famous in Arthur C. Clarke's book, 2001: A Space Odyssey and the 1968 film of the same name directed by Stanley Kubrick.

- On a smaller scale, the Star Trek Voice Operated Dimmer controls the brightness of any lamp fixture. It is designed to emulate the commands prevalent in the television series of the same name. The lamp brightness is controlled via a speech command i.e., "Computer ... Low," and controller confirms the command with both the sound effects present in the television series and verbal response recorded by Majel Roddenberry, the original voice of the Federation computer in the Star Trek TV series and films. It also functions as a night light.

Figure 1: Star Trek Voice Operated Dimmer Control

2.4 Commercial Vendors

As mentioned in Chapter 2.1, both small companies and large corporations invested heavily in speech recognition advancements and directed their efforts to the commercial market during the 1980s and 90s. These companies include IBM, Philips, Lernout & Hauspie, Kurzweil, Dragon Systems, Microsoft, and Apple. By 1995, Dragon Systems released a discrete word dictation speech recognition software called Dragon Dictate. It was the first such dictation speech recognition technology available to consumers. Within a few months, IBM and Kurzweil followed with the release of their own products, IBM *VoiceType* and Kurzweil *Voice* respectively. In April 1997, Dragon Systems introduced *Dragon NaturallySpeaking*, the first continuous speech dictation software, which was part of its full line of multilingual speech recognition products. The increased adoption of the software by consumers was predicated on the availability of faster microprocessors and increases in RAM¹³ memory.

Consolidation of the Industry

The industry experienced consolidation beginning in 1998 when Lernout & Hauspie purchased Kurzweil. Later, Microsoft formed a partnership with Lernout

¹³ Random access memory (RAM) is the memory that can be read and written to by the microprocessor in a computer system.

& Hauspie that allowed it to use the Lernout & Hauspie speech recognition technology in their systems. In 2000, Lernout & Hauspie acquired Dragon Systems for approximately \$460 million (US). That same year, Philips ceased to market their commercial speech recognition solution, *FreeSpeech*. Philips continued to apply their technology to custom applications for large-scale enterprise deployment.

During 2000, accounting irregularities arose within the Lernout & Hauspie company. The company's value plummeted and their stock was delisted. ScanSoft acquired the speech and language technology assets of Lernout and Hauspie in 2001. This acquisition included the technological property of the former Kurzweil and Dragon Systems companies. ScanSoft also acquired the Philips Speech Processing Telephony and Voice Control business units and the related intellectual property in 2003. Philips retained the Speech Processing Dictation business unit and continues to develop applications for its medical businesses and other professions using dictation systems.

This consolidation has resulted in the reduction of PC commercial dictation software vendors to two: ScanSoft, and IBM. Furthermore, IBM has agreed to distribute its *ViaVoice* product through ScanSoft's marketing channels. The current operating systems developed by Microsoft Inc. (XP) and Apple Computers (OSX) also allow the direct integration of speech recognition engines. This allows for command and control as well as navigation between speech-

enabled software applications. Any software vendor can now utilize the speech recognition engine for free within the operating system for command and control functions. This also means that the user can operate the computer by speech control without having to purchase an external or third party recognition software. End users wishing to dictate text, however are still better served by purchasing the ScanSoft and IBM products due to their increased performance and functionality.

Web and Enterprise ASR Deployment

Interest in the deployment of speech-enabled applications across larger corporate enterprise environments has increased with the presence of solutions developed by IBM and Microsoft.

IBM has been supporting the integration of ASR in web application through its *Websphere* products. The product allows for users to interact with a web site via speech recognition. This includes contacting site support via voice-over-internet protocol (VOIP) in a fashion similar to making a telephone call.

Recently, Microsoft has announced the release of the *Speech Server 2004*, which processes both speech and general telephony activity, and is designed to provide speech-based interactions with PCs and applications. Similar to IBM's *Websphere*, Microsoft has designed the product to support speech-based

deployment across large enterprises, call-centers as well as the World Wide Web.

The commitment and presence of both of these large vendors to the development of ASR applications will continue to increase the visibility of ASR to end users. This presence will be demonstrated across many domains and will expand the use of ASR beyond the current niche markets.

2.5 Summary

Crude automatic speech recognition has evolved from the research laboratory test systems of the 1950s to commercial application in a plethora of digital devices today. Consumers can now purchase ASR software that allows for the efficient dictation of text using continuous speech recognition as well as permitting the user to control the computer and software through speech commands. Additionally, business corporations have effectively employed speech recognition to increase productivity of employees working in “hands-busy” and “eyes-busy” environments.

Through the efforts of large corporations such as IBM and Microsoft, automatic speech recognition will continue to gain user visibility and grow out of the current niche markets. (Deng and Huang, 2004) Through this exposure users will

become more experienced and comfortable interacting with speech recognition interfaces. We will also see within graphical user interfaces the increased presence of speech as an additional or “adjunctive” input modality in conjunction with the keyboard and mouse.

This thesis will further explore past and present attempts at incorporating ASR into software applications to draw insight from their successes and failures. This will include technological hurdles or limitations as well as interface design issues that effect user satisfaction and adoption.

3.0 PRIOR STUDIES AND COMMERCIAL APPLICATION

“Unlike the keyboard, spoken language leaves the hands and eyes free for other tasks, and avoids cluttering the display with menu selection information.” (Hauptmann 1993)

This chapter will review prior studies on how automatic speech recognition may be implemented as an input modality. These studies reflect the use of speech recognition as a sole method of input and also as a supplemental or adjunctive input mode. The research examines the efficacy of these input modes in applications for dictation as well as graphics creation. A review and analysis of commercial implementations of ASR within CAD/CAM software applications is also presented. Observations drawn from the studies will be considered in the design of ASR within a multimodal user interface in Chapter 4.

Automatic speech recognition and gestural input compliment each other when used in concert as demonstrated by Bolt with his famous “Put that there” experiment at Massachusetts Institute of Technology, in 1980¹⁴. Assessments of

14 The experiment illustrated the convergence of speech recognition with position sensing of gestures to move simple shapes across a large screen display. The test reviewed how two modalities could converge to disambiguate and enhance the other. An example of a command would have been, “Put that there.” The command phrase would have been spoken while simultaneously pointing at the desired object and gesturing towards the intended location. (Bolt, 1980)

this combination of speech and gesture input continue to be undertaken.

Initially, speech was explored as a sole mode of input in computer software applications. Subsequently, speech was evaluated as an additional or adjunctive mode of input along with keyboard input, keyboard shortcut or accelerator keys, pointing devices, pen, etc.

Past studies have drawn different conclusions on the efficacy of speech recognition as applied to various tasks, suggesting that the use of ASR in concert with other input modalities needs further study.

Today, most computer graphics applications use only a mouse and a keyboard as input devices. Current speech recognition software, allows for the integration of speech command and control into software applications including CAD and graphics applications. Grafting ASR into an existing user interface, however would not show the potential of speech recognition as effectively as if properly designed into the software application.

3.1 ASR as an Adjunctive Input Mode

Many studies in the early 1980s measured the speed and accuracy of ASR against other input modes such as the keyboard and mouse pointer. These studies include the work of Benbasat (1981), Biermann (1985), and Pooock

(1982). While these studies focused on ASR as a sole input mode, rather than adjunctive input mode, they provided a baseline of research for others to follow with comparative testing of ASR against other modes of input. The following section reviews prior studies of ASR when used as an additional mode of input for both command and control¹⁵ and text entry for a variety of software applications.

Command and Control

Damper and Tranchant simulated experiments in which speech input, abbreviated commands and full command keying were compared while a secondary task was performed. They reported that ASR was not noticeably faster than abbreviated keying (shortcuts) for the primary task, but allowed more secondary information (transcription) to be completed. They felt that if recognizer errors were removed, as opposed to speaker errors, ASR would show superiority in performance over keyboard input, especially for high workload situations involving concurrent tasks. (Damper and Tranchant, 1996)

¹⁵ "Command and control" is one of two modes of ASR application and refers to the control of the computer operating system or application functions. The second mode is dictation, where text is derived from spoken utterances.

Command Activation during Word Processing

The study, *Speech-Activated Versus Mouse-Activated Commands for Word Processing: an Empirical Evaluation*, (Karl, 1993) reviewed the effectiveness of speech input for command activation during word processing. Eighteen speech activated commands were used in the study including, “boldface,” “copy,” “italic,” and “paste.” Word processing was regarded as a “hands-busy” application that would benefit from ASR to activate commands that would primarily be accessed using a mouse. Speech activated commands were found to be faster to execute than mouse activated menu commands. Users experienced similar error rates in both modes. It was also noted that task time decreased with increased user familiarity of the speech interface. Users preferred ASR to mouse activation for commands even though they experienced similar error rates. An unexpected finding of the experiment identified that the issuance of speech commands can interfere with working memory (former referred to as short-term memory) tasks. This competition for cognitive resources is discussed further in Chapter 4.2.

Navigating between Windows

In a study reviewing how ASR can augment window navigation, (Schmandt, 1990), participants used a speech interface to execute navigational tasks that would normally be done with a mouse. The premise underlying the research was

that user's attention should remain focussed on the screen and keyboard and not be distracted by frequent keyboard-to-mouse exchanges. Testing revealed no appreciable difference between speech and mouse in change of focus tasks, but did show that speech added value when moving between partially occluded or completely hidden windows. Speech allowed interaction with windows directly regardless of the degree of exposure. Given the advancements in current computer operating systems, navigating between application windows using keyboard shortcuts may lessen the significance of this study. The pertinent observation in the study identified that users successfully navigated between fully or partially occluded windows by using ASR. The results indirectly support the use of ASR to access menus located deep in a menu hierarchy.

Summary:

The preceding studies support the use of ASR as an adjunctive input mode in particular when used to access commands, navigate through menus and limit pointer movement on the screen. The studies were not performed on graphics creation software but have come to conclusions that may apply. The following studies of ASR use in graphics creation software further explore these conclusions.

3.2 ASR as an Adjunctive Input Mode in CAD and Graphics

Applications

Bolt (1980) and Hauptmann (1989) have both shown that will users naturally provide simultaneous voice and gestural input when a system can support both. Although the studies proved that speech could be combined with other input devices, they did not study the effectiveness of that integration when completing common tasks. Pausch & Leatherby (1991)¹⁶ studied the effectiveness of combining ASR and mouse input in graphics software. The study determined drawing speed increased by an average of 21% when speech input was combined with other modes. Users performed the tasks with the voice recognition software of the day rather than using the “Wizard of Oz”¹⁷ approach. The primary measure was of speed of task completion between the control and experimental groups. The elapsed time included the errors and the time to correct them.

The following section reviews studies of ASR when used as an additional mode of input for CAD and graphics applications. Conclusions drawn from the research relevant to this thesis are also highlighted.

¹⁶ The study is detailed in this section under “Speech used in Commercial Graphics Software”.

¹⁷ “Wizard of Oz” refers to a method of prototyping which involves a user interacting with a computer system which is controlled by another human who simulates the computer system’s response.

Integration with a CAD System

Martin, (1989) was one of the first to experiment with speech recognition as an adjunctive input mode to a mouse and keyboard in a CAD system. In particular, the test found that subjects were more efficient when using speech than when typing in command names. It was observed that the users were also able to maintain their attention to their activities on the screen while issuing speech commands.

Speech used with Graphics

In research performed by the Boeing Company, Salisbury et al, (1990) developed a workstation for the Airborne Warning and Control System (AWACS) that incorporated speech recognition. The goal was to develop a system that would respond to simultaneous spoken and graphical input.

Operator tasks were analysed to discern where better performance might be achieved through the integration of speech input. A 70-word vocabulary and corresponding grammar were identified. The speech recognizer was continuous, speaker dependant, and had a 500 word maximum capacity. Operator feedback to speech misrecognition (noise being interpreted as a command, or misinterpretation of speech) was provided via text-to-speech responses.

The researchers made several conclusions that could apply when designing the integration of ASR within a graphics software application. They are:

- To reduce misrecognition incidences due to environmental noise, the test subjects needed to be able to turn the microphone on and off easily;
- Short commands were used to easily integrate into the operational procedures so that users could perform other tasks while speaking commands;
- TTS feedback phrases should be kept as succinct as possible so as to be less distracting to the operator.

Utility of Speech to Manipulate 3-Dimensional Graphics Objects

Hauptmann and McAviney studied the effectiveness of speech recognition and gesture input in 1993. The goal was to find common features among the gestures¹⁸ of different users when interacting with a computer that could understand language and recognize 3D gestures. Three modes of interaction were used in the test: only gestures, only speech recognition, and gestures combined with speech recognition.

The researchers concluded that any mode by itself was less suitable to complete the tasks than when used in conjunction with another. They also identified that

¹⁸ Study participants were asked to assume that the computer could see their hands and understand their gestures.

small vocabularies can be perfectly adequate for limited domain communication. The latter conclusion confirmed earlier findings by Ford, Weeks, and Chapanis (1980).

The authors' conclusions support and add further design insight for the integration of ASR in a graphics application.¹⁹

Speech used in Commercial Graphics Software

In 1991, Pausch and Leatherby performed a study integrating speech recognition within commercially available graphics software. It compared the time required to create drawings by a control group using a mouse only with an experimental group using discrete voice recognition in combination with the mouse. The test was performed on a Macintosh II system using the graphics programme Claris MacDraw and Voice Navigator XA from Articulate Systems with a head mounted microphone. The authors placed great emphasis on finding the correct tasks to perform in order not to bias the study²⁰.

The experiment results indicated a reduction in task completion time by up to 56%, with an average reduction of more than 21% when speech recognition was used in conjunction with a mouse pointer. The authors attributed this increase in

19 The study is further detailed in Appendix A.

20 The study is further detailed in Appendix A.

efficiency to reduced mouse movement and a reduction in task operations when commands were applied directly to the objects or screen locations. They perceived this “noun-verb”²¹ pairing to allow parallel execution at both the physiological and cognitive levels.

Through the development of the experiment the researchers identified that tailored command grammars/vocabularies could maximize speech recognition efficacy. They also observed a reduction of mouse movement and better retention of focus when a command could be applied directly rather than navigating through the menu. These observations support the inclusion of ASR and guide the design approach taken by this thesis with regard to the integration of speech recognition into graphics software.

Speech and Shortcuts used in Commercial Graphics Software

In a follow up to the 1991 study noted above, Leatherby and Pausch examined whether ASR, when combined with mouse input, increased user efficiency more than using keyboard shortcut keys.(Leatherby and Pausch, 1992) The earlier test did not allow test participants to use keyboard accelerators/shortcut keys whereas this study included shortcut keys in the test regime.

21 Noun-verb refers to a selectable object as the noun and an applicable action as the verb.

The keyboard shortcuts were developed and implemented similar to standard keyboard accelerators. Some command names were modified in order to make them more mnemonic. The commands “Cut,” “Paste,” “Select All,” and “Undo” followed the standard keyboard conventions.

The authors assessed their results by measuring the reduction in the time required to complete a task with speech recognition as compared to with the mouse only. The study results showed that the combined use of ASR and mouse input decreased task completion time by 21.2 % while keyboard shortcuts and mouse input reduced task completion time by only 14.5 %. Therefore, the authors concluded that ASR significantly increased productivity when combined with mouse input, as compared to traditional input modes when used in a graphics software application.

This study clearly identified that the integration of speech recognition as an additional input mode into a graphics software application was beneficial in reducing task completion time. It was also noted that keyboard accelerators which required that one hand be removed from the mouse benefited most from speech input. In addition, when the keystrokes became less obvious or more cumbersome to complete, speech input became increasingly useful. This is a key observation which supports the principles guiding this thesis.

Speech used in a Custom Graphics Software Application

In 1994, Hiyoshi and Shimazu completed research on a multimodal drawing tool that used mouse, keyboard and speech recognition input. (Hiyoshi and Shimazu, 1994) The authors developed a drawing tool that allowed the user to choose the input modality unrestrictedly, express commands on multiple objects simultaneously, and shorten speech inputs by inferring commands by the existing context. In order to implement these capabilities, the authors reviewed the effects of how concurrent commands from separate modalities should be processed for greatest accuracy of command meaning.²²

Upon completion of the test, the authors observed that users experienced the greatest difficulty with speech input due to ambiguous pointing events because the test system did not require the users to explicitly target objects. Their study conclusions identified that the use of context, such as input history or active object information is necessary to correctly bind the speech command to the proper event. They also highlighted the importance of an easy return from an error condition should the rules not determine how to resolve the error condition.

²² The study is further detailed in Appendix A.

Speech used in a Custom Graphics Software Application (2)

In 1995, Nishimoto et al released a paper on their custom multimodal drawing tool that combined speech, mouse, and keyboard input called S-tgif. (Nishimoto et al, 1995) The system was designed to record the motion of the mouse pointer and the time required to complete each task.

The authors proposed three “basic principles of interface” for their design. The first was to use the minimum number of operations and to minimize motion. The second promoted ease in learning operations by making them easy to remember and consistent. The third was to make operations transparent, make it easy to perceive system status and to anticipate responses and results. They wished to provide “perfect feedback” to the user.²³

The results of the study established that the average operation time was reduced by 18% with the use of speech recognition input. The number of command inputs was reduced by 11% and mouse movement was reduced by 47%. The authors noted that cancellation commands were far fewer than the number of speech recognition errors.

The study concluded that the addition of speech recognition as an adjunctive

²³ The study is further detailed in Appendix A.

input mode to graphics applications reduced operation time for a task, by reducing mouse pointer movement and the number of commands, and that speech can be beneficial in learning as well as reducing task difficulty.

Elements of the study relevant to the development of this thesis include:

- To learn commands easily, command names were developed that were easily associated with the functions they performed.
- Keyboard shortcuts could be executed with one hand thereby keeping hold of the mouse pointer in the other.
- To increase the ease of learning the system recognized all words associated with a particular command e.g., “cut,” “delete,” “erase,” and “remove.”
- To enhance system transparency, the status of the speech recognition engine was always visible.

Summary:

The preceding studies support the use of ASR as an adjunctive input mode when used in graphics creation software. The studies also demonstrate that performance enhancements are achieved through ASR integration. Design conclusions and considerations derived from the studies will be synthesized and

incorporated into the thesis Case Study detailed in Chapter 5. The following reviews of ASR application in commercial graphics creation software further explore these observations and conclusions.

3.3 Commercial Speech-Enabled Graphics or CAD/CAM

Applications

To date, few speech-enabled commercial CAD/CAM applications have been developed. Moreover, ASR has not been integrated into any commercial graphics creations software.

During the 1990s, two applications were developed to enable speech recognition in Auto CAD via adjunctive recognition engines. Both products were designed to act as an assistive input mode for users with physical limitations. They differed from dictation systems in that they were designed primarily to assist the end user with controlling the functions of the software. This feature is particularly useful for persons suffering from repetitive strain injuries (RSI) and who suffer discomfort when inputting via a keyboard or mouse. System performance degradation was expected as both the speech recognition programme and AutoCAD were running simultaneously.

The first such product was *IN CUBE Mark II Pro* by Command Corp developed in 1993. IN CUBE produced multiple speech recognition products with Mark II Pro being best suited for use with command-intensive applications like CAD. It operated on any IBM compatible 386, 486 or Pentium system with Windows 3.1 and Windows for Workgroups and provided continuous real-time speech recognition for window navigation and voice macro command. It used the speech template system of recognition with a restricted grammar and as such achieved high recognition accuracy. The user was required to train the templates a minimum of three times before operation. Mark II Pro also permitted multiple commands to be grouped into command macros that could be instantiated through a spoken command. A command could execute up to 64 keystrokes. The user interface was presented as a small dialogue where the user could edit the command list, and control the features of the software.

Figure 2: IN CUBE Command Panel

The second product, *VoiceEZcad*, was developed by Metroplex Voice Computing, Inc., and added speech recognition to AutoCAD versions 13 or 14 using the *Dragon Dictate* speech recognition software. The product was developed to speech control all of the commands within AutoCAD. The company's other products related to math and scientific calculations are targeted specifically for users with physical impairments. The software was never developed for AutoCAD 2000 and subsequent versions as market interest did not warrant this migration. (McClellan, 2004)

The most notable examples of speech recognition embedded directly into commercial CAD programmes were developed by International Microcomputer Software, Inc. (IMSI) in the late 1990's. IMSI was the first company to promote and market its entire line of CAD software as speech-enabled. This included the programmes *Turbo Sketch 4.0*, *Talk & Draw Architect*, and *TurboCAD Professional* (versions 5-7).

The speech enablement could be seen as a function of IMSI having acquired a range of diagramming products from Corel Corporation in September of 1997. Some of these products were already speech-enabled, notably the *CorelCAD Modeler*. Many of the Corel development staff migrated to IMSI with the purchase of the software. Their expertise in this area allowed the company to quickly speech enable the balance of the CAD product suite.

In March of 1998, IMSI released *TurboSketch* and *TurboCAD Pro 5* with the inclusion of speech recognition. The product used ASR technology to recognize virtually all commands by the release of version 7. The speech-enabled menu commands allowed a single spoken command to substitute for multiple menu picks. The company focussed much of their marketing efforts on this innovative use of speech to distinguish their product from the industry-standard products.

Simultaneously to the speech enablement of their CAD product IMSI also marketed *VoiceDirect*, a line of speech recognition software products that embedded technology licensed first from Dragon Systems and later from IBM. VoiceDirect was targeted to users wishing to dictate text into word processing applications.

In June 1999, IMSI restructured and redirected their product development away from slow growth retail products toward its growing Internet business. IMSI subsequently reduced its product line by approximately 75%.

TurboCAD Professional v7 released in 2001, was the last speech-enabled version released by IMSI. This may have been due to a low return on investment and cost of continuing to license the speech recognition software. The introduction of ASR to their CAD products added significant marketing presence at a time when speech recognition was being hyped in many other products. The removal of speech recognition may not have been as much a function of the

efficacy of ASR as poor timing in the corporation's revenue cycle.

Currently, the only CAD/CAM software product being marketed as speech-enabled is thinkdesign by think3 Inc. The application offers full-featured 2D to solid modeling, surfacing, and shape deformation tools. thinkdesign 6.0, released at the end of 2000, was marketed as the first CAD system with a user interface designed to specifically take advantage of speech input. It allows the user to execute commands, enter numbers, and navigate the interface. All commands available within the product are accessible via speech in the current product release Version 8.4.

The thinkdesign interface includes a toolbar that displays the audio level and allows the user to enable and disable the microphone via a button. The microphone button provides feedback if the spoken command is not recognized. The toolbar also has a "Display Box" that presents the name of recognized speech command as well as other option settings.

The product includes a "learning mode" that assists the user to learn the software commands through text-to-speech feedback. As the user selects a tool or function using their mouse, the text-to-speech system speaks the name of the command. This feedback is intended to shorten the speech command learning curve.

The product also employs an “echo or confirmation mode” whereby the software will repeat the name of the command to confirm that it was recognized. The company states that this mode should only be used when the user is wearing headphones so that the system will not try to recognize its own outputs and so that others will not be disturbed. Echo could also affect the user’s speech files should the system be configured to automatically adapt to the speaker’s voice during use.

When spoken commands require parameter input, the parameter input can be given via speech or typing. Dialogue boxes are not used, rather label style input fields are shown over the actual objects on the design canvas or are incorporated into the object design. The software also prohibits a subsequent parameter command to be invoked until the initial command has been completed or cancelled. Nevertheless, global commands that do not require parameter input may be executed before a parameter command is completed.

Figure 3: thinkdesign Mini-Dialogue Input Fields

Summary:

The preceding review of the implementation of ASR as an input mode in commercially released CAD and graphics creation software identified that all attempts were targeted to the CAD market space. Some of the applications were created specifically to assist users with physical limitations whereas others were directed to the general market. The success of the initial products was limited in part by the challenges of early adoption, i.e., recognition accuracy, hardware performance, and user unfamiliarity with a new input mode. The removal of ASR

from the IMSI CAD products reflected a change in corporate focus and should not be interpreted as an indicator of the value of ASR within the applications.

The most recent integration of ASR was found in the thinkdesign product. The product was designed with ASR as a primary component within the application. As a consequence, the user interface supports some design elements that are key to successful ASR integration. Nevertheless, it is difficult to measure the extent to which ASR integration has been successful in the product. Product sales reflect all aspects of the product and are not directly related to the inclusion of ASR. The limited media review and feedback may only reflect the product's minor presence within the market.

One striking observation of the review revealed that there have been no prior attempts at integrating ASR within a commercial graphics creation software application. This further encourages me to assess user interest in ASR as an adjunctive mode of input given the advancements of current technology and design knowledge.

Design observations derived from this review will be synthesized and incorporated into the thesis Case Study detailed in Chapter 5.

3.4 Summary

The studies reviewed in this chapter drew comparisons between speech and other modes of input to test the efficacy of speech input. Though the tests were often mired by poor recognition results, in spite of designing the systems to maximize the current technology, they all suggest that speech is a viable input mode when properly incorporated into an interface. The studies further identified that the use of ASR as an additional input mode allowed users to: 1) maintain focus on the work area, 2) reduce mouse movement on the screen, and 3) reduce task completion time.

The review of commercial attempts at integrating speech recognition into a CAD application identified that earlier adopters of the technology suffered from inherent technological limitations, i.e., recognition accuracy, speed of execution, as well as user inexperience with speech interfaces. The more recent implementation has benefited from technological advancements and has also placed greater emphasis on proper integration of speech recognition within the user interface. Generally speaking, the success of the integration of ASR within a product does not necessarily correlate to the success of the product in the market.

The proposed multimodal input approach of this thesis incorporates speech recognition as an adjunctive mode rather than the primary input mode. My studies will review how speech recognition can be implemented to enhance the user experience taking into consideration lessons learned from prior studies and commercial applications. The design related observations drawn from the chapter will be considered in the case study creation. The observations include:

- Short ASR commands allow users to easily perform other tasks while speaking the commands;
- Small vocabularies can be perfectly adequate for limited domain communication;
- Refine command vocabularies to maximize speech recognition accuracy;
- Tailor the command vocabulary to recognize all words associated with a particular command e.g., “cut,” “erase,” and “remove”;
- Develop command names that are easily associated with the functions they perform;
- Make it easy to perceive system status and to anticipate responses and results;
- Make it ease to turn the microphone on and off to reduce misrecognition incidences due to environmental noise;
- Design to minimize the need to remove the hand from the pointing device.

4.0 SPEECH RECOGNITION AND DESIGN

CONSIDERATIONS

As automatic speech recognition technology matures, users expect increases in recognition accuracy as well as ease of use. The value of continued improvements in speech recognition accuracy diminishes if the user interface is not designed to support users' goals. To ensure speech recognition is an effective modality when included as an adjunctive input mode within a graphical user interface, we must assess how best to support a mixed initiative or multimodal interface²⁴.

It is important to draw a distinction between the integration of ASR into a graphical user interface and ASR applications that are known to use speech for voice user interfaces (SUI or VUI). A SUI interface is best described as the interface designed for interactive voice response systems that are used to complete financial transactions or query bus or train schedules. In these systems the user is led through a path of prompts and responses to achieve their goals. The prompts and responses are structured to limit vocabulary size and help reduce errors.

²⁴ Multimodal refers to using more than one input modality. The term is used inconsistently and commonly refers specifically to modalities of communication between people, such as speech, gestures, handwriting, and gaze.(Suhm et al, 2001)

This chapter will review the design challenges distinct to interfaces that integrate ASR. It will identify key issues that I took into account when designing the prototype. Finally, it will consider elements of previous attempts at ASR integration within graphics software.

4.1 Speech Interface Expectations

Since the release of Stanley Kubrick's film 2001: A Space Odyssey, 35 years ago, ASR technology has made extensive progress. Yet for the foreseeable future, humans will not be actively conversing with sentient computers such as the HAL 9000. The perception that speech recognition implies discussion between a human and computer -- in a similar fashion to HAL or to those portrayed in the television series Star Trek -- will continue to create false expectations for users of currently-available desktop ASR products. (Noyes, 2001, Shneiderman, 2002)

In the late 1990s, shrink-wrapped ASR software was heavily marketed and hyped by the media. (Armstrong, 2003) Purchasers anticipated an easy-to-use product with low error rates. Unfortunately, their experiences did not live up to their expectations. While the software could perform well, it required a quiet environment, suitable hardware (computer and microphone), and extensive user

training. Many users who tried ASR for text dictation, became frustrated with the performance, and subsequently never used the product again.(Karat J., et al, 2000) To ensure the best recognition rates users needed to speak as cleanly and smoothly as if reading text aloud. This is a difficult task for a novice user and/or for someone who is composing his/her thoughts as they speak. Fallout from these poor experiences has been difficult to overcome even though both ASR technology and the supportive hardware have matured greatly.

4.2 ASR Integration

The implementation of ASR in commercial desktop software can be characterized by three strategies. The first can be described as “ASR-enabled.” This type of implementation adds ASR on top of the existing graphical user interface, with features supplied by a 3rd-party application. The ASR enabled application is totally unaware of the presence of the speech recognition engine. The second strategy may be considered as “ASR-aware.” Such implementation includes minor changes to the GUI with the target application being aware of the ASR components. ASR would supplement the existing features within the interface. The third, “ASR-centric,” refers to an application that was designed for speech enablement, not to the exclusion of but in addition to other modes of input. Key features may be designed to be accessible primarily by speech. In this style of implementation, the programme might prompt users via TTS to

facilitate a hands-free/eyes-free interaction. These categorized distinctions are not all-inclusive and many applications fall somewhere along this speech integration continuum.

As mentioned, the predominant use of commercial ASR software on personal computers relates to text dictation, where the user can both input text and control functions of the word processor using a 3rd party application. Despite continuing increases in speech recognition accuracy, users prefer to edit or correct dictated text by using menus, mouse and a keyboard rather than via spoken commands. (Karat C. M., et al, 1999; Halverson et al, 1999) Selecting and modifying text using ASR alone was shown to reduce both user performance and satisfaction. Studies have shown that despite the increase in performance of speech recognition for dictation, the gains were offset by the longer task completion time when editing tasks were completed solely using speech commands. In many instances, cascading errors²⁵ arose when speech commands were used exclusively. When both speech and mouse selection were combined, task completion times reduced and user satisfaction increased.

ASR is most beneficial when used as one of multiple input modes. When used for dictation, it is best when another method of selection is combined with ASR to complete text editing and formatting. When dictation is not the primary task,

²⁵ Cascading errors are defined as multiple attempts to correct the original utterance.

however, ASR can be an effective tool for controlling system commands and software.

The effective integration of ASR for the use of command and control should ensure that commands are succinct and concise to limit unnecessary user cognitive load. Human working memory has a limited capacity and therefore cognitive loading occurs when working memory is burdened with large amounts of information being processed simultaneously. Working memory is used for problem solving and is divided into two areas or buffers that address visio-spatial and phonological tasks. (Jonides, 2002) Speaking consumes phonological cognitive resources and therefore may hinder simultaneous problem solving. (Shneiderman, 2000) Mouse or pointer use consumes visio-spatial resources and when executed simultaneously with speech distributes cognitive load across both buffers or channels. The distribution of tasks across non-competitive channels and reducing cognitive load is important when integrating ASR into multimodal interfaces. (Buxton, 1995)

It is the use of ASR in a graphics creation software environment that is the focus of this thesis. In particular I will be examining the addition of ASR to the user interfaces of several graphics (i.e., non-dictation oriented) software applications.

4.3 Unique Elements of ASR Interface Design

The graphical user interface is a well-established method for interaction between a computer user and the computer itself. GUIs allow direct manipulations to be applied to visual representations of objects and actions. The goal of a good user interface is to ensure ease of use, efficiency, and to be as invisible to the user as possible as they complete their tasks. Over the past 20 years, successful design principles for the GUI have been developed by many members of the computing industry.²⁶ Design guidelines and standards have also evolved for both the Windows and Apple operating system environments and provide the software development community a means to standardize the user experience across products. These guidelines promote interface and interaction consistency for the user within and across software applications. Interface design must also ensure that a product is usable. “Usable” can be defined as achieving a goal in an effective, efficient and satisfying manner within a specific environment. (ISO 9241)

Similarly, the creation of ASR applications has prompted researchers to propose user interface design guidelines for speech integration. Though many of the underlying principles of GUI design may be applied to multimodal ASR

²⁶ For example, Ben Shneiderman proposed the “Eight Golden Rules of Interface Design” that define critical elements for successful interaction design. (Shneiderman, 1998)

application development, many unique development issues require specific attention. One such element is the transient nature of a spoken utterance. In a typical WIMP²⁷ application, the user can immediately see the input or action when a keyboard or pointing device is used. For example, in response to a mouse click a menu appears or an icon state changes. In a similar fashion, a spoken utterance provides feedback based on how it was recognized and interpreted by the ASR system. As with ASR, handwriting or gestural input are also considered to be recognition based. In all recognition based systems, recognition software must interpret the input stream from a user in order to identify its content. With the addition of this software comes the need for accompanying user feedback.

Unlike other interface styles, recognition-based interfaces must manage interpretation errors of a users input. Interface design should address error reduction as well as error handling and display. In a GUI interface the user is immediately aware if they have selected the wrong action or object with their pointing device. With an ASR system by contrast, an error is not always immediately identifiable and can take many forms. ASR errors may be the result of a substitution, injection or rejection by the system.(Biermann et al, 1985)

Substitution errors occur when the recognition software incorrectly interprets the spoken utterance and replaces the intended input with another within its

27 The acronym WIMP stands for Windows, Icons, Mice and Pointing, and it is used to refer to the direct manipulation style of user interface.

vocabulary. An injection error is the insertion of a word from an unintended input signal. This may occur when a user speaks to someone or a sound from the user's environment is interpreted as an utterance. Rejection errors occur when the software is unable to interpret the input signal. The kinds of recognition errors in a given ASR system will be dependent upon the recognition system's accuracy. Generally errors are characterized by a long pause or ambiguous inactivity. Users can be confused as to whether the command was not heard or was heard but could not be recognized or interpreted. Consequently, it is not clear whether the command should be repeated or if the user should wait until the utterance is recognized. By immediately repeating the command the computer may subsequently perform the intended action twice and compound the computer-human interaction breakdown.

As recognition errors are not immediately evident to the user and may be discovered only after the utterance has been processed, the user must be able to monitor and correct the input stream interpretations. (Mayhew, 1992) To ensure user satisfaction with a speech interface, this recognition and disclosure of feedback should be provided as quickly as possible. When a user clicks on a command button in a GUI interface they perceive the action as occurring immediately. Researchers have identified that actions and their subsequent events should take no longer than 70 ms in order to be perceived by the user as co-occurring. (Karat J., et al, 1999) This identifies an ideal threshold for a speech recognition interface to both process and display the input or to alert the user that

an error occurred. A lag of more than 70 ms between the utterance and the interpretation will affect the user's ability to maintain their flow²⁸. In earlier dictation systems this delay between speech and visible feedback in part led to a generalized best practice recommendation that users complete all speech dictation input before revising errors or doing any text formatting i.e., separating dictation from the use of command and control functions.(Shneiderman, 2000)

While speech is a natural and common means of communicating between humans, most people have limited experience speaking with machines. This inexperience in itself presents designers with many challenges on how to clearly identify what commands can be spoken by the user. In interactive voice response systems where no visible interface is present, dialogue structure and lexicon design are crucial to a satisfying user experience. When designing for a multimodal GUI interface, the current commercial speech dictation applications allow the user to invoke a "what can I say" list of commands if the user cannot find cues in the visible interface. For example, Dragon NaturallySpeaking provides a list of sample commands for controlling the microphone which include:

- "Go to sleep/Stop listening"
- "Wake up/Listen to me"
- "Microphone off"

²⁸Alan Cooper defines flow as occurring when a software user "really concentrates on an activity" or is extremely productive when engaged in process oriented tasks such as engineering, design development and writing. (Cooper, 1995)

Generally when using command and control functions within an application, designers strive to mimic the naming conventions seen on the interface as the command phrase required to invoke the associated action. For example, Dragon NaturallySpeaking provides the following commands for controlling Microsoft *Word* and Corel *WordPerfect*:

- “Open a document“
- “Save the file“
- “Save as“
- “Print file“
- “Close the document“

This command and phrase matching reduces the learning curve for novice users and increases the user’s ability to interpret the correct command while exploring the visual interface. (Yankelovich, 1996) It has been noted in prior research that as users gain experience within ASR applications their efficiency with the application is proportional to their level of satisfaction. (Dillon & Norcio, 1997)

I have identified the cursor as a particularly appropriate tool for providing user feedback with respect to rejection errors. It is therefore the potential of ASR combined with cursor feedback that I will examine in greater detail.

4.4 Prototype Design Considerations

After reviewing the prior studies literature, it became evident that little research has been conducted with regard to integrating speech recognition within graphics applications since 1995. This is in contrast to findings indicating that speech can increase user efficacy when integrated into multimodal software applications. (Pausch & Leatherby, 1991) Given the advances in ASR technology, computer processing speed, memory and microphone technology, operating systems as well as improvements to user interface controls and design, it seems appropriate that the integration of ASR within graphics and CAD applications should be revisited.

What I envision, is a speech-enabled multimodal interface allowing users to select tools, control and fill dialogues and manipulate objects using speech while working simultaneously with the keyboard and pointing device. This integration of modalities is supported by the increased sophistication of current software user interfaces. In particular, graphics software applications have increasingly utilized the cursor to provide programme feedback to the user. I intend to extend this use of cursor feedback to present the status of the ASR software to the user.

As part of my assessment of the potential of ASR as an adjunctive input into a user interface, and the use of the cursor to indicate the ASR system state, I have

developed and tested a prototype. The prototype described below incorporated many of the observations provided by the existing studies of ASR integration in multimodal applications. Lessons learned from earlier commercial attempts at integrating ASR into CAD software are also included. The features available within the software applications on which I have chosen to focus in this thesis are representative of those available within their respective markets/industries.

4.4.1 Prototype Constraints

A prototype was created to represent how ASR may function within a graphics application for survey participants who did not have previous experience with ASR. Although the ideal implementation of the proposed prototype would include embedding the ASR technology directly within the graphics software, that approach is beyond the scope of this thesis. Accordingly, the prototype utilized Dragon NaturallySpeaking 6 Professional and was limited to the functionality provided by NaturallySpeaking. The technical implementation of the prototype is detailed further in Chapter 5.3 – Phase One – Prototype Development.

4.4.2 Current Patterns of Use / Software Evolution

As software has evolved from command line input to today's current graphical user interfaces, the power and complexity of graphics and CAD software has increased manifold. The graphical user interface has matured in tandem with this increase in functionality. New technology has afforded software developers and designers to enhance the software and the user interface with many powerful creative design features. For a typical user, many of the most frequently accessed features reside on the visible interface. To access the more advanced features, users must develop their knowledge of where the feature resides deeper within the user interface.

Furthermore, "power users" of these graphics and CAD applications generally modify their software environment to optimise their work-flow. Many use shortcut keys²⁹ to accelerate command invocation and customize the interface to maximize their productivity. Now that large 21" display monitors and multi-monitor PC graphics display cards are less expensive and are becoming more common in the workplace, the professional user may be viewing a single large

²⁹ Shortcut keys (also known as accelerator keys) are keys or key combinations that provide users quick access to functions they perform frequently. A well designed software application will adhere to the standard shortcut keys associations that are incorporated into an operating system. This allows users to interact with different software applications by using the same shortcut functionality mappings. Significant rearrangement or changes of these key assignments add significant difficulty for the user.

display or splitting their attention among multiple displays.³⁰ Inherent in this hardware configuration is an increase in pointer travel distance and time to interact with the software interface³¹.

4.4.3 User Profile

The use cases I tested with the prototype were targeted to graphics or CAD power users and domain experts. These users are proficient with the software and work in a fast paced production environment. They likely have modified the software to reflect their personal preferences and use keyboard shortcuts to optimize their workflow. Users of CAD programmes must routinely input alphanumeric characters to modify objects and insert text.

³⁰ For example, Matrox Graphics Inc., multi-display solutions support the Architecture, Engineering & Construction (AEC) and 3D graphics markets. The Matrox DualHead® and TripleHead graphics cards allow a user to create a contiguous desktop across 2 or 3 monitors.

³¹ A human performance model for reaching tasks, known as Fitts' Law, has been widely used in studying pointing devices and techniques. In 2000, it was officially adopted by ISO 9241-9 as one of the standard frameworks in which to study input devices.

4.4.4 Unique Functionality – One: ASR as an Adjunctive Input

Modality

Through the use of the prototype application I sought to create an ASR enabled graphics application in which productivity would be enhanced by reducing pointer movement as well as hand movement between the pointing device and the keyboard.

To add ASR control to the target applications, verbal shortcuts were implemented through the use of Dragon NaturallySpeaking. The verbal shortcuts allowed an adjunctive input mode to access frequently used commands, embedded menu features and shortcut keys. These commands could control any tool, menu or dialogue within the software interface. The verbal shortcuts were named in alignment with the product feature name as displayed in the tool tip e.g., “line” would invoke the line tool. Numeric input could be placed into any appropriate dialogue box or input field. For example, “50” could be “spoken” into a dialogue box to reduce an objects scale by 50%.

The recognition vocabulary was small in size and limited to alpha-numeric characters and to the words or phrases mapped to the verbal shortcuts. For the purpose of the prototype the command vocabulary was kept small and therefore reduced computer resource use to ensure maximum recognition performance.

Details on the implementation of ASR into the target applications can be found in Chapter 5.3 Phase One – Prototype Development.

4.4.5 Unique Functionality – Two: Enhanced Cursor Feedback

With the advancement of 2D and 3D graphics creation software, the use of the cursor to display information about the state of the object or system has evolved significantly. The cursor or pointer is the graphic representation on the screen of a user's mouse or pointing device. Cursors must have a single pixel designated as the 'hotspot' regardless of their shape. The hotspot is the single point that interacts with the visible interface on the screen.

One of the primary tasks of the cursor is to identify the areas of pliancy on the interface where a user may interact with the software. Pliancy is communicated in three ways: 1) no change to the interface object, 2) a visual change to the interface object, or 3) a visual change to the cursor as it passes over the object. The visual change of the cursor as it passes over the object is also referred to as "active cursor hinting." (Cooper, 1995) "Wait cursor hinting," such as an hourglass, is used to identify that the system is processing or completing a task in the background. In Microsoft Windows operating systems, the hourglass is only shown within the window of the application to which the state applies.

Graphics software applications have adopted the use of many different cursors to convey the tool state to the user. Within current Microsoft Windows operating systems, for example, there are 15 pointing actions defined by different cursors. These include Normal Select, Help Select, Busy, Unavailable, etc., (see Appendix B). As an example, AutoCAD LT 2002 uses 48 distinct cursors and Corel PHOTO-PAINT 11 employs 125 distinct cursors to represent selected tools and tools in different states (see Appendix C). As an example, four PHOTO-PAINT cursors that convey the state of the tool to the user are shown below:

Figure 4: Rectangular Mask Tool Cursor States

Each of the major graphics creation software vendors, i.e., Adobe, Corel, and Macromedia, make extensive use of the cursors to display tool and tool state to the user. Advanced graphics applications such as Corel Draw 11 and Autodesk VIZ 4 each use over 240 different cursors to represent tool sets and states. The use of many different cursors supports the design guideline to present the user

with immediate and appropriate feedback. (Shneiderman, 1998)

With such extensive use of the cursor already being made, then, it may be a challenge to identify the error states of the speech recognition software. Such additional use of the cursor would provide immediate feedback to the user on the start of the ASR utility and would present the feedback at the user's central focus point i.e., the cursor. This enhanced cursor feedback could be presented in various ways including colour change and/or by appending modifiers to the cursor. What I envision, then, constitutes an enhancement to the current implementation of presenting the recognition status in a separate dialogue somewhere in the interface periphery as is normally done in most ASR enabled software.

Dragon NaturallySpeaking, for example, uses the "Results box" to identify the status of a recognition process. This box can be placed anywhere on the screen and can be set to auto-hide when not in use. For example, two Results box states include:

Processing utterance

Unrecognized utterance

Figure 5: Dragon NaturallySpeaking Results Box - Example States

Similarly, the thinkdesign mechanical CAD application places a speech toolbar into the interface when the speech recognition software is in use. The speech toolbar identifies the audio level, controls the microphone and options and provides a display area for commands names and descriptions.

Figure 6: thinkdesign Speech Toolbar

The design and use of enhanced cursor feedback must support computer users who are exposed to substantial information while performing their normal tasks. Exceptional conditions or time sensitive information should be presented so as to attract their attention. The design of the enhanced cursor feedback must also recognize that novice and expert users comprehend meaning with differing levels of display or feedback. Novice users benefit from presentation of greater detail and organization whereas experts may only require subtle highlighting.

(Shneiderman, 1998) My research indicates that changes in colour or the use of blinking are not commonly used in CAD and graphics software cursor feedback. This suggests that this may be an appropriate way to add additional feedback to the cursor.

4.4.6 Observations from Commercial Implementations

As discussed in Chapter 3.3, there has been limited integration of ASR in the user interface within the commercial graphics and CAD software market. A notable exception is the think3 product, *thinkdesign*. The user interface for this product is designed to support the incorporation of ASR. Other than enabling all commands within the interface with ASR, the product has a dedicated toolbar that presents the recognizer and microphone status. Commands are separated into those requiring parameters and those that do not. A parameterized command requires the input of a parameter value or acceptance of the default values before another parameterized command can be invoked. If a parameterized command has not been completed, non-parameterized commands such as “zoom” can still be invoked to alter the image display. This is a unique separation of command types that is understandable programmatically, but may not be intuitive to the user.

The thinkdesign software also has learning mode which helps users learn the command names. As the user selects tools with their pointer from the interface, a TTS output states the tool name. A confirmation mode is also available that confirms the tool selected by ASR. Here the TTS output re-states the name of the tool selected to confirm ASR recognition. The thinkdesign manual strongly recommends that users only invoke the learning and confirmation modes when

wearing headphones so as not to disturb others in their working environment.

This type of feedback was determined to not be relevant to power users and therefore was not included into the prototype.

4.5 Summary

The legacy of HAL 9000 on the common perception and expectations of what a speech interface should be was discussed. These expectations -- plus extremely optimistic advertising statements, technological weakness, and user naïveté with the interface -- bore heavily on users' reaction to the first generation of commercially released, large vocabulary, general purpose, continuous speech dictation software. As a result, the poor experiences of the past continue to hinder the adoption of commercial speech recognition other than in niche market areas.

The ASR integration continuum within multimodal interfaces was presented and elements unique to recognition integration were discussed. Among these elements, fragile error handling currently remains the greatest interface issue for recognition based technologies such as speech. (Oviatt et al, 2001)

Due to technological advances it is now possible to envision an alternate use of ASR in graphics applications i.e., outside of its "traditional" use for dictation in

word processing software. In such applications ASR would be used primarily to invoke commands accessed through keyboard input or menu selection by a pointing device. Other elements unique for multimodal design with ASR integration, such as the use of verbal shortcuts for command and control of the application and the inclusion of cursor based error identification, are key to the success of the implementation.

In the following chapter the development and methodology of the case study is described. This includes the speech enablement of the graphics applications, the creation and execution of the video survey and the results of the video survey.

5.0 CASE STUDY

My research into ASR integration has taken two forms: 1) a review of previous studies and commercially developed products, 2) the creation of a prototype by which two assumptions could be tested.

This case study (or prototype) explores the value of incorporating automatic speech recognition as an adjunctive input mode for graphics and CAD software applications. In addition, the prototype also explores the use of cursor based feedback for speech recognition error state identification.

To derive focused feedback on ASR as a user interface enhancement, it was important to ASR-enable software programmes with which case study participants may be familiar. The prototype was developed whereby Autodesk AutoCAD LT and Corel PHOTO-PAINT were augmented with speech recognition using Dragon NaturallySpeaking to perform specific tasks.

Sample tasks were recorded and presented to the survey participants in video format. Speech recognition and system error state identification scenarios were developed from each of the sample tasks and cursor-based error identification and correction solutions were proposed. These scenarios were presented to the survey participants for evaluation and feedback.

The following sections present, in discursive form, a description of the case study and its results. The case study itself (the prototype developed to gather user feedback) is captured on a CD and is bound to this thesis. It is recommended that readers review the CD as they read through this chapter.

5.1 Study Participants

Familiarity with the software and the tasks was an important criterion in my selection of survey participants. Accordingly, persons with extensive computer experience were recruited to participate. Most ranked themselves as moderate to expert users of either the graphics software or the architectural CAD programme I chose. Few had recent experience with automatic speech recognition other than to control cellular telephone dialling systems.

5.2 Experiment Design

The experiment involved two distinct phases for which separate feedback was solicited. The first phase was the integration of speech recognition into Autodesk AutoCAD LT 2002 and Corel PHOTO-PAINT 11. The second phase was the creation of error state scenarios using video captures of the above. Multiple

versions of enhanced cursor feedback assistance for each error state scenario were edited into the video captures to create the survey scenarios.³²

5.3 Phase One – Prototype Development

A sample task was defined for each of the two software applications to illustrate how automatic speech recognition might be used within the application. The AutoCAD example began with the selection and deletion of a set of elements on the work space. A rectangle was then created using the line tool, interior walls were added using the offset tool, a doorway was defined using the line and trim tool and finally, a circle was drawn, copied and both circles were moved. The undo command, and zoom and pan tools were used at various times throughout the scenario. The task set used multiple tools selected from the toolbar and numerical input via the command line.

The PHOTO-PAINT example involved the creation of a roll-over button for use on a web page. The task set began with opening and defining the size of a new file,

³² The ASR demonstration system was developed using commercially available hardware and software. The system was developed on an Intel Pentium III, 600 MHz processor computer with 784 Mb of RAM memory. The operating system used was Microsoft Windows 2000 Professional. Speech input was captured using an Andrea DA-400 desktop array microphone through a SoundBlaster Live 5.1 audio card. Autodesk AutoCAD LT 2002 and Corel PHOTO-PAINT Version 11 were the graphics software applications that were speech-enabled using ScanSoft's Dragon NaturallySpeaking Pro Version 6.1 speech recognition software. The demonstration prototype was video screen captured using TechSmith's Camtasia Studio 2.0.2 and a Telex H-531 USB headset microphone.

followed by the creation of the button using the rectangle tool, an effect was then applied to the button, text added, the roll-over states were created, and the colour of the button for each state was changed. The task set required multiple tool selections from the toolbar, menu navigation, as well as numerical input into dialogues.

Speech recognition was added into the software applications using Dragon NaturallySpeaking Pro. An empty dictation vocabulary that included a language model but no words was used to lessen RAM and CPU load. The standard speech recognition enrolment training was followed as per the Dragon NaturallySpeaking software setup guide. The Results box feature provided by Dragon NaturallySpeaking was not displayed in either the prototype task examples or the enhanced cursor feedback scenarios. This feature was omitted as the intention of this study was to test the use of the cursor as a feedback mechanism.

Software application commands were speech-enabled by mapping the existing shortcut/accelerator keys to speech commands. Where shortcut keys were not pre-defined, custom shortcut keys were created within each software application. The recognition engine was trained to recognize specific spoken command words and phrases (e.g. "line," "offset," "crop," etc.). Mapping and training were created through the use of Dragon's Command Browser feature. This feature allows single keystrokes, text strings or macros to be mapped to user defined

speech command words or phrases.

The use of a small dictation set with Dragon NaturallySpeaking increased system responsiveness and allowed the recognition software, the graphics application, and the video screen capture software to run simultaneously without a significant decrease in response time. The software used for the video screen captures was *Camtasia Studio*.³³

The sample tasks were developed to illustrate how the software applications could function when ASR was available as an adjunctive input mode. The tasks were presented to the participants to assess this style of ASR enablement and to gather general feedback.

5.4 Phase Two – Error State Scenarios

A series of six error state scenarios were derived from the AutoCAD and PHOTO-PAINT speech-enabled task scenarios. Enhanced cursor feedback for each error state was edited into the Phase One video captures to create the

³³ Note, the Dragon Speech recognition engine and the Camtasia software could not share a single speech input source. This required the use of two microphones using different hardware resources to permit both software applications to access separate hardware resources and capture the speech input signal simultaneously. The recognition engine signal source was provided by an Andrea DA-400 desktop array microphone and a SoundBlaster Live 5.1 audio card and the second input signal source was captured using a Telex H-531 USB headset microphone.

second set of scenarios. The cursor feedback sequences were created in Corel PHOTO-PAINT and edited into the segments of the Phase One screen capture videos using Adobe *Premiere*.

Each of the “enhanced cursor feedback” scenarios allowed for a one second delay prior to displaying any visual feedback. This reflects a hypothesized system design rule that any spoken command that can be processed in one second or less will not provide a visual feedback to the end user. This ensures that the end user will not be unnecessarily prompted by recognition processes that fall within a normal execution period. The normal execution period may be adjustable as a user preference setting for end-users, for example those with computers that have slower computer processing speeds.

5.5 Survey Procedure and Evaluation Criteria

As mentioned, participants with extensive computer experience were recruited to participate in the survey. The survey was developed to display on any Windows-based personal computer with a monitor capable of images at 1024x768 pixel resolution.

The survey was conducted in two parts. First the participants were shown the video presentation of the sample tasks with and without the use of speech

recognition commands. Participants were informed that the focus of the test was on how speech recognition could be integrated and used within the application and not on how quickly the recognition software responded to the spoken command. The video presentation was followed by questions querying the participants as to whether they would use speech recognition while working with their preferred graphics application and how speech recognition could be used in other ways with their preferred graphics software application.

After viewing Phase Two – Error State Scenarios (enhanced cursor feedback) participants were asked which of the sample solutions they preferred and why. It was suggested that the participants not focus their responses on whether the colours or modifiers used were correct but rather that the concepts presented conveyed the appropriate meaning. Their comments were recorded and collated into the results tables (see Chapter 5.7). After all of the scenarios were viewed, participants were queried on this implementation of speech recognition and the value of enhanced cursor feedback to assist users when speech recognition is integrated within a graphics software application. The survey form can be found in Appendix D.

In the following section, I will explain each of the Phase Two scenarios in detail.

5.6 Enhanced Cursor Feedback Scenarios

Scenario One: Processing an input signal

Scenario One presented the speech recognition software processing an input signal to identify its content. After the “Select All” command was spoken within AutoCAD, there was a four second pause before the command was applied. The four versions in this scenario illustrate differing cursor feedback strategies for the end user to identify that the speech recognition engine is processing the input signal.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor (orange) to identify that speech processing was taking place prior to command execution. The third version applied an intermittent colour change to the cursor (orange) every 350 milliseconds. The fourth version did not change the cursor state but appended an animated modifier to the bottom right corner of the cursor. The animated modifier cycled through 3 colours, red, orange and yellow.

Figure 7: Scenario One Cursor State Versions

Scenario Two: Input signal unrecognized

Scenario Two presented the speech recognition software processing an input signal that could not be identified. In this scenario, the “Pan” command was spoken in AutoCAD and the speech recognition software could not identify the spoken utterance until it was repeated.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor (orange) to signal that speech processing had taken place. The cursor colour then changed to red to identify that the speech recognition software could not match the utterance to a known command. The third version applied an intermittent colour change to the cursor (orange then red) every 350 milliseconds. The fourth version applied an intermittent colour change to the cursor (orange then red) every 350 milliseconds

and appended an animated modifier to the bottom right corner of the cursor.

The animated modifier in the form of a question mark, cycled through 2 shades of red.

Figure 8: Scenario Two Cursor State Versions

Scenario Three: Input signal recognized, wait state required

Scenario Three presented the speech recognition software processing an input signal that could be identified. However, after the input signal was identified, the computer required a greater period of time to process the instruction. In this scenario, the “Line” command was spoken in AutoCAD and the speech recognition software identified the command, but there was a six second delay before the command was invoked.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor (orange) to signal that speech processing had taken place throughout the entire six second wait. The third version initially applied a colour change to the cursor (orange) and then added an animated modifier (hourglass). The fourth version first appended an animated modifier to the bottom right corner of the cursor to signal that the speech recognition software was processing the spoken input. The animated modifier then changed to an hourglass to indicate that the command was being processed.

Figure 9: Scenario Three Cursor State Versions

Scenario Four: Input signal recognized, application non-responsive and waiting for command-related information

Scenario Four presented the speech recognition software processing multiple input signals. In this scenario, the “Trim” command was spoken in AutoCAD and invoked by the speech recognition software. After the selection of the items to be trimmed, the speech recognition software required another command to change the Trim tool from the object selection state to the trim objects state. The enhanced cursor feedback being evaluated was applied to the second spoken command in this scenario. This scenario assumed that the recognition software and application software were integrated so that prompting commands could be passed from one application to the other.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor (orange) to indicate that another command or parameter was required. The third version applied an intermittent colour change to the cursor (orange) every 350 milliseconds. The fourth version applied an intermittent colour change to the cursor (orange) every 350 milliseconds and appended a modifier in the form of a text prompt to the bottom right corner of the cursor. The text prompt provided situation-specific feedback.

Figure 10: Scenario Four Cursor State Versions

Scenario Five: Input signal recognized, application non-responsive due to user error

Scenario Five presented the speech recognition software processing multiple input signals. In this scenario, the initial “Create” command spoken in PHOTO-PAINT was not recognized by the speech recognition software. The second command spoken “Create Rollover” was identified by the speech recognition software but could not be invoked due to the state of the application software. After a single object was selected, the “Create Rollover” command was spoken again and was recognized by the speech recognition software as a valid command. The enhanced cursor feedback being evaluated was applied to the second spoken command in this scenario. This scenario assumed that the recognition software and application software were integrated so that prompting commands could be passed from one application to the other.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor (orange) and a modifier to indicate that the command was recognized but could not be processed. The third version applied a colour change to the cursor (orange) and an animated modifier to the bottom right corner to indicate that the command was recognized but could not be processed. The fourth version applied an intermittent colour change to the cursor every 350 milliseconds and appended a modifier in the form of a text prompt to the bottom right corner of the cursor. The text prompt provided situation-specific feedback.

Figure 11: Scenario Five Cursor State Versions

Scenario Six: Input signal recognized, potentially multiple responses

Scenario Six presents the speech recognition software processing multiple input signals. The speech recognition software identified a known command but then could not identify the spoken text input. The unrecognized spoken text was identified on the second attempt.

In this scenario, the initial command, “Create Rollover” was spoken in PHOTO-PAINT and was recognized by the speech recognition software. The second spoken signal was a text string to be placed into an input field. The word “banana” was spoken but was not identified by the speech recognition software. After the word “banana” was spoken a second time, it was recognized by the speech recognition software as a valid text input. The enhanced cursor feedback being evaluated was applied to the text input utterance in this scenario.

While the first version did not provide any cursor based feedback, the second version applied a colour change to the cursor to indicate that the text input utterance was not recognized. The third version applied a colour change to the cursor and an animated modifier to the bottom right corner to indicate that the text input utterance was not recognized. The fourth version applied an intermittent colour change to the cursor until a radial style menu³⁴ prompt was

³⁴ The design and interaction with the menu was patterned on those available within Alias® SketchBook™ Pro software.

displayed. The correct word was selected from the four words presented in the menu. The fifth version presented the system's "best guess" text match and waited for a spoken yes or no acceptance command. The initial best guess text match was declined but the second text match was accepted.

Figure 12: Scenario Six Cursor State Versions

5.7 Survey Results

Using the survey procedure and scenarios described, data were collected from 14 participants. Each subject participated in a single session typically lasting from 45-60 minutes. The surveys were conducted at a location convenient to the participant.

5.7.1 Participant Self-Evaluation on Software Experience

The survey began by having participants rate their familiarity with AutoCAD, PHOTO-PAINT, other graphics software, and speech recognition applications. They ranked themselves on a scale from 0 to 10 where 0 meant that they had never used or had not seen the applications and 10 indicated that they consider themselves 'power users'.

The demographic information gathered on participant familiarity with the software on which the survey was based is reflected in the following figures.

Figure 13: AutoCAD Familiarity

Many participants noted that they were familiar with AutoCAD but had not used the software in the past 2-5 years. As such, they rated their familiarity low on the scale. After viewing the demonstration videos, most commented that it reflected much of the functionality and user interface that they were familiar with. All but two of the participants had some experience with AutoCAD.

Figure 14: PHOTO-PAINT Familiarity

The participants' familiarity with PHOTO-PAINT was heavily distributed on both ends of the expertise ranking scale. They either had little to no exposure or rated themselves as Power Users.

Figure 15: Other CAD or Graphics Familiarity

All but one of the survey participants had experience with other graphics or CAD applications. Curiously, the sole individual who was not familiar with any other graphics or CAD applications ranked himself as a Power User of AutoCAD. Six of the participants had experience with other CAD or CAM products other than AutoCAD, and nine had extensive experience with other graphics applications. Six of the nine had experience with CAD or graphics applications on both the Windows and Apple operating systems.

It is worthwhile noting that all individuals who participated in the study had some familiarity with and were comfortable using either a graphics or CAD software application. Twelve of 14 users rated themselves as expert to power users (7 to 10 on a scale of 10) of CAD or graphics software. Therefore the majority of the study participants matched the skill level of the hypothetical user as proposed in Chapter 4.4.3 – User Profile.

Figure 16: Speech Recognition Familiarity

Although not a prerequisite for the survey, only one participant ranked herself as a power user of automatic speech recognition software. Six participants had no experience with ASR. Of those who ranked themselves two or above, five had experience with Dragon NaturallySpeaking and four were experienced with the speech recognition embedded in the Apple operating system. Generally, their

use of ASR had occurred sometime in the past 5 years.

The survey data can be found in Appendix E.

5.7.2 Enhanced Cursor Feedback Evaluation

The evaluation of enhanced cursor feedback tested my hypothesis that cursor-based feedback is preferable to the separate dialogue box or toolbar as provided by other ASR or ASR-enabled products. As described above, users were presented with a number of variations on enhanced cursor feedback. These scenarios were designed to assess the participants preferred method of cursor feedback when the speech recognition software processed an utterance. The duration of the ASR error or system states³⁵ were extended to ensure that the user would have adequate time to review the cursor modifications. Each scenario presented unique error states which are described below.

Scenario One: Processing an input signal

The first scenario presented the speech recognition software processing an input signal to identify its contents. This system state commonly occurs in commercial

³⁵ For the purpose of the survey, error or system state refers to the ASR condition that requires a notification message be displayed to the user.

ASR software when an utterance is being processed. All participants selected a modification to the cursor in some form. The blinking cursor and colour change examples were preferred by 42.9% and 35.7% of the participants respectively. The balance of the participants chose the animated modifier³⁶.

Figure 17: Scenario One Cursor Feedback Preference

The participants that preferred the cursor changing colours commented that this was enough to identify that the software state had changed and that the ASR software was functioning. Some also commented that the blinking was visually distracting or disturbing. The most common statement made by those that preferred the blinking cursor was that it communicated that something was happening and that the system had not “frozen”. An expert CAD user observed that the modifier could become lost in a complicated CAD drawing and would

³⁶ For the purpose of the survey, a modifier refers to an additional graphic appended a short distance from the right bottom corner of the cursor. In some instances they are static images, in others they are animated.

therefore need to be visually distinctive.

Scenario Two: Input signal unrecognized

The second scenario presented the speech recognition software processing an input signal that could not be identified. This system state commonly occurs in existing ASR software. All participants selected a modification to the cursor in some form. The cursor with an animated modifier, a question mark displaying alternating colours, was preferred by 78.6% of the participants. The cursor colour change and the blinking cursor were chosen by 14.3% and 7.1% of the participants respectively.

Figure 18: Scenario Two Cursor Feedback Preference

The participants who preferred the cursor with the animated modifier expressed

that the modifier was “obvious,” required little interpretation, and clearly indicated what needed to be done. Some expressed that the modifier need not be animated.

A participant that chose the cursor colour change in Scenario 1 expressed that the blinking cursor would be a natural evolution from their choice in the initial scenario. Another participant commented that a text-to-speech prompt would also be beneficial, e.g., “unrecognized command.”

Scenario Three: Input signal recognized, wait state required

The third scenario presented the speech recognition software processing an input signal that was identified but required additional time for the system to execute the command. Outside of this survey, this system state could occur but the ASR software used (Dragon NaturallySpeaking) would not present feedback information. All participants selected a modification to the cursor in some form. The cursor with the animated modifier, an hourglass, was preferred by 64.3% of the participants. The cursor with two animated modifiers and the cursor with colour change were chosen by 21.4% and 14.3% of the participants respectively.

Figure 19: Scenario Three Cursor Feedback Preference

The participants who preferred the cursor with the single animated modifier, the hourglass, expressed that they were familiar with the modifier, and that it clearly indicated what was occurring. Some expressed that the cursor colour change which preceded the modifier clearly identified that the system understood the speech utterance and that the system was subsequently processing the command.

The participants who preferred two modifiers commented that they did so because each modifier clearly identified that it was associated to a different function. A participant who preferred the cursor colour change commented that all error states could be represented by different cursor colours. They believed that once the user understood the meaning of each colour, each error state would be easily identified.

Scenario Four: Input signal recognized, application non-responsive and waiting for command-related information

The fourth scenario presented the speech recognition software processing multiple input signals. The initial spoken command was recognized but a second spoken input was required to complete the command. The enhanced cursor modifier was placed on the second spoken input. This error state would not be presented in commercial ASR software and represents a higher level of integration between ASR software and the ASR enabled application.

The blinking cursor and the blinking cursor with text were preferred equally by 35.7% of the participants. A change to the cursor colour and the cursor without feedback were preferred equally by 14.3% of the participants. The selection of the cursor without enhancement in this scenario was unique in the study. In all other scenarios, the participants choose some form of cursor enhancement, either by colour change, blinking or appended modifier. This may be related to the nature of the system state presented. The system state that required additional user interaction was internal to the CAD application and not related to the performance of the ASR software.

Figure 20: Scenario Four Cursor Feedback Preference

Many of the participants who preferred the blinking cursor without the text modifier remarked that the addition of descriptive text would be beneficial to novice users, but may not be necessary for expert users.

The participants who chose the unmodified cursor commented that a user with experience in the graphics application should be able to identify the error without prompting. They also remarked that the system state was not associated to the ASR software and therefore need not be presented with visual feedback.

The participant's responses supported the concept of the graphics software application status being identified through the ASR cursor feedback mechanism.

Scenario Five: Input signal recognized, application non-responsive due to user error

The fifth scenario presented the speech recognition software processing multiple input signals. The initial spoken command was not recognized. The second spoken command was recognized but could not be completed due to the graphics application state. The error condition within the graphics application was corrected and the second spoken command was repeated and recognized. The enhanced cursor modifier was placed on the second spoken command. This scenario examined the potential of adding the graphics application's help information to the cursor. As with scenario four, the error state was not related to the performance of the ASR software.

The blinking cursor with text was preferred by 85.7% of the participants. A cursor with a modifier, a warning triangle with an exclamation mark, was preferred by 14.3% of the participants.

Figure 21: Scenario Five Cursor Feedback Preference

In contrast to scenario four, the majority of the participants selected the cursor with the assistive text modifier to help resolve the graphics application error state. Many commented that the help-style text appended to the cursor quickly and clearly identified how to resolve the problem. This removed the need to search the product Help documentation to identify the problem and solve a “user error.” This follows an “active” Help approach versus the predominant use of “passive” Help whereby the user must search for additional Help information. (Delisle and Moulin, 2002)

Scenario Six: Input signal recognized, potentially multiple responses

Scenario Six presented the speech recognition software processing multiple input signals. The speech recognition software identified a known command but subsequently could not identify the spoken utterance to fill a text input field. The unrecognized spoken utterance was identified on the second attempt. The enhanced cursor modifier was added to the unrecognized spoken text input utterance. This error condition can arise in current ASR software applications.

This scenario presents two types of error correction approaches: 1) repetition and 2) choice. (Mankoff et al, 2000) A repetition strategy has the user repeat the unrecognized word or phrase. A modified repetition approach was included to assess the participants' acceptance of using speech input to resolve the recognition error. The choice mediation strategy presents a list that the user selects from to determine the correct interpretation of their input.

When using ASR for dictation in word processing software, typically the software will either resolve the utterance to provide its best guess or reject it as unrecognizable. This scenario explored the treatment of a "best guess" recognition and alternate ways to select the intended meaning. Versions 1 to 3 follow the repetition strategy where the initial input could not be recognized. The second input was recognized and accepted as the intended word. Version 4 and 5 followed the choice strategy by presenting alternate methods of selection after

the first misrecognition. Version 4 presented a choice strategy that required selection by means of a different mode, i.e., the pointing device. Version 5 presented a choice strategy that required selection by way of another utterance (yes or no).

The radial menu prompt was preferred by 42.9% of the participants. A blinking cursor with a modifier was preferred by 28.6% of the participants and the Yes/No speech prompt was preferred by 21.4% of the participants. A cursor colour change was preferred by the remaining 7.2% of the participants.

Figure 22: Scenario Six Cursor Feedback Preference

Many participants expressed their preference to have a list of potentially correct words presented visually. They identified that it would be easier and faster to select the correct word from such a list. This supports earlier research

observations that users will naturally switch input modes when they encounter a recognition error. (Oviatt, 1999) Some commented on the ease of use demonstrated by the radial style menu for selecting the correct word. Others would prefer a straight listing similar to a context menu.

A participant who preferred the Yes/No speech prompt proposed a hybrid solution which entailed using a word list that was marked A, B, C, etc., so that the correct word could be chosen without repeating the word itself. The participant felt that this may reduce the potential of second misrecognition of the same word.

Assessed separately, the radial menu was favoured by 42.9% of the participants. Should the versions that required repetition be combined, then 57.1% participants preferred repetition over menu selection. Given the sample size, a clear design preference was not identified and would require further study.

The survey data can be found in Appendix E.

5.7.3 Question Responses and General Comments

The participants answered three Yes/No style questions. Their responses are charted and analysed below.

ASR as a User Interface Enhancement

The questions posed relating to ASR as a user interface enhancement:

Figure 23: Survey Question One

Figure 24: Survey Question Two

As identified above, 93 % of the participants believed speech recognition would enhance the user interface of their preferred graphics applications. The responses are based on the method of ASR integration as presented in the survey. This proposal maps the users most frequently used commands to verbal shortcut keys allowing the user to invoke any tool or dialogue located anywhere in the interface. This also allows for the input of alpha numeric characters in any accepting control field. Based on this description, 100% of the participants believed that verbal shortcut keys were a good implementation of ASR. Some participants suggested that Pareto's Principle³⁷ would apply when mapping commands to verbal shortcuts.

Participants commented that a benefit of ASR enablement would be to reduce the time spent moving their hands between the keyboard and their pointing device. This was of great issue for participants who utilized a graphics tablet and pen. They stated that placing the pen in the holder prior to working on the keyboard and then resuming the pen position on the tablet was very inconvenient and bothersome. The use of ASR to input values as well as execute shortcuts that would normally require their pointer-use hand would add tremendously to their efficiency.

³⁷ Pareto's Principle or the 80/20 Rule states that 20 percent of something is always responsible for 80 percent of the results.

In addition, 50% of the participants commented that they perceived great value in using ASR to input alpha-numeric content. All participants with an expert to power user ranking with AutoCAD stated that inputting alpha-numeric content into a CAD application with ASR would be exceptionally useful.

Concerns were raised by 43% of the participants regarding ASR use in an open concept office. Their comments ranged from attracting attention as the single user of ASR to the potential of degraded ASR performance if ASR were used concurrently by many persons in an office. Some participants also expressed concern over vocal loading³⁸ if they used ASR the entire work day. The use of new microphone technologies to mitigate these issues was not discussed during the survey.

Cursor Feedback to Enhance the Use of ASR

Figure 25: Survey Question Three

³⁸ Vocal loading refers to the stress inflicted on the speech organs when speaking for long periods.

Participants were asked whether they considered enhanced cursor feedback provided enough information to assist users with ASR in a graphics application. The participants endorsed the use of enhanced cursor feedback unanimously. Many commented that when they use graphics creation software their focus remains exclusively on the cursor while performing the majority of their tasks. Having the ASR software feedback elsewhere on the user interface, as noted in Chapter 4.4.5, would not be as convenient and would require them to change their point of focus from the object they are creating to elsewhere on the screen and back. The participants felt that all ASR error conditions could be suitably represented by modifying cursor presentation. Many participants felt that the cursor enhancements presented required refinement and that users should be allowed control over the enhancement display preferences.

It should be noted that although all of the participants endorsed the use of verbal shortcut keys, not all regarded this as the optimal implementation of ASR within a graphics creations application. Other suggested implementations included a fully ASR enabled interface, a conversant interface that included TTS responses, and an ASR driven intelligent design interface where the software would design and the user would be the client.

The survey data can be found in Appendix E.

5.7.4 Results Synopsis & Summary

I have summarized items of note from the survey information presented above into an abbreviated list of highlights.

Participant Self-Evaluation on Software Experience:

- All study participants were familiar with a CAD or graphics software application of which 12 of 14 identified themselves as expert to power users;
- The study participants matched the hypothetical target user profile relating to expert knowledge of the CAD or graphics software domain;
- Half of the participants had some experience with ASR, one user rated herself as an expert.

Cursor Feedback Evaluation:

- Each type of cursor modification i.e., colour change, blinking and appended modifier, utilized was selected as appropriate to convey the intended meaning within one or more of the scenarios;
- A direct correlation was observed between participant selection of increased supplementary information on the cursor and increased

assistance required to resolve the error or system state;

- The clarity or familiarity of an appended modifier led to quicker participant identification of meaning and greater satisfaction;
- User's feedback preferences were not influenced by their expertise with graphics or CAD software;
- Scenarios presenting state modifiers could go unnoticed in a complicated CAD drawing if not presented in a highly visible fashion.

Question Responses and Comments:

- 93 % of the participants believed ASR would enhance the user interface of their preferred graphics applications;
- 100% of the participants believed that the proposed use of verbal shortcut keys was a good implementation of ASR;
- Participants felt that ASR can reduce hand movement between pointing device and keyboard, especially for those using pen tablets;
- Participants perceived value in using ASR to input alpha-numeric content;
- Participants deemed experts in AutoCAD believed that inputting alpha-numeric content into a CAD application with ASR would be exceptionally useful;
- 100% of the participants believed that cursor feedback would provide enough information to assist users with ASR while using a graphics

application;

- Participants commented that the cursor was the focal point of their attention while completing tasks and as such, was the logical place to display ASR related feedback;
- Participants expressed that cursor enhancements should be customizable;
- 43% of the participants expressed concern about using ASR within an open office environment.

5.8 Summary

Two prototypes of ASR-enabled graphic software applications were developed.

The applications demonstrated how speech recognition could be used to adjunctively control elements of the user interface in addition to mouse and keyboard input. The prototype displayed the use of speech to select tools, directly open dialogues deep within the menu topology, and input alpha or numeric characters within dialogue controls. The prototypes were used to create demonstration videos of specific tasks that were presented to survey participants. Additionally, scenarios were developed showing how error state feedback from the speech recognition software could be presented through enhancements to the cursor display. Survey participants reviewed the speech-enabled prototypes and the enhanced cursor feedback error condition scenarios and evaluated and commented on each.

In the following chapter the survey results will be interpreted. Discussion on design refinements and future research directions will be proposed. Lastly, a conclusion will be drawn from an overview of the thesis content.

6. Future Research and Conclusion

“Clearly then, as expected, speech activation of commands will be superior to mouse activation of commands in command-intensive tasks as opposed to typing-intensive tasks.” (Karl et al, 1993)

This thesis has examined the potential of automatic speech recognition to enhance the user experience in command rich applications such as CAD and graphics software. It is believed that ASR would be best used as an adjunctive or supplemental input mode to improve user workflow and interaction with the graphical user interface. The implementation of ASR through the use of verbal shortcuts allowed commands or dialogues within the interface to be accessed using a single word or short phrase. This also allowed placing alpha-numeric input within control fields and other areas of the interface as well. This method of integration supports Shneiderman’s humorous statement that “Speech is the bicycle of user interface design: It is great fun to use and has an important role, but it can carry only a light load.” in reference to ASRs potential in replacing the GUI. (Shneiderman, 1998)

In addition, this thesis studied the use of enhanced cursor feedback as a means of identifying and handling error or system state messages. In most ASR-enabled applications, a dialogue box, which may be placed anywhere within the user

interface, presents ASR recognition related feedback in dictation software.

This style of feedback is not optimal when integrating ASR into CAD or graphics creation software since the focus of the user's attention is the cursor. The proposal to use the cursor to display ASR related information is a natural extension of the use of different cursors to represent the application tools and their states within current CAD or graphics software. Several styles of cursor enhancements were proposed to assess the suitability of the feedback related to the error severity and ensure that the additional feedback was visually distinctive from the other cursor states within the application.

As part of the thesis research, I developed a prototype to demonstrate the adjunctive use of ASR within a task creation workflow. Several versions of this prototype were augmented with enhanced cursor feedback and multiple scenarios were developed to present these proposals to a target group of users who were then surveyed for feedback. The scenarios displayed cursor feedback representing multiple styles of error conditions including common recognition errors, application-based state errors, and text input selection options. The results and interpretation of the survey are summarized.

Observations and insights gained both through the review of prior studies and from user feedback on the prototype are synthesized and presented as recommendations for future research. Finally, an overview and conclusion will be drawn from this study.

6.1 Future Research

6.1.1 Design Refinement & Recommendations

“Future research needs to continue to investigate strategies for developing graceful error handling in multimodal systems.” (Oviatt et al, 2001)

Design refinements were collected through the development of the ASR prototypes and cursor feedback scenarios as well as through discussion with survey participants during the survey. The following recommendations summarize the most beneficial areas for refinement:

1. Investigate a new style of interaction with the interface when using ASR to input alpha-numeric data.
2. Optimize the ASR lexicon for greatest recognition accuracy.
3. Refine the cursor feedback visual style and assess when different styles should be applied.
4. Allow users to customize both cursor feedback styles and ASR preferences as well as make them portable.

I will discuss each of these in greater detail below.

Recommendation 1: Enhanced Input Method

During the creation of the test system that augmented Corel PHOTO-PAINT with speech recognition, it became evident that some control input fields responded differently to speech input. When invoked by a speech command, some controls highlighted the input field and allowed the content to be replaced, whereas others would place the caret in the left position within the field and would not replace the contents. This resulted in some scenarios displaying different methods of field selection and input.

This also prompted the author to conceptualize an alternate method of data input. This method would allow the user to input data by simply positioning the pointer device over a control or input field. The user would identify the selected control by the cursor invoked pliancy. In circumstances where dialogues are presented, this may prove to be a faster method of data field selection and input.

Recommendation 2: ASR Lexicon Definition

This study proposed the use of verbal “shortcut keys” to invoke a limited set of commands. These commands reflected those that would provide the greatest assistance given the user’s workflow and most frequent tasks. Although current recognition accuracy has increased substantially, good design practice ensures that the names given to the speech commands are distinctive and do not sound

too similar to other commands. This helps reduce recognition substitution errors.

Recommendation 3: Refinement of Cursor Feedback

The cursor modifiers presented in this study would benefit from further refinement. Computer users are exposed to substantial information while performing their normal tasks, consequently exceptional conditions or time sensitive information should be presented so as best to attract their attention. The cursor enhancements I presented to users followed some of the data display techniques proposed by Shneiderman (1998). Those included the use of two levels of colour intensity, blinking, and colour blinking. The creation of the cursor enhancements should take into consideration that novice and expert users comprehend meaning with differing levels of display or feedback. Therefore novice users may benefit from greater detail and organization while experts may only require subtle highlighting.

In situations where users required extended assistance, they stated that having Help text appended to the cursor was an exceptional way to convey feedback and instruction. This binding of Help information to the cursor, in an “active” rather than “passive” help system would require extensive integration between the speech recognition software and the target software application. (Delisle and Moulin, 2002).

The cursor enhancement modifiers, i.e., colour change, blinking and appended modifiers, and Help text, could be tested in a similar fashion to those that were presented in this study and are not dependant on a functioning ASR prototype.

Recommendation 4: Preference Portability

Users should be able to select and control the colours and types of cursor modifications to reflect their personal tastes as well as to ensure that the cursor enhancements are presented clearly in the user's software environment. It may also be of benefit to allow the users to save these preferences and export them so that they can transfer them to other systems. This would include any ASR user preferences, customized commands and lexicons or dictionaries.

6.1.2 Creating the Next Test Model

Should this research be continued, in order to move from the proof-of-concept phase to a functional test model, the verbal shortcut commands and enhanced cursor feedback must be more fully integrated into a graphics or CAD application.

Speech enablement can be accomplished through the use of a speech vendors software development kit (SDK). Currently, the Dragon NaturallySpeaking

product is regarded as the most accurate speech recognition product and therefore would be recommended for the test model development. The Dragon NaturallySpeaking SDK allows software developers to speech-enable programmes using the Dragon NaturallySpeaking application programming interface (API).

Although the proposed enhanced cursor feedback examples were created for the survey, the inclusion of a cursor-based modifier is technically feasible in a working system. The enhanced cursor feedback could be created through the use of the Graphics Device Interface³⁹ (GDI) in the Microsoft Windows 2000 operating system or later. The GDI provides a set of functions that allow applications to send graphics commands to a display device such as a cursor.

An example of this integration can be observed in the freeware software Time Cursor⁴⁰ which appends the system time to the cursor and allows the user to modify the color, size and position. The software takes advantage of the transparent forms in the Microsoft GDI available in the operating system.

39 The Graphics Device Interface allows applications to send graphics commands to a display or print device without being concerned about the details of the display or print hardware. A Microsoft Win32 GDI receives rendering requests from applications and passes those requests to the kernel-mode GDI. The kernel-mode GDI passes the rendering requests to a display or printer driver. The kernel-mode GDI also provides services that a display or printer driver can call.

40 Published by Ferdinando Rizzi - http://www.geocities.com/gustave_us/.

Therefore, should this approach be integrated into commercial speech recognition software, the ASR software could append a modifier to the target application cursor when error or system status changes occur without requiring the target application vendor to create and maintain specific cursors for this purpose.

6.1.3 Obstacles to Success

Early commercial attempts to integrate ASR into CAD software applications found limited success. Some were the efforts of smaller organizations that wanted to provide assistance to users with disabilities. Others were aiming to capitalize on the market presence of new technology to sell to early adopters. The fact that these implementations were eventually withdrawn from the market is not a reflection of the benefits of ASR as much as of inopportune timing, or of technological impediments (i.e., ASR accuracy and hardware requirements), or of poor implementation.

Although the thinkdesign software may be successful in a market niche, its integration of ASR has not lead to greater awareness of the benefits of ASR by consumers or the general marketplace. In order to increase consumer adoption, ASR should be incorporated into applications that are considered industry standard and have a higher user visibility (i.e., available at retail outlets). Users

prefer or are required to work with industry standard products and as a result the integration of ASR into industry standard software applications such as AutoCAD and PHOTO-PAINT would assist in promoting the value of ASR integration.

Research has identified that poor recognition performance is the largest obstacle to user adoption. This study has also identified that users perceive speaking aloud in an open office environment as a potential inhibitor to the use of speech recognition in the workplace. This may also include perceptions of user discomfort when wearing traditional headset microphones and being tethered to the computer.

These concerns may be partially assuaged by the use of microphones that are more sensitive and comfortable. For example, array microphones placed on a desk or atop the computer monitor now provide recognition accuracy comparable to USB headset microphones. An array microphone contains a series of microphone elements that apply algorithms to the input signal to create a focused beam directed at the user. The audio input is processed and enhanced by cancelling background noise. This style of microphone allows a user to sit 24 to 36 inches away. Wireless headsets are now also appearing in the marketplace and would provide users with greater comfort and freedom of movement.

Technology continues to produce new means for capturing and inputting audio signals. For example, researchers have used 'microphone domes' that are suspended from the ceiling to capture input signals. (Krahnstoeve et al, 2002)

The National Aeronautics and Space Administration (NASA) recently announced that small, button-sized sensors, affixed under the chin and on either side of the 'Adam's apple,' could gather nerve signals, and process and translate them into words. This subvocal speech is similar to when a person reads to himself.

Recently, Intel Corporation released a product called Audio Visual Speech Recognition (AVSR) technology which gathers both speech and visual information as input into a recognition engine. Combining the information from a variety of inputs greatly enhances the recognition accuracy. This approach is particularly useful when significant background noise is present in offices or other real world environments.(Oviatt, 2000)

6.2 Conclusion

This study reviewed prior research in the area of speech recognition integration within computer software interfaces. Through this review it was discovered that little research in speech enabling CAD or graphics software had been performed since 1995. Curiously, the reduction in CAD or graphics software research coincided with the introduction of commercially available speech dictation

programs. Nevertheless, research has continued in the field of speech recognition -- most notably in the areas of multimodal integration with pen or handwriting for mobile computing and within IVR systems. These studies included the use of command and control as well as data input.

A review of commercial products that integrated ASR identified several CAD/CAM applications. No speech-enabled graphics creation products have been or currently are available on the market. The CAD/CAM products were targeted to both the general consumer market and to persons with physical disabilities. The product most recently released, thinkdesign by think3 Inc., was developed specifically to include ASR as an integral part of the user interface. Currently, this is the only ASR enabled CAD/CAM product available. The fact that earlier products were withdrawn from the marketplace does not invalidate the concept of using ASR within a CAD or graphics software application. The other attempts were withdrawn due to insufficient purchase interest resulting primarily from early adoption performance limitations.

This study proposed to evaluate the potential of integrating ASR within industry-standard CAD and graphics applications with current technology. The target user proposed was a domain professional who would use ASR as an adjunctive means to complete command and control functions within the application. The focus was to enhance the user's productivity by reducing mouse movement within the software user interface as well as hand movement between the mouse

or pointer device and the keyboard. The proposed method of ASR integration was endorsed by 100% of the survey participants with 93% stating that they would use ASR within their preferred graphics application were it available. Some participants expressed concern that noise in the work place might limit the adoption in some environments.

The other component of the study was to assess the potential of providing ASR error identification to the user via enhanced cursor feedback. The concept was to place critical ASR error feedback information directly where the user maintains their focus, on the cursor. ASR error identification and handling was identified throughout the research literature as the primary interface enhancement necessary to increase user satisfaction and acceptance. Currently, commercial (shrink-wrapped⁴¹) ASR products are designed primarily for the dictation of text input. These products provide feedback in a separate dialogue that can be placed anywhere on the user interface. This style of feedback is less than optimal for users of CAD or graphics applications as their point of focus throughout their task completion is the cursor. The use of cursor feedback to provide error and status identification was endorsed by 100% of the study participants. The precise method of cursor enhancement, either by colour change or appended modifiers, will require further study to define the optimal visual cues.

41 'Shrink-wrapped' applies to software available for purchase through commercial retail outlets.

The survey results support the proposed use of ASR and enhanced cursor feedback within CAD or graphics applications and indicate that this proposed proof of concept was successful. The development of a functioning prototype with current technology and a refined set of cursor modifiers would be the logical next step in the evolution of this concept.

BIBLIOGRAPHY

Armstrong, B. "Hype vs. Effective Speech Technology", *Speech Technology Magazine*, January/February 2003.

Benbasat, Izak, Dexter, Albert S., Masulis, Paul S. An Experimental Study of the Human/Computer Interface. *Communications of the ACM*, 24(11), 752-762, 1981.

Biermann, A. W., Rodman, D., Rubin, D., and Heidlage, F. Natural Language with Discrete Speech as a Mode for Human To Machine Communication. *Communications of the ACM*, 28(6), 628-636, 1985.

Bolt, R.A., "Put-That-There: Voice and Gesture at the Graphics Interface", *Computer Graphics* 14, 3, 262-270, 1980.

Buxton, W. Speech, Language & Audition. In *Readings in Human-Computer Interaction: Toward the Year 2000*, R. M. Baecker, J. Grudin, W. A. S. Buxton, & S. Greenberg, Eds. Morgan Kaufmann Publishers, 1995.

Cohen, P.R., Dalrymple, M., Moran, D.B., Pereira, F.C.N., et al, Synergistic Use of Direct Manipulation and Natural Language, *Proceedings of CHI 1988*, Human Factors in Computing Systems, New York: ACM, 1989.

Cooper, A. *About Face: The Essentials of User Interface Design*. Foster City, CA: IDG Books, 1995.

Damper, R. I., Tranchant, M. A. and Lewis, S. M., Speech versus keying in command and control: Effect of concurrent tasking. *International Journal of Human-Computer Studies*, 45, 337-348, 1996.

Davis, H. K., Biddulph, R., Balashek, S. Automatic Recognition of Spoken Digits. *American Journal of Otolaryngology*, 24, 637-642, 1952.

Delisle, S., and Moulin B. User Interfaces and Help Systems: From Helplessness to Intelligent Assistance. *Artificial Intelligence Review*, 18, 117-157, 2002.

Deng, L. and Huang, X. Challenges in Adopting Speech Recognition, *Communications of the ACM*, 47(1), 69-75, 2004.

Dillon, T., Norcio, A. User performance and acceptance of a speech-input interface in a health assessment task. *International Journal of Human-Computer Studies* 47(4), 591-602, 1997.

Ford, W.R., Weeks, G.D. and Chapanis, A. The Effect of Self-imposed Brevity on the Structure of Diadic Communication. *The Journal of Psychology*, 104, 87-103, 1980.

Halverson, C., Horn, D., Karat, C., & Karat, J. (1999). The Beauty of Errors: Patterns of Error Correction in Desktop Speech Systems. *Proceedings of INTERACT '99*, 133-140, 1999.

Hauptmann, A.G. Speech and Gestures for Graphic Image Manipulation. *Proceedings of CHI 1989, Human Factors on Computing Systems*, ACM Press, 241-245, 1989.

Hauptmann, Alexander G., & McAvinney, Paul. Gestures with Speech for Graphic Manipulation. *International Journal of Man-Machine Studies*, 38(2), 231-249, 1993.

Hayes, P.J. Steps towards Integrating Natural Language and Graphical Interaction for Knowledge-Based Systems. *ECAI*, 543-552, 1986.

Hiyoshi, Mayumi, and Shimazu, Hideo. Drawing Pictures with Natural Language and Direct Manipulation. *Proceedings of the 15th International Conference On Computational Linguistics*, Coling, 722-726, 1994.

ISO/IEC 9126. Information technology - software product evaluation - quality characteristics and guidelines for their use. International Organization for Standardization, 1991.

ISO/IEC 9241-9. Ergonomic requirements for office work with visual display terminals (VDTs) - Part 9: Requirements for non-keyboard input devices. International Organization for Standardization, 2000.

Jonides, J., Sylvester, C.-Y. C., Lacey, S .C., Wager, T. D., Nichols, T.E. and Awh, E. Modules of Working Memory. In *Principles of Working Memory*, R. H. Kluwe, G. Luer, and F. Rosler, Eds. Springer-Verlag, New York, 2002.

Karat, C. M., Halverson, C., Horn, D., and Karat, J. Patterns of Entry and Correction in Large Vocabulary Continuous Speech Recognition Systems. *Proceedings of the International Conference on Computer-Human Interaction, Human Factors on Computing Systems*, ACM Press, 568–575, 1999.

Karat, J., Lai, J., Danis, C., Wolfe, C. Speech User Interface Evolution. In *Human Factors and Voice Interactive Systems*, D. Gardner-Bonneau , Ed. Kluwer Academic Publishers, Boston, 1999.

Karat, J., Horn, D.B., Halverson, C.A., and Karat, C. Overcoming Unusability: Developing efficient strategies in speech recognition systems. *Proceedings of the International Conference on Computer-Human Interaction*, The Hague, The Netherlands, 2000.

Karl, Lewis R., Pettey, Michael, and Shneiderman, Ben. Speech-Activated Versus Mouse-Activated Commands for Word Processing: An Empirical Evaluation, *International Journal of Man-Machine Studies*, 39(4), 667-687, 1993.

Krahnstoever, N., Schapira, E., Kettebekov, S., Sharma, R. "Multimodal Human Computer Interaction for Crisis Management Systems", IEEE Workshop on Applications of Computer Vision, Orlando, FL USA, December 3-4, 2002.

Leatherby, James H. and Pausch, Randy. Voice Input as a Replacement for Keyboard Accelerators in a Mouse-Based Graphical Editor: An Empirical Study. *Journal of the American Voice Input/Output Society*, 11(2), July 1992.

Mankoff, J., Hudson, S.E. and Abowd, G.D. Interaction Techniques for Ambiguity Resolution in Recognition-based Interfaces. *Proceedings of UIST '00*, 11-20, 2000.

Martin, G. F. The Utility of Speech Input in User-Computer Interfaces. *International Journal of Man-Machine Studies*, Academic Press Limited, 355 – 375, 1989.

Mayhew, Deborah J. Principles and Guidelines in Software User Interface Design. Englewood Cliffs, NJ: Prentice Hall, 1992.

McClellan, Nancilu. Email correspondence with the author. March 25, 2004.

Nishimoto, T., Shida, N., Kobayashi, T., Shirai, K. Improving Human Interface in Drawing Tool Using Speech, Mouse and Keyboard. *Proceedings of 4th IEEE International Workshop on Robot and Human Communication*, ROMAN95, 107-112, 1995.

Noyes, J. Talking and writing-how natural in human-machine interaction? *International Journal of Human-Computer Studies*, 55, 503-519, 2001.

Oviatt, S., MacEachern, M., and Levow, G. Predicting Hyperarticulate Speech During Human-Computer Error Resolution. *Speech Communication*, 24, 87–110, 1998.

Oviatt, S. Mutual Disambiguation of Recognition Errors in a Multimodal Architecture. *Proceedings of the International Conference on Computer-Human Interaction*, Human Factors on Computing Systems, ACM Press, 576–583, 1999.

Oviatt, S. Taming recognition errors with a multimodal interface. *Communications of the ACM*, 43(9), 45-51, 2000.

Oviatt, S., Cohen, P., Wu, L., Vergo, J., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J., and Ferro, D. Designing the User Interface for Multimodal Speech and Pen-Based Gesture Applications: State-of-the-Art Systems and Future Research Directions. *Human Computer Interaction in the New Millennium*, J. Carroll, Ed. Addison-Wesley Press, Boston, 2001.

Pausch, Randy and Leatherby, James H. An Empirical Study: Adding Voice Input to a Graphical Editor. *Journal of the American Voice Input/Output Society*, 9(2), 55-66, July 1991.

Poock, G. K. Voice Recognition Boosts Command Terminal Throughput. *Speech Technology* 1, 36 – 39, 1982.

Rodman, Robert D. *Computer Speech Technology*. Artech House, Norwood, USA 1999.

Salisbury, M. W., Hendrickson, J. H., Lammers, T. L., Fu, C., Moody, S. A. Talk and Draw: Bundling Speech and Graphics. *IEEE Computer*, 23(8), 59-65, August 1990.

Schmandt, C., Ackerman, M. S., Hindus, D. Augmenting a Window System with Speech Input. *IEEE Computer*, 23(8), 50-58, August 1990.

Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Reading, MA: Addison Wesley, 1998.

Shneiderman, B. The Limits of Speech Recognition. *Communications of the ACM*, 43(9), 63-65, 2000.

Shneiderman, B. *Leonardo's Laptop: Human Needs and the New Computer Technologies*. MIT Press, 2002.

Suhm, B., Myers, B. and Waibel, A. (2001). Multimodal Error Correction for Speech User Interfaces. *ACM Transactions on Computer-Human Interaction*, 8(1), 60-98.

Yankelovich, N. How Do Users Know What to Say? *ACM Interactions*, 3(6), November/December 1996.

APPENDICES

Appendix A – Chapter 3.2 Extended Reviews

Appendix A contains extended reviews of studies included within Chapter 3.2 - ASR as an Adjunctive Input Mode in CAD and Graphics Applications.

Utility of Speech to Manipulate 3-Dimensional Graphics Objects

Hauptmann and McAvinney studied the effectiveness of speech recognition and gesture input in 1993. The goal was to find common features among the gestures⁴² of different users and classes of users when interacting with a computer that could understand language and recognize 3D gestures. Three levels of users were studied: the computer-naïve, those moderately familiar with computers, and those proficient in spatial reasoning i.e., designers, machinists, or architects. The third group of users was selected to discern whether there were common characteristics shared among “spatial thinkers”.

The authors noted that although the keyboard is the most common means of text input, it does not permit simple, intuitive manipulation of graphics objects. They perceived the mouse as “a significant improvement over the keyboard in dealing with graphics objects” but still considered its use a disadvantage by adding a 3rd focus of attention to the user. (Hauptmann 1993)

⁴² Study participants were asked to assume that the computer could see their hands and understand their gestures.

Three modes of interaction were used in the test: only gestures, only speech recognition, and gestural combined with speech recognition. Participants were tested in seven operations grouped into three areas, rotation, translation, and scaling.

The test measured candidate spoken utterances against the following communication classifications:

- Number of words used in describing the intended action;
- Number of words used in non-goal-directed phrases;
- Number of lexical items (different words used);
- The words used for goal-related tasks only;
- Syntactic structure of the utterance;
- Sufficiency of the speech act to complete the command.

Test observations proved that there were no appreciable differences in the variables analysed between the computer literacy groups and that almost all utterances were in the form of imperatives. The studies conclusion confirmed the findings earlier findings by Ford, Weeks, and Chapanis (1980) that small vocabularies can be perfectly adequate for limited domain communication.

A high degree of commonality was observed among subjects in the use of hand gestures and speech. The results further support the belief that any mode by itself was less suitable to complete the tasks than when used in conjunction with another. Evidence was also presented that both speech and gestures capture complementary information which assist in disambiguating the meaning of each other.

The authors concluded that “much of the evidence argues for a combination of input modes. Subjects preferred speech and gestures as an ideal combination, both in actual use and in stated preference.”

Speech used in Commercial Graphics Software

In 1991, Pausch and Leatherby performed a study integrating speech recognition within a commercially available graphics software. It compared the time required to create drawings by a control group using a mouse only with an experimental group using discrete voice recognition in combination with the mouse. The test was performed on a Macintosh II system using the graphics programme Claris MacDraw and Voice Navigator XA from Articulate Systems with a head mounted microphone. The authors placed great emphasis on finding the correct tasks to perform in order not to bias the study.

Nineteen voice commands were used in the study. Each test participant was required to train the recognition engine in the available commands. Brief training was given on how to use MacDraw to create objects with and without speech input. The experiment facilitator manually logged the number of speech commands used and the errors encountered by the test subjects using speech recognition. The keyboard was used for text entry, and the speech recognizer was dedicated to command use.

The pilot study highlighted three areas of the test that required modification. It was determined that the command "Line" was misrecognized frequently but "Line Segment" greatly increased the recognition rate. This reflected the need to tailor the command vocabulary to maximize speech recognition efficacy. In addition, the speech recognition software was originally designed to emulate mouse motion and button presses. This was revised so that commands were directly sent to MacDraw without requiring that the cursor move to the associated command menu.

Finally, the original test required the user to navigate the menus using speech recognition in order to select a command. Initially, this required the users to learn the command name as well as its location in the menu. This was revised to allow for direct application of the command when spoken rather than navigating through each menu level via speech commands. This change also resulted in reduced mouse movement and better retention of focus. Each of these changes

reflect the current approach to the speech recognition integration.

The experimental data indicated a reduction in task completion time by up to 56%, with an average reduction of more than 21% when speech recognition was used in conjunction with a mouse pointer. The authors attributed this increase in efficiency to reduced mouse movement and a reduction in task operations when commands were applied directly to the objects or screen locations. They perceived this “noun-verb” pairing to allow parallel execution at both the physiological and cognitive levels.

The authors also state that the benefits of speech integration in a graphics programme are greater than those of keyboard shortcuts or tear-away menus. Keyboard shortcuts allow parallel input via two hands but they must be memorized. Tear away menus reduce travel time but have two drawbacks, they obscure a portion of the working area, and force a mouse context switch. “Voice provides the advantages of parallel input without requiring user memorization or consuming screen space.” (Pausch and Leatherby, 1991)

They postulated that with a system designed explicitly for dual mode input, a minimum of 21 % time enhancement should be considered expected.

Speech and Shortcuts used in Commercial Graphics Software

In 1992, Leatherby and Pausch examined whether ASR, when combined with mouse input, increased user efficiency more than using keyboard shortcut keys. This was a follow-up study to their earlier work of adding speech recognition input to a graphics image editor which did not include shortcut keys (Pausch and Leatherby, 1991). That study concluded that using ASR in conjunction with mouse input, significantly reduced task completion time.

The authors hypothesized that the increase in efficiency was related to reduced mouse travel required to access menus. The earlier test did not allow test participants to use keyboard accelerators/shortcut keys. This study included shortcut keys in the test regime. Two groups of users were tested: a novice group that did not memorize the keyboard shortcuts (and were allowed a “cheat sheet”) and advanced users who had memorized the 16 keyboard shortcuts available. Novice users were able to learn these keys throughout the completion of the task. None of the participants were expert users of the test programme, MacDraw. They were timed during the creation of four drawings after one practice session. Random drawings from magazines were chosen rather than being created by the study authors in order to avoid biasing the task. The drawings were identical to those used in their earlier study.

Keyboard shortcuts were developed and implemented similar to standard keyboard accelerators. Some command names were modified in order to make them more mnemonic. The commands “Cut”, “Paste”, “Select All”, and “Undo” followed the operating system standard keyboard conventions. It was also noted that keyboard accelerators that required one hand be removed from the mouse benefited most from speech input. In addition, when the keystrokes became less obvious or more cumbersome to complete, speech input became increasingly useful.

The authors assessed their results by measuring the reduction in the time required to complete a task with speech recognition as compared with the mouse only. The study results showed that combined ASR and mouse input decreased task completion time by 21.2 percent while keyboard shortcuts and mouse input reduced task completion time by only 14.5 percent. A reduced performance was observed in the novice group who were not familiar with the keyboard shortcuts and required the assistance of a ‘cheat-sheet’. Their task completion time was reduced by only 9.9 percent. Therefore, Pausch and Leatherby concluded that ASR significantly increases productivity when combined with mouse input as compared to traditional input modes when used in a graphics software application.

Speech used in a Custom Graphics Software Application

In 1994, Hiyoshi and Shimazu completed research on a multimodal drawing tool that used mouse, keyboard and speech recognition input. (Hiyoshi and Shimazu, 1994) They observed a number of problematic issues when creating graphics using only a mouse and keyboard. These issues include:

- Focus on the working canvas is lost when accessing a menu by using the cursor;
- Applications with increased functionality have deep menus which in turn become cumbersome to memorize and navigate;
- It is troublesome to move one's hand between mouse and keyboard to provide, for example, numerical input;
- The selection of objects when using the mouse allows for pointing inaccuracies.

The authors hoped to validate the use of speech recognition to remediate the first three of the above-noted problems. The authors also tried to overcome the remaining issues by developing a drawing tool that allowed the user to choose the input modality unrestrictedly, express commands on multiple objects simultaneously, and shorten speech inputs by inferring commands by the existing context.

In instances when user's pointings were ambiguous and the object was not selected, the system selected the nearest object that matched the users' command utterance. This intelligent selection helps distinguish the command target from the other objects and created a "snap-to" effect.

In order to implement these capabilities, the authors reviewed the effects of how concurrent commands from separate modalities should be processed for greatest accuracy of command meaning. They deemed the order of command execution as insufficient. (Hayes 1987, Cohen 1989) They determined that the system should be imbued with the ability to make intelligent object selections. The system must match pointing inputs with speech inputs in a way other than by anaphora, which is the repetition of a word or phrase preceding a speech command. This meant that the pointing event should be handled separately from the speech event. When the pointing event is kept separate, the pointing inputs that occur during a speech utterance are kept in a queue until the system determines which pointing input should be bound to the phrase. Should the pointing input be missing then three possible causes are considered: the pointing occurred before speech input; an anaphoric reference was made before the speech input; the pointing event will occur after the voice input.

Upon completion of the test, the authors observed that users experienced the greatest difficulty with speech input due to ambiguous pointing events because the test system did not require the users to select menus or explicitly target

objects. They also highlighted the importance of an easy return from an error condition should the rules not determine how to resolve the error condition. Their study conclusions identified that the use of context, such as input history or active object information is necessary to correctly bind the speech command to the proper event.

Speech used in a Custom Graphics Software Application (2)

In 1995, Nishimoto et al released a paper on their custom multimodal drawing tool that combined speech, mouse, and keyboard input called S-tgif. The system was designed to record the motion of the mouse pointer and the time required to complete each task. The test subjects were not familiar with the programme prior to the test, but given 15 minutes to practice freely. An available word list and shortcuts associated with speech commands were visible in a separate window throughout the duration of the test. The system used a stationary microphone programmed to reject user utterances when speech volume was low. This allowed users to speak away from the microphone and make non-command related comments without creating false speech recognition rejections. To learn commands easily, command names were developed that were easily associated with the functions they performed.

The authors proposed three “basic principles of interface” for their design. The first was to use the minimum number operations and to minimize motion. The

second promotes ease in learning operations by making them easy to remember and consistent. The third was to make operations transparent: make it easy to perceive system status and to anticipate responses and results. They wished to provide “perfect feedback” to the user.

The authors also proposed three “organization principles of interface” for speech recognition integration. The first was to provide sufficient support for novice users. The second was to treat frequent users with priority, by tailoring the design to the highest use user group. The third was to provide a “guidance mechanism” or help to assist beginners in increasing their knowledge of the application and its use.

To follow the above-noted principles, the system minimized operations by limiting mode changes to speech input only and by executing the commands directly without invoking menus. In addition, keyboard shortcut keys could be executed with one hand thereby keeping hold of the mouse pointer in the other. To increase the ease of learning the system recognized all words associated with a particular command e.g., “cut”, “delete”, “erase”, and “remove”. To enhance system transparency, the status of the speech recognition engine was always visible and acceptances of command phrases were identified with a system beep. Tool specific cursor states were also applied. Users were provided guidance through a persistent display of the available keyboard shortcuts, speech commands and abbreviations.

The results of the study established that the average operation time was reduced by 18% with the use of speech recognition input. The number of command inputs was reduced by 11% and mouse movement was reduced by 47%. The authors noted that cancellation commands were far fewer than number of speech recognition errors.

As a result of the study, the authors concluded that the addition of speech recognition as an adjunctive input mode to graphics applications reduced operation time for a task, by reducing mouse pointer movement and the number of commands, and that speech can be beneficial in learning and reducing task difficulty.

Appendix B – Cursor States

Default Cursor States - Microsoft Windows Operating System

Each pointer shape is a visual cue that indicates what the operating system is doing or what you can do in a given context.

Normal Select
Help Select
Working In Background
Busy
Precision Select
Text Select
Handwriting
Unavailable
Vertical Resize
Horizontal Resize
Diagonal Resize 1
Diagonal Resize 2
Move
Alternate Select
Link Select

Appendix B – Cursor List

Cursor List: Autodesk AutoCAD LT 2002 (48 cursors)

AcIt 3 (#44, 2 formats)	AcIt 4 (#45, 2 formats)	AcIt 5 (#46, 2 formats)	AcIt 10 (#51, 2 formats)	AcIt 11 (#52, 2 formats)	AcItres 1 (#17, 1 format)

AcItres 2 (#18, 1 format)	AcItres 3 (#19, 1 format)	AcItres 4 (#20, 1 format)	AcItres 5 (#21, 1 format)	AcItres 6 (#22, 1 format)	AcItres 7 (#23, 1 format)

AcItres 10 (#26, 1 format)	AcItres 11 (#27, 1 format)	AcItres 12 (#28, 1 format)	AcItres 13 (#29, 1 format)	AcItres 14 (#30, 1 format)	AcItres 15 (#31, 1 format)

AcItres 16 (#32, 1 format)	AcItres 21 (#37, 1 format)	AcItres 22 (#38, 1 format)	AcItres 23 (#39, 1 format)	AcItres 24 (#40, 1 format)	AcItres 25 (#41, 1 format)

AcItres 26 (#53, 1 format)	AcItres 27 (#54, 1 format)	AcItres 28 (#55, 1 format)	AcItres 30 (#57, 1 format)	AcItres 31 (#58, 1 format)	AcItres 32 (#59, 1 format)

AcItres 38 (#65, 2 formats)	AcItres 39 (#66, 2 formats)	AcItres 40 (#67, 2 formats)	AcItres 41 (#68, 2 formats)	Gridres 1 (#1, 2 formats)	Gridres 2 (#2, 2 formats)

Gridres 3 (#3, 1 format)	Gridres 4 (#4, 1 format)	Gridres 5 (#5, 1 format)	Gridres 6 (#6, 2 formats)	Gridres 7 (#7, 2 formats)	Gridres 8 (#8, 2 formats)

Gridres 9
(#9, 1 format)

Gridres 10
(#10, 1 format)

Gridres 11
(#11, 1 format)

Gridres 12
(#12, 1 format)

Gridres 15
(#15, 1 format)

Gridres 16
(#16, 2 formats)

Cursor List: Corel PHOTO-PAINT (125 cursors)

PPIntl 1 (#1, 1 format)	PPIntl 2 (#2, 1 format)	PPIntl 3 (#3, 1 format)	PPIntl 4 (#4, 1 format)	PPIntl 5 (#5, 1 format)	PPIntl 6 (#6, 1 format)

PPIntl 7 (#7, 1 format)	PPIntl 8 (#8, 1 format)	PPIntl 9 (#9, 1 format)	PPIntl 10 (#10, 1 format)	PPIntl 11 (#11, 1 format)	PPIntl 12 (#12, 1 format)

PPIntl 13 (#13, 1 format)	PPIntl 14 (#14, 1 format)	PPIntl 15 (#15, 1 format)	PPIntl 16 (#16, 1 format)	PPIntl 17 (#17, 1 format)	PPIntl 18 (#18, 1 format)

PPIntl 19 (#19, 1 format)	PPIntl 20 (#20, 1 format)	PPIntl 21 (#21, 1 format)	PPIntl 22 (#22, 1 format)	PPIntl 23 (#23, 1 format)	PPIntl 24 (#24, 1 format)

PPIntl 25 (#25, 1 format)	PPIntl 26 (#26, 1 format)	PPIntl 27 (#27, 1 format)	PPIntl 28 (#28, 1 format)	PPIntl 29 (#29, 1 format)	PPIntl 30 (#30, 1 format)

PPIntl 31 (#31, 1 format)	PPIntl 32 (#32, 1 format)	PPIntl 33 (#33, 1 format)	PPIntl 34 (#34, 1 format)	PPIntl 35 (#35, 1 format)	PPIntl 36 (#36, 2 formats)

PPIntl 37 (#37, 1 format)	PPIntl 38 (#38, 1 format)	PPIntl 39 (#39, 1 format)	PPIntl 40 (#40, 1 format)	PPIntl 41 (#41, 1 format)	PPIntl 42 (#42, 1 format)

PPIntl 43
(#43, 1 format)

PPIntl 44
(#44, 1 format)

PPIntl 45
(#45, 1 format)

PPIntl 46
(#46, 1 format)

PPIntl 47
(#47, 1 format)

PPIntl 48
(#48, 1 format)

PPIntl 49
(#49, 1 format)

PPIntl 50
(#50, 1 format)

PPIntl 51
(#51, 1 format)

PPIntl 52
(#52, 1 format)

PPIntl 53
(#53, 1 format)

PPIntl 54
(#54, 1 format)

PPIntl 55
(#55, 1 format)

PPIntl 56
(#56, 1 format)

PPIntl 57
(#57, 1 format)

PPIntl 58
(#58, 1 format)

PPIntl 59
(#59, 1 format)

PPIntl 60
(#60, 1 format)

PPIntl 61
(#61, 1 format)

PPIntl 62
(#62, 1 format)

PPIntl 63
(#63, 1 format)

PPIntl 64
(#64, 1 format)

PPIntl 65
(#65, 1 format)

PPIntl 66
(#66, 1 format)

PPIntl 67
(#67, 1 format)

PPIntl 68
(#68, 1 format)

PPIntl 69
(#69, 1 format)

PPIntl 70
(#70, 1 format)

PPIntl 71
(#71, 1 format)

PPIntl 72
(#72, 1 format)

PPIntl 73
(#73, 1 format)

PPIntl 74
(#74, 1 format)

PPIntl 75
(#75, 1 format)

PPIntl 76
(#76, 1 format)

PPIntl 77
(#77, 2 formats)

PPIntl 78
(#78, 2 formats)

PPIntl 79
(#79, 2 formats)

PPIntl 80
(#80, 2 formats)

PPIntl 81
(#81, 2 formats)

PPIntl 82
(#82, 2 formats)

PPIntl 83
(#83, 1 format)

PPIntl 84
(#84, 2 formats)

PPIntl 85
(#85, 1 format)

PPIntl 86
(#86, 1 format)

PPIntl 87
(#87, 1 format)

PPIntl 88
(#88, 1 format)

PPIntl 89
(#89, 1 format)

PPIntl 90
(#90, 1 format)

PPIntl 91
(#91, 1 format)

PPIntl 92
(#92, 1 format)

PPIntl 93
(#93, 1 format)

PPIntl 94
(#94, 1 format)

PPIntl 95
(#95, 1 format)

PPIntl 96
(#96, 1 format)

PPIntl 97
(#97, 1 format)

PPIntl 98
(#98, 2 formats)

PPIntl 99
(#99, 2 formats)

PPIntl 100
(#100, 2 formats)

PPIntl 101
(#101, 2 formats)

PPIntl 102
(#102, 2 formats)

PPIntl 103
(#103, 1 format)

PPIntl 104
(#104, 2 formats)

PPIntl 105
(#105, 2 formats)

PPIntl 106
(#106, 2 formats)

PPIntl 107
(#107, 2 formats)

PPIntl 108
(#108, 2 formats)

PPIntl 109
(#109, 2 formats)

PPIntl 110
(#110, 2 formats)

PPIntl 111
(#111, 2 formats)

PPIntl 112
(#112, 2 formats)

PPIntl 113
(#113, 2 formats)

PPIntl 114
(#114, 2 formats)

PPIntl 115
(#115, 2 formats)

PPIntl 116
(#116, 2 formats)

PPIntl 117
(#117, 2 formats)

PPIntl 118
(#118, 2 formats)

PPIntl 119
(#119, 2 formats)

PPIntl 120
(#120, 2 formats)

PPIntl 121
(#121, 2 formats)

PPIntl 122
(#122, 2 formats)

PPIntl 123
(#123, 2 formats)

PPIntl 124
(#124, 2 formats)

PPIntl 125
(#125, 2 formats)

Appendix D – Survey Form

Survey Questionnaire

Last name: _____
 First name: _____
 Job title: _____
 Date: _____

Please indicate if you have a working knowledge of the following products and rate your expertise with them:

(0 means 'you have never seen a demo or used this product' and 10 means 'you are a power user'):

	0	1	2	3	4	5	6	7	8	9	10
Autodesk AutoCAD											
Corel PHOTO-PAINT											
Other Graphics/CAD Applications											
Speech Recognition Software											

Comments:

Introduction

The intent of this survey is to illustrate how speech recognition may be introduced as an adjunctive input mode for graphics based software applications such as Autodesk AutoCAD and Corel PHOTO-PAINT and to obtain your opinion on how enhanced cursor feedback may assist users. The premise is that speech recognition can be mapped to a users most commonly used software functions, in a sense becoming verbal shortcut keys. The survey will begin with video examples of how speech recognition may be used in AutoCAD and PHOTO-PAINT to complete a short task. After viewing these examples, sample processing states and error correction scenarios will be presented. These examples will show different cursor feedback mechanisms to assist in resolving these states.

Video - Questions

Now that you have had an opportunity to experience speech recognition used with these graphics applications, could you see yourself using speech recognition while working with your preferred graphics application?

How else could you see speech recognition being utilized in graphics applications?

Do you have any other suggestions on how this may be implemented?

Note: The following scenarios allow for a 1 second delay prior to showing any visual feedback. This is based on the design rule that any spoken command that can be processed in less than one second will not show visual feedback to the end user.

Scenario 1 – Speech input being processed

"In this scenario, the speech recognition software is processing an input signal to identify its contents. We will now view four versions of cursor-based feedback while the process is completed."

Version Preference

☐ Version 1 – No feedback	☐ Version 4 – Cursor with animated modifier
☐ Version 2 – Cursor colour change	
☐ Version 3 – Cursor blinking	

Subject comments:

Scenario 2 – Speech input unrecognized

"In this scenario, the speech recognition software is processing an input signal and cannot identify its contents. We will now view four versions of cursor-based feedback while the process is completed."

Version Preference

☐ Version 1 – No feedback	☐ Version 4 – Cursor with animated modifier
☐ Version 2 – Cursor colour change	
☐ Version 3 – Cursor blinking	

Subject comments:

Scenario 3 – Speech input being processed

"In this scenario, the speech recognition software will identify the input signal and waits for the computer to process the instruction. We will now view four versions of cursor-based feedback while the process is completed."

Version Preference

☐ Version 1 – No feedback	☐ Version 3 – Cursor with animated modifier
☐ Version 2 – Cursor colour change	☐ Version 4 – Cursor with two animated modifiers

Subject comments:

Scenario 4 – Speech input recognized, software application non-responsive -A

"In this scenario, the speech recognition software processes and identifies an input signal. Additional prompting is required to complete the software command. We will now view four versions of cursor-based feedback to assist in completing the software command."

Version Preference

☐ Version 1 – No feedback	☐ Version 3 – Cursor blinking
☐ Version 2 – Cursor colour change	☐ Version 4 – Cursor blinking with text prompt

Subject comments:

Scenario 5 – Speech input recognized, software application non-responsive -B

"In this scenario, the speech recognition software cannot identify the first input signal but does the second. Additional prompting is then required to complete the software command. We will now view four versions of cursor-based feedback to assist in completing the software command."

Version Preference

☐ Version 1 – No feedback	☐ Version 3 – Cursor with animated modifier
☐ Version 2 – Cursor with modifier	☐ Version 4 – Cursor blinking with text prompt

Subject comments:

Scenario 6 – Speech input recognized, multiple responses

"In this scenario, the speech recognition software doesn't recognise the initial input signal but does identify the second input signal which has multiple responses. We will now view five versions of cursor-based feedback while the process is completed."

Version Preference

☐ Version 1 – No feedback	☐ Version 4 – Cursor with menu
☐ Version 2 – Cursor colour change	☐ Version 5 – Cursor with YES/NO prompts
☐ Version 3 – Cursor blinking with modifier	

Subject comments:

Open Questions

The versions presented used speech recognition as “verbal shortcut keys”; do you find this a good implementation of speech recognition?

Do you find that cursor feedback can present enough information to suitably assist a user with speech recognition?

Appendix E – Survey Data

The data gathered during user surveys are compiled below.

Self-evaluated Software Experience

Participant	AutoCAD	PHOTO-PAINT	Other Graphics Software	Speech
1	8	0	2	0
2	10	0	0	0
3	7	2	3	2
4	3	7	10	10
5	4	10	10	3
6	3	9	10	3
7	2	9	10	5
8	1	2	7	0
9	0	1	10	4
10	1	1	8	2
11	1	0	3	0
12	0	0	4	0
13	3	9	10	1
14	3	10	7	0

Scale: **0** means 'you have never seen a demo or used this product' and **10** means 'you are a power user'

Cursor Feedback Scenario Preferences Scores

Scenario One	Scenario Two	Scenario Three
0: No Feedback 5: Colour Change 6: Blinking 3: Animated Modifier	0: No Feedback 2: Colour Change 1: Blinking 11: Animated Modifier	0: No Feedback 2: Colour Change 9: Animated Modifier 3: Two Animated Modifiers
Scenario Four	Scenario Five	Scenario Six
2: No Feedback 2: Colour Change 5: Blinking 5: Blinking & Text Modifier	0: No Feedback 2: Static Modifier 0: Animated Modifier 12: Blinking & Text Modifier	0: No Feedback 1: Colour Change 4: Blinking with Modifier 6: Radial Menu 3: Yes/No Voice Prompt

Yes/No Style Questions

Question 1)

“Now that you have had an opportunity to experience speech recognition used with these graphics applications, could you see yourself using speech recognition while working with your preferred graphics application?”

93% (13/14) responded Yes

7% (1/14) responded No

Question 2)

“The versions presented used speech recognition as ‘verbal shortcut keys’, do you find this a good implementation of speech recognition?”

100% (14/14) responded Yes

Question 3)

“Do you find that cursor feedback can present enough information to suitably assist a user with speech recognition?”

100% (14/14) responded Yes

Participant Occupation Titles

Architect

Architectural Technologist

Project Manager - Construction

Software Program Manager

Software Program Manager

Sr. User Experience Designer

Sr. User Experience Designer

UI Designer

UI Graphics Designer

UI Standards Specialist

Usability Specialist

Usability Specialist

User Experience Designer

Visual Interaction Designer

Appendix F – Thesis Presentation Slides

Speech Recognition and Cursor Feedback in Graphics Software Applications

- Explore the value and viability of automatic speech recognition (ASR) as an adjunctive input mode for computer aided design (CAD) and graphics creation software
 - The integration of ASR into the user interface in such a method that enhances workflow and maintain focus on the design object
 - Reduce task completion time
 - invoke commands, access features/menus buried in the interface, and input alpha-numeric data
 - reduce hand movement between input devices
- Address the most problematic integration issue, the identification and handling of recognition errors

Overview of Approach

- Review of speech recognition history
- Review of speech recognition technology
- Completed an analysis of prior studies of the integration of ASR as an adjunctive input mode including within graphics software
- Review commercial applications of ASR in CAD & graphics software
- Synthesis of design issues related to the integration of ASR within a GUI application derived from previous studies and commercial applications
- Designed and created the case study prototype and testing methodology
- Completed survey based on prototype
- Analyzed results against initial hypotheses
- Developed future research directions and conclusions

Speech Recognition History

ASR is in development for 50+ years

1950s – 1960s hardware and analysis progression

- 1971 - Defence Advanced Research Projects Agency (DARPA) established the Speech Understanding Research (SUR) programme

- 1980s - hardware advances increased cpu speeds and vocabulary sizes

- use in manufacturing begins

- 1990s - personal computers prompted the release first of discrete and then continuous recognition dictation software

- "I wrecked a nice beach." or "I recognize speech."

- interactive voice response systems

- 2000s - refinements to accuracy

- movement to embedded application use

Speech Recognition Technology

Speech Recognition Steps

- Speaker independent versus speaker dependant (enrolment)

- Discrete versus continuous speech recognition (250ms pause between discrete words)

- Automatic voice recognition (AVR) (speaker verification and identification)

- Spoken Output and Text-to-Speech (TTS)

- Pre-recorded speech - concatenated

- Synthesized speech dynamically converts TTS

- Modes - dictation and command & control

Studies of the integration of ASR to 1995

- Focus on sole input versus adjunctive input mode
- Focus on "hands-busy"/"eyes-busy" applications
- Bolt (1980) "Put That There" speech & gesture
- Leatherby and Pausch (1992) - 15% task time reduction when ASR used with shortcuts/keyboard and mouse
 - Short-cuts that required one hand removed from keyboard benefited the most
- Karl et al (1993) - command activation during word processing, task times reduced by 18%
- Nishimoto - (1995) task time reduced by 18% and mouse travel reduced by 47%

Studies of the integration of ASR with graphics software to 1995

Observations:

- ASR as an additional input mode allowed users to: 1) maintain focus on the work area, 2) reduce mouse movement on the screen, and 3) reduce task completion time;
- Short ASR commands allow users to easily perform other tasks while speaking the commands;
- Small vocabularies can be perfectly adequate for limited domain communication;
- Refine command vocabularies to maximize speech recognition accuracy;
- Tailor the command vocabulary to recognize all words associated with a particular command e.g., "cut," "erase," and "remove";
- Develop command names that are easily associated with the functions they perform;
- Make it easy to perceive system status;
- Make it easy to turn the microphone on and off to reduce misrecognition incidences due to environmental noise.

Commercial CAD & Graphics Software

Commercial attempts for CAD, directed to assistive needs - IN CUBE, VoiceEZcad

IMSI consumer market Turbo Sketch, Talk & Draw Architect, TurboCAD

- Enabled application commands but was not developed explicitly to enhance user experience;
- Purchased ASR license with software from Corel Corporation;
- Marketing novelty during media hype;
- Created their own distribution of speech dictation applications.

Think3 & thinkdesign

- All commands speech enabled, provides a speech toolbar;
- Learning mode - TTS feedback of tool selected;
- Confirmation mode - repeat selection
- Uses the Microsoft speech recognition engine.

No graphics software applications ASR enabled

Design Elements of Speech Recognition Integration

Transient nature of speech

- Interpretation errors
- Types of errors rejection, injection, substitution
- What does a pause in ASR activity mean - Wait? Repeat?
- Requires prompt feedback mechanisms
- Good error display and handling

User Expectations - science fiction depiction of sentient machines

- Poor experiences of the past continue to hinder ASR adoption

Inexperience with modality - what can be said?

- Addressed best by mimicking naming conventions of visible interface

ASR most beneficial as one on multiple input modes

- ASR integration continuum within multimodal interfaces - enabled/aware/centric
- Use OS guidelines as well as GUI standards
- Usable = effective, efficient, and satisfying

Case Study and Testing Methodology

Drawn from research insight

Sample tasks and scenarios were created to test assumptions

Phase I – Integrated ASR into AutoCAD & Corel PHOTO-PAINT using Dragon NaturallySpeaking 6 Pro

- Developed sample tasks ASR “verbal shortcut keys” were enabled by mapping to commands
- Video screen captures created from ASR enabled software – narrated & ASR enabled

Phase II – Created video with error states and proposed cursor feedback solutions

- 1 second delay before displaying any feedback
- Extended/protracted error states to allow users to easily witness cursor enhancements

Procedure – Survey participants reviewed the speech-enabled prototypes and the enhanced cursor feedback error condition scenarios and evaluated and commented on each

Survey - ASR Prototype & Scenarios

ASR Enabled Prototypes

- AutoCAD 2002 LT & Corel PHOTO-PAINT sample tasks

Survey Materials

Enhanced Cursor Feedback Scenarios -- 6 for participants to review

- Basic = A) response to processing B) response to unrecognized C) response to wait state
- D) Advanced = integration where CAD could not process command
- E) Advanced = integration where graphic software was in a state that did not allow the ASR to process the input
- F) Advanced = integration where multiple answers to the input with different choice scenarios

Survey Results

Participant Profile

- 12/14 ranked within the power user to expert categories for either CAD or graphics applications
- >50 % of participants had ASR familiarity

ASR Integration

- 93 % of the participants would use ASR as an additional input mode for their preferred graphics application
- 100% of the participants believed that the proposed use of verbal shortcut keys was a good implementation of ASR
- Participants felt that ASR can reduce hand movement between pointing device and keyboard, especially for those using pen tablets
- Participants perceived value in using ASR to input alpha-numeric content
- Participants deemed experts in AutoCAD believed that inputting alpha-numeric content into a CAD application with ASR would be exceptionally useful
- 43% expressed potential uneasiness with using ASR within an open office environment

Survey Results

Cursor Feedback Responses

100% of the participants believed that cursor feedback would provide enough information to assist users with ASR while using a graphics application;

- Participants commented that the cursor was the focal point of their attention while completing tasks and as such, was the logical place to display ASR related feedback;
- Participants expressed that cursor enhancements should be customizable;
- Scenario Preferences:

- Scenario 1 preference of simpler feedback mechanisms
- Scenario 2 clear choice due to familiarity with modifier - question mark
- Scenario 3 clear choice due to familiarity with modifier - wait state hourglass
- Scenario 4 blinking preference with 2 no feedback selections
- Scenario 5 text modifier clear choice - (active Help system)
- Scenario 6 review choice versus repetition options - radial menu highest response, but not decisive

Cursor Feedback Integration

Cursor Feedback Evaluation

- Each type of cursor modification i.e., colour change, blinking and appended modifier, utilized was selected as appropriate to convey the intended meaning within one or more of the scenarios
- A direct correlation was observed between participant selection of increased supplementary information on the cursor and increased assistance required to resolve the error or system state
- The clarity or familiarity of an appended modifier led to quicker participant identification of meaning and greater satisfaction
- User's feedback preferences were not influenced by their expertise with graphics or CAD software
- Error state modifiers must highly visible or they could go unnoticed in a complicated CAD drawing

Design Refinements & Recommendations

1 Investigate a new style of interaction with the interface when using ASR to input alpha-numeric data

Input data by simply positioning the pointer device over a control or input field. The user would identify the selected control by the cursor invoked pliancy.

2 Optimize the ASR lexicon for greatest recognition accuracy

Good design practice ensures that the names given to the speech commands are distinctive and do not sound too similar to other commands. This helps reduce recognition substitution errors.

Separate modes for Dictation and Command & Control

3 Refine the cursor feedback visual style and assess when different styles should be applied

Exceptional conditions or time sensitive information should be presented to attract user attention

Consider that novice and expert users comprehend meaning with differing levels of display or feedback

Extended assistance i.e., an "active" help system, appended to the cursor was identified as an exceptional way to convey feedback and instruction

Cursor enhancement modifiers, i.e., colour change, blinking and appended modifiers, and Help text, could be tested in a similar fashion to those that were presented in this study and are not dependant on a functioning ASR prototype

4 Allow users to customize both cursor feedback styles and ASR preferences as well as make them portable

Select and control the colours and types of cursor modifications to reflect their personal tastes as well as to ensure that the cursor enhancements are presented clearly in the user's software environment

Creating the Next Test Model

Speech enablement can be accomplished through the use of a speech vendors software development kit (SDK)

Dragon NaturallySpeaking SDK allows software developers to speech-enable programmes using the Dragon NaturallySpeaking application programming interface (API)

- Enhanced cursor feedback could be created through the use of the Graphics Device Interface (GDI)
- An example, Time Cursor appends the system time to the lower right corner of the cursor
- This approach allows the ASR software to append a modifier to the target application cursor when error or system status changes occur without requiring the target application vendor to create and maintain specific cursors for this purpose

Summary – Adjunctive ASR

Hypothesis – ASR as adjunctive input for CAD and graphics software

The target user proposed was a domain professional who would use ASR as an adjunctive means to complete command and control functions within the application. The focus was to enhance the user's productivity by reducing mouse movement within the software user interface as well as hand movement between the mouse or pointer device and the keyboard. The proposed method of ASR integration was endorsed by 100% of the survey participants with 93% stating that they would use ASR within their preferred graphics application were it available.

Summary – Enhanced Cursor Feedback

Hypothesis – Enhanced Cursor Feedback to display error states

ASR error identification and handling was identified throughout the research literature as the primary interface enhancement necessary to increase user satisfaction and acceptance. Current products provide feedback in a separate dialogue that can be placed anywhere on the user interface. This style of feedback is not optimal for users of CAD or graphics applications as their point of focus throughout their task completion is the cursor. The survey results support the proposed use of ASR and enhanced cursor feedback within CAD or graphics applications and indicate that this proposed proof of concept was successful. The development of a functioning prototype with current technology and a refined set of cursor modifiers would be the logical next step in the evolution of this concept.